OXFORD CDE

Journal of Computational Design and Engineering, 2022, **9**, 2103–2133

DOI: 10.1093/jcde/qwac100
Advance access publication date: 20 September 2022
Research Article

Differential evolutionary cuckoo-search-integrated tabu-adaptive pattern search (DECS-TAPS): a novel multihybrid variant of swarm intelligence and evolutionary algorithm in architectural design optimization and automation

Hwang Yi [1,*] and Inhan Kim [2]

[1]Department of Architecture, Ajou University, 206 World cup-ro, Yeongtong-gu, Suwon-Si, Gyeonggi-do 16499, South Korea
[2]Department of Architecture, Kyung Hee University, 1732 Deogyeong-daero, Yongin-si, Gyeonggi-do 17104, South Korea
*Corresponding author. E-mail: hwy@ajou.ac.kr

Abstract

One of the critical limitations in architectural design optimization (ADO) is slow convergence due to high-dimensional and multiscale variables. For the rapid and optimal digital prototyping of architectural forms, this paper proposes a novel metaheuristic optimization technique that hybridizes standard low-level algorithms: the differential evolutionary cuckoo-search-integrated tabu-adaptive pattern search (DECS-TAPS). We compared DECS-TAPS to 10 major standard algorithms and 31 hybrids through 14 benchmark tests and investigated multi-objective ADO problems to prove the computational effectiveness of multiple algorithm hybridization. Our findings show that DECS-TAPS is vastly efficient and superior to the covariance matrix adaptation evolution strategy algorithm in multifunnel and weak structural functions. The global sensitivity analysis demonstrated that integrating multiple algorithms is likely conducive to lowering parameter dependence and increasing robustness. For the practical application of DECS-TAPS in building simulation and design automation, Zebroid—a Rhino Grasshopper (GH) add-on—was developed using IronPython and the GH visual scripting language.

Keywords: metaheuristic optimization, hybrid algorithm, architectural design optimization, building performance simulation

1. Introduction

1.1. Architectural design optimization

Environmental performance-based building design (EPBD) is a non-prescriptive and evidence-driven approach to achieving desirable building performance, including improved thermal comfort, daylighting, energy saving, and acoustics (Loftness *et al.*, 2005; Hensen & Lamberts, 2011). EPBD has drawn increasing attention in architecture because early-phase design decision making has the most significant impact on the full spectrum of environmental building performance (Kohler & Moffatt, 2003; Kolarevic & Malkawi, 2005; Ataman & Dino, 2021). The performance-based understanding of architectural practices places a high priority on the work of optimal form finding as a basis for sustainable building (Wortmann, 2019a). Although architectural design problems have been less investigated in the study of mathematical optimization, the optimal design of environmental building forms often poses complex engineering problems, entailing solid decision support gathered from interdisciplinary scientific knowledge, including a greater level of technical expertise (Shi *et al.*, 2016; Kheiri, 2018).

For the past few years, collective commitments to architectural design optimization (ADO) in EPBD have led to remarkable progress in innovating design methodologies and developing design computation tools (Waibel *et al.*, 2019; Wortmann, 2019b; Yi *et al.*, 2019; Li *et al.*, 2020). The application of building performance simulation and optimization solvers on designers' tools has allowed the active use of metaheuristic optimization algorithms (MHOAs) in design areas, such as building layout planning, human behavior organization, or spatial topology of building components (Lanza Volpe, 2018; Song & Sun, 2021). Yi and Yi (2014) and Dino (2016) presented the automated generation of optimal 3D building massing. Furthermore, the growing awareness of the dominant contribution of indoor activities and spatial settings to high-performance buildings led to the study of optimal thermal zoning or behavior-driven space configuration in the context of architectural design practice (Yi, 2016, 2020). Various designer-friendly solver toolkits, including GenOpt (Wetter, 2018), MOBO (Hasan & Palonen, 2013), Galapagos, Goat (Flöry *et al.*, 2018), Octopus, TopOpt (DTU, 2018), and Opossum (Wortmann, 2017), have been built and introduced in building practice along with

Received: March 21, 2022. **Revised:** August 31, 2022. **Accepted:** September 16, 2022

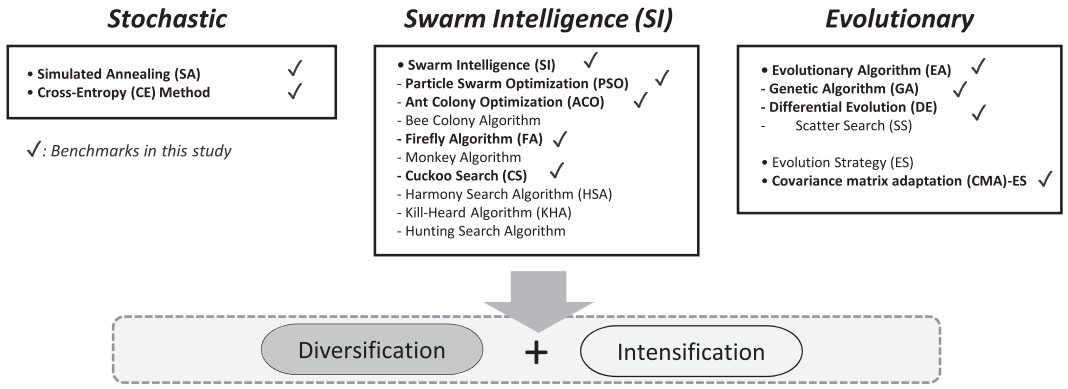


Figure 1: MHOA categorization.

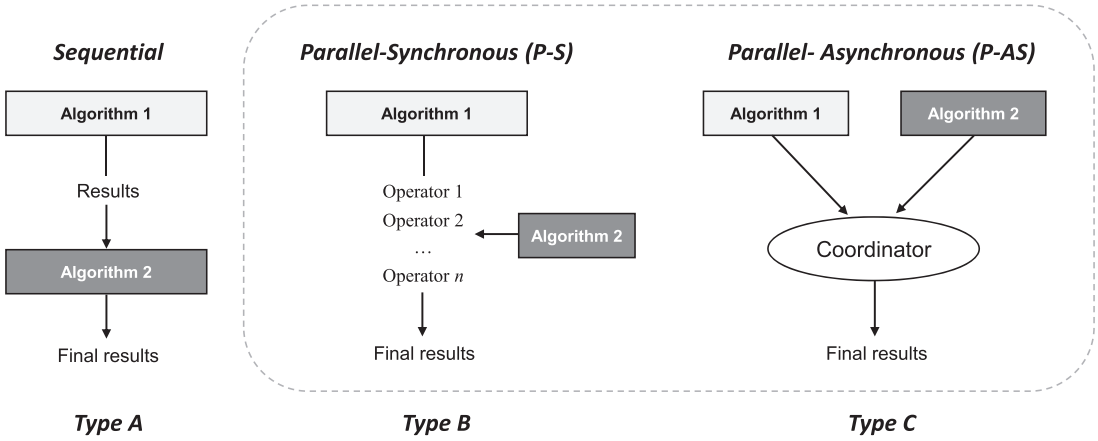


Figure 2: Types in MHOA hybridization recipes (Xu & Zhang, 2014).

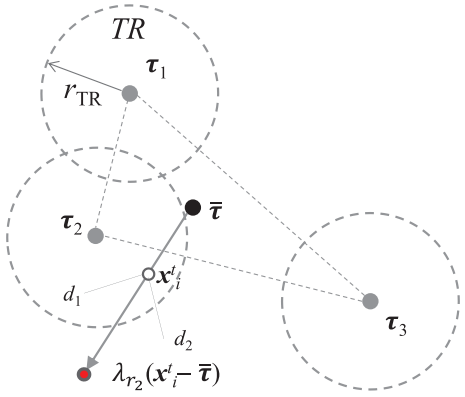


Figure 3: Definition of tabu areas (Yi *et al.*, 2019).

the widespread usage of architectural design simulation. Also, as many different mathematical algorithms have become available in architecture, an extensive analysis of the algorithm performance in EPBD has been made by Waibel *et al.* (Li *et al.*, 2020) and Wortmann (Song & Sun, 2021).

Efforts to improve ADO are still ongoing and evolving. Nevertheless, further elaborations are required for the algorithms because high-dimensional variables with different data scales in building geometry optimization often delay the design process (Ciccozzi *et al.*, 2018). Consequently, it is imperative to develop a more efficient and accurate algorithm in ADO. The most widely used techniques for solving building design problems currently are graph theory, simulated annealing (SA), and genetic algorithm (GA). However, SA suffers from premature convergence, and the single-solution-based and undirected stochastic operation is inefficient for solving complex multimodal problems. In GA, the exhaustive repeated evaluation of populations and delayed optimization thereof have been noted in the literature (Shi, 2010). In the EPBD optimization practice, a single function evaluation may take minutes to hours of performance simulation (e.g., energy, daylight, and airflow). The slow convergence rate is the biggest obstacle that architects face during the early-stage design development (Yi *et al.*, 2019). On top of that, orchestration of GA and SA parameters calls for skilled experience, as coarse evolution is ill-suited to dealing with data scalability and solution robustness (Katoch *et al.*, 2021). Parallel computing on multiprocessors or problem decomposition can help overcome this limitation immediately (Brunetti, 2015; Su & Yan, 2015); however, the computational expenses may increase exponentially, and they might not always ensure satisfactory solutions. Moreover, high-level expertise is required to decompose an ADO problem. Instead, the hybridization of well-established algorithms can provide much cheaper methods to resolve those challenges.

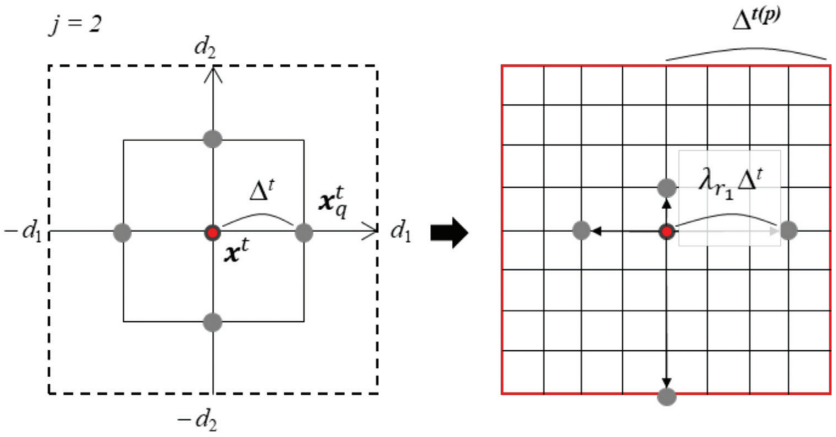


Figure 4: Mesh adaptation strategy (Yi *et al.*, 2019).

1.2. Brief review of metaheuristic optimization algorithms and hybridization

Metaheuristics refer to high-level numeric procedures based on iterative black-box learning—search and evaluation with little prior information of objective formulae or problem domains. MHOAs are thus designed to approximate optimal solutions non-deterministically. Many different MHOAs, variants, and hybrids have been introduced (Abdel-Basset & Abdel-Fatah, 2018). MHOA families can be categorized according to space exploration mechanisms. The two primary criteria for this classification are whether or not (i) an algorithm deals with multiple candidates simultaneously and (ii) trials and evaluations depend on probabilistic measures.

In most MHOAs (Fig. 1), randomness is an essential ingredient of algorithm construction because a certain degree of randomness extraordinarily strengthens the ability of solution search. Almost every algorithm attempts to encode randomness in a particular manner, balancing the two procedural components—diversification (global exploration) and intensification (local exploitation). For example, SA is based on random greedy search, and, to complement Markov chain hill climbing, it allows random moves based on the Boltzmann distribution. Cross-entropy (CE) also builds on the stochastic Monte Carlo simulation, and a solution update is inferred by random samples generated from a particular underlying probability density function for rare-event estimation. In CE, candidates are parametrized by means and variances of feasible distributions, and the important sampling technique is used to measure sampling similarity (Kroese *et al.*, 2006).

Standard SA and tabu search (TS) are a subset of trajectory-based algorithms (Shehab *et al.*, 2017) because they generally breed a single candidate at every iteration. However, when solving multimodal problems, many recent studies have revealed that population-based fitness updates perform much more robustly (Mohamed, 2017; Ahmed *et al.*, 2021). In evolutionary algorithm (EA) to support collective exploration, GA (Holland, 1975) is among the most well-known branch techniques, including evolutionary strategy (ES) and genetic programming. Based on the natural principle of "survival of the fittest," individual candidates in GA evolve through generative choices and breeding of improved attributes—the operations of crossover, reproduction (recombination), and mutation. Classic GAs are based on the binary form of numbers (genotype), while ES supports real numbers (phenotype). During the evolutionary stages at the phenotype levels, ES focuses more on the stochastic mutation of individuals, leaving out the crossover. Gaussian distributions and related parameters are generally involved in controlling the ES process.

On the other hand, to solve ill-conditioned/non-separable objective functions, the covariance matrix adaptation-evolution strategy (CMA-ES) was developed and successfully applied to various multidimensional and multimodal optimization problems (Hansen & Ostermeier, 1996). It effectively exploits the fitness landscape of principal components on a transformed coordinate using multivariate distribution. The CMA-ES and its variants are often considered one of the most successful metaheuristics. Nonetheless, (i) algorithmic and computational complexity due to eigendecomposition, (ii) complicated stopping criteria, and (iii) several constraints concerning hyperparameter setup (Krause *et al.*, 2016; Jin *et al.*, 2020) are the main drawbacks. These often cause significant errors in large-scale problems (Caraffini *et al.*, 2019; Wortmann, 2019b). In particular, the biased determination of an initial learning rate (step size) often leads to inferior candidates and local stagnation because it is originally based on the cumulative step-size adaptation strategy (Wang *et al.*, 2019). Another computational barrier in algebraic stability is to keep covariance matrices positive definite.

It can be said that the EA operations are a reinforced manipulation of the entry population, lacking "internal" cooperative/social intelligence (Juang, 2004). Swarm intelligence (SI) algorithms have been developed alternatively by taking advantage of physical or biological paradigms found in nature. Particle swarm optimization (PSO) mimics the movement of entities in a migratory flock of birds or fish schooling (Banks *et al.*, 2007), and cuckoo search (CS) (Yang & Suash, 2009) and firefly algorithm (FA) (Yang, 2009) were inspired by the ecological behavior of bird species. The ant colony optimization (ACO) algorithm incorporates the collective communication system of food-finding ants (Dorigo *et al.*, 1996; Dorigo *et al.*, 2006). The whale optimization algorithm (WOA) (Mirjalili & Lewis, 2016) and its improved variant hybridizing DE (Bozorgi & Yazdani, 2019) have been recently proposed to represent the Humpback whale's prey-chasing behavior through a population update with a spiral-shape function. The unusual mating pattern of Scottish deer has led to the development of the red deer algorithm (RDA) that employs an improved crossover method based on the elitism of population update (Fathollahi-Fard *et al.*, 2020). Swarm-based algorithms are popular in hybridization due

Table 1: Benchmark functions and list of simulation tests.

ID	n_d	Test function	Type	$f(x)$	$[v_{min}, v_{max}]$	$f(x*)$
T1	2	Ackley (f_1)	M, I, AS	$-20\exp[-0.2\sqrt{(x_1^2+x_2^2)/2}]$ $-\exp[(\cos 2\pi x_1+\cos 2\pi x_2)/2]+e+20$	$[-5.0, 5.0]^{2}$	0
T2	2	Drop wave (f_2)	HM, I	$-\frac{1+\cos(12\sqrt{x_1^2+x_2^2})}{0.5(x_1^2+x_2^2)+2}$	$[-5.12, 5.12]^{2}$	−1
T3	2	Levi N.13 (f_3)	HM, I, AS	$\sin^2 3\pi x_1+(x_1-1)^2(1+\sin^2 3\pi x_2)$ $+(x_2-1)^2(1+\sin^2 2\pi x_2)$	$[-10.0, 10.0]^{2}$	0
T4	2	Michalewicz (f_4)	M, S	$-\sum_{i=1}^{2}\sin(x_i)[\sin(\frac{ix_i^2}{\pi})]^2$	$[0.00, 3.14]^{2}$	−1.8013
T5	2	Eggholder (f_5)	M, I, WS	$-(x_1+47)\sin(\sqrt{\lvert x_2+0.5x_1+47\rvert})-$ $x_1\sin(\sqrt{\lvert x_1-(x_2+47)\rvert})$	$[-512.00, 512.00]^{2}$	−959.64
T6	5	5D Ackley (f_1)	M, I, AS	$-20\exp[-0.2\sqrt{\sum_{i=1}^{5}x_i^2/5}]$ $-\exp[\sum_{i=1}^{5}\cos 2\pi x_i/5]+e+20$	$[-5.00, 5.00]^{5}$	0
T7	5	Styblinski-Tang (f_6)	M, S, HC	$0.5(\sum_{i=1}^{5}x_i^4-16x_i^2+5x_i)$	$[-5.00, 5.00]^{5}$	−195.83
T8	5	Bent cigar (f_7)	U, I, HC	$x_1^2+10^6\sum_{i=2}^{5}x_i^2$	$[-1.0E+2, 1.0E+2]^{5}$	0
T9	5	Griewank (f_8)	M, S, WS	$\sum_{i=1}^{5}\frac{x_i^2}{4000}-\prod_{i=1}^{5}\cos\frac{x_i}{\sqrt{i}}+1$	$[-6.00E+2, 6.00E+2]^{5}$	0
T10	10	Rastrigin (f_9)	HM, S, AS	$10d+\sum_{i=1}^{n_d}[x_i^2-10\cos(2\pi x_i)]$	$[-5.12, 5.12]^{10}$	0
T11	30				$[-5.12, 5.12]^{30}$	
T12	50	Sphere (f_{10})	U, S, AS	$\sum_{i=1}^{n_d}x_i^2$	$[-1.0E+3, 1.0E+3]^{50}$	0
T13	50	Rosenbrock (f_{11})	M, I, LC, WS	$\sum_{i=1}^{n_d-1}[100(x_{i+1}-x_i^2)^2+(x_i-1)^2]$	$[-5.00, 10.00]^{50}$	0
T14	50	Lunacek bi-Rastrigin (f_{12})	HM, S, WS	$\min\{\sum_{i=1}^{n_d}(x_i-\mu_1)^2, dn_d+s\sum_{i=1}^{n_d}(x_i-\mu_2)^2\}+$ $10\sum_{i=1}^{n_d}[1-\cos(2\pi(x_i-\mu_2))]$	$[-5.00, 5.00]^{30}$	0

Note: n_d – dimension, U – unimodal, M – multimodal, HM – highly multimodal, S – separable, I – inseparable, LC – low-conditioning, HC – high-conditioning, AS – adequate global structure, and WS – weak global structure.

to their structural simplicity. Although each claims difference, their search mechanisms bear similarities in many ways. For instance, CE, CS, and ACO take elitist selection like GA. Continuous ACO assigns probability distributions to populations and fitness values similar to CE. PSO, CS, and FA are drawn from the idea of successive random walks, and FA can be considered a special case of PSO. The individual recognition of a spatial position in PSO corresponds to the response of a population element to the light intensity in FA. Also, by employing Lévy distributions as a substitute for the velocity, PSO can be converted to CS.

The hybridization of established MHOAs is an active topic of research with considerable interest in various areas (Il-Seok *et al.*, 2004; Amaran *et al.*, 2014; Li & Gao, 2016; Li *et al.*, 2019; Ficarella *et al.*, 2021). Since different parameters, iterative durations, and operating frequencies of heterogeneous mathematical components render an individual MHOA uniquely characterized (El-mihoub *et al.*, 2006; Xu & Zhang, 2014), numerous recipes of mixed algorithms have been extensively suggested to improve computational performance and solution quality. Talbi (Talbi, 2002) divided the various synthesis methodologies into three primary schemes: (i) sequential, (ii) parallel-synchronous (P-S), and (iii) parallel-asynchronous (P-AS). Sequential is the simplest type wherein two or more algorithms perform in a hierarchical order so that a solution from the former becomes the input to the latter. In the P-S type, the subalgorithm operators are integrated into the main algorithm, whereas two algorithms are concatenated through a third-party joint in P-AS (Fig. 2).

In early hybrid MHOAs, well known are TS-integrated SA (Osman & Christofides, 1994), extreme-point TS (Blue & Bennett, 1998), embedding GA into the SA framework (GA-SA) (Mori *et al.*, 1996; Li *et al.*, 2002; Soke & Bingul, 2006), and PS-based neighbor search in the course of SA's iteration (PS-SA) (Hedar & Fukushima, 2004; Vasant & Barsoum, 2010). In this approach, evolutionary schemes were preferred for computational collaboration, such as

Table 2: Benchmark hybrid algorithms.

	PS	DE	CMA-ES	PSO	ACO	CS	FA	CE	GA	SA
TS	**B** (Audet & Dennis, 2006)					**B** (Terki & Bouber-takh, 2021)			**B** (Zhang *et al.*, 2013)	**B** (Osman & Christofides, 1994)
PS		**C** (Zhu, 2011)				**B**	**B** (Tawhid & Ali, 2016)		**B** (Nassef *et al.*, 2000)	**B**
DE			**A** (Mohamed *et al.*, 2017)	**C** (Araújo & Uturbey, 2013)		**B** (Wang *et al.*, 2012)	**C** (Zhang *et al.*, 2016)		**C** (Nithya & Jeyachidra, 2021)	**B** (Xiaobing *et al.*, 2021)
CMA-ES				**B** (Xu *et al.*, 2019)						
PSO					**C** (Rahmani *et al.*, 2013)	**C** (Ding *et al.*, 2019)		**C**	**B** (Juang,)	**B** (Shieh *et al.*, 2011)
ACO						**B** (Zhang *et al.*, 2019)			**C** (Wang *et al.*, 2020)	
CS							**C**		**C** (Kanagaraj *et al.*, 2013)	**B** (Alkhateeb & Abed-alguni, 2019)
FA								**B** (Li *et al.*, 2019)	**B** (Rahmani & MirHassani, 2014)	**B** (Devi & Sabrigiriraj, 2019)
CE									**B** (Lopez-Garcia *et al.*, 2016)	
GA										**C** (Li *et al.*, 2002)

Table 3: Hyperparameter setting.

Parameter	MADS	SA	PSO	DE
n_p			40	40
r_l	λ_{r1} (equation 13)	$\alpha = 0.9$	K	$\mu_F^0 = 0.5$
$T_{initial}$ (K)		1E + 5		$\mu_{CR}^0 = 0.9$
T_{final} (K)		1E-5		
Parameter	**GA**	**CS**	**ACO**	**DECS-TAPS**
n_p	40	40	40	20
N_{max}	1E + 4	1E + 4	1E + 4	1E + 3
l_{tb}				25
SF				25
r_m	0.2			
r_{elite}				0.5
λ		1.5		1.5
r_{cs}		0.5		0.25
CMA-ES				
$n_p = 4 + 3\ln n$, $w_i = \ln 0.5(n_p + 1) - \ln i$ (i = 1, …, $n_p/2$)$c_c = c_\sigma = 4/n$, $c_1 = 4/n^2$, $c_\mu = 0.3n_p/n^2$, $d_\sigma = 1 + \sqrt{(0.3n_p/n)}$				

Note: n_p – population size, l_{tb} – tabu memory size, r_l – learning rate, r_{elite} – ratio of the best candidates in population, N_{max} – maximum iteration, r_m – ratio of mutation in GA, r_{cs} – ratio of CS, $K = 2r/|2 - m - \sqrt{m^2 - 4m}|$ (Eberhart & Shi, 2000; Barrera *et al.*, 2016), $r \in (0, 1)$, $m = c_1 + c_2$ ($c_1 = c_2 = 2.05$) (Clerc & Kennedy, 2002), SF – stagnation factor, c_c, c_σ, c_1, c_μ, and d_σ in CMA-ES refer to the learning rates of rank-one and rank-μ update and the damping parameter (Hansen & Ostermeier, 1996; Source Code for Module barecmaes2, 2014).

mesh-adaptive direct search (MADS)-GA (Pandian, 2011), DE-TS (Ponsich & Coello Coello, 2013), TS-GA (Garai & Chaudhurii, 2013; Zhang *et al.*, 2013), and GA-DE (Trivedi *et al.*, 2016; Nithya & Jeyachidra, 2021).

In many studies, SI and evolutionary mechanisms are important hybridization ingredients. In particular, PSO-sourced parameters are widely used as a catalyst for synthesizing advanced algorithms. For example, hybrids of PSO-GA/PS integration have received great attention, and several variants have been introduced by Ali and Tawhid (2017), Juang (2004), Kao and Zahara (2008), Kuo and Han (2011), and Sahu *et al.* (2015a). On the other hand, Sha and Hsu (Sha & Hsu, 2006) used a tabu-based memorizing technique for the population update in PSO. Variants of PSO-SA (Xia & Wu, 2006; Shieh *et al.*, 2011; Javidrad & Nazari, 2017) and PSO-ACO (Kıran *et al.*, 2012; Rahmani *et al.*, 2013; Mahi *et al.*, 2015) also exist. Growing interest in hybrid swarm-based approaches has led to the development of several novel algorithms based on CS or FA, such as PSO-CS (Fan *et al.*, 2011; Ding *et al.*, 2019), DE-CS (Wang *et al.*, 2012), GA-CS (Kanagaraj *et al.*, 2013), ACO-CS (Nancharaiah & Mohan, 2014; Zhang *et al.*, 2019), CS-SA (Alkhateeb & Abed-alguni, 2019), FA-GA (Rahmani & MirHassani, 2014), PS-FA (Sahu *et al.*, 2015b), DE-FA (Zhang et al., 2016), and GA-ACO (Wang *et al.*, 2020). A few CE-informed hybrids have also been suggested, such as GA-CE (Lopez-Garcia *et al.*, 2016) and CE-FA (Li *et al.*, 2019). Recently, RDA and WOA have been hybridized with GA and SA, such as HRDGA and HWISA, respectively (Fathollahi-Fard *et al.*, 2021). However, not all hybrids are equally successful because they often

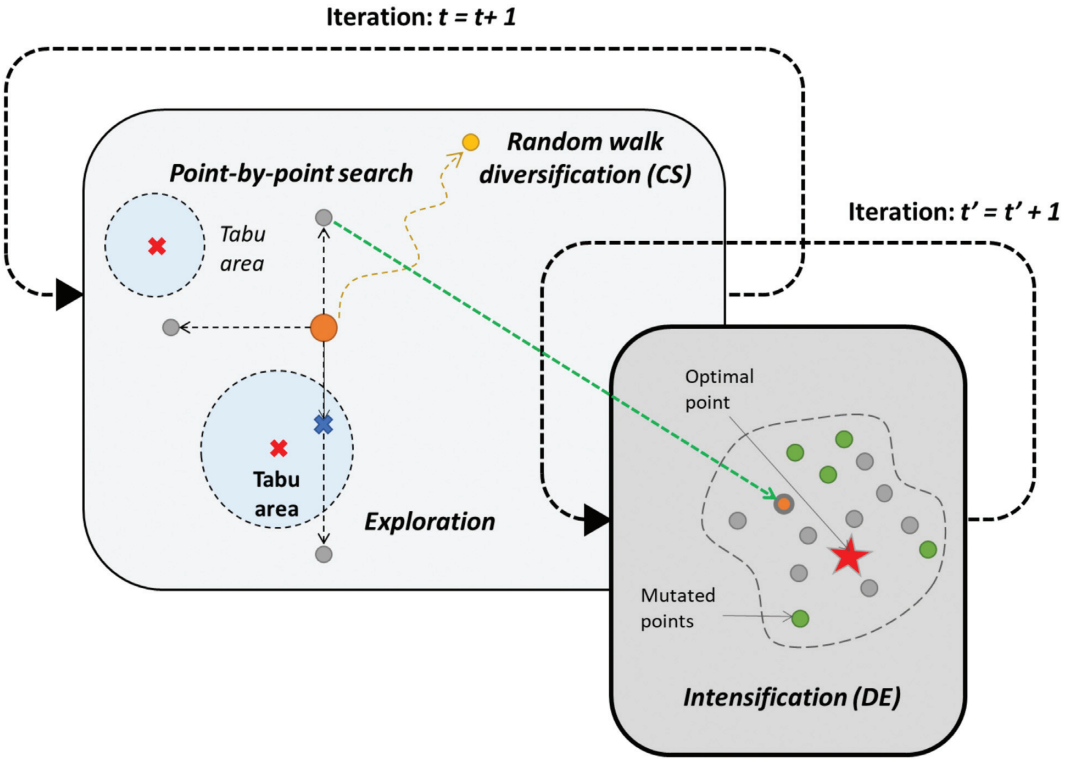


Figure 5: DECS-TAPS conceptual mechanism.

require tedious control parameter tuning depending on empirical findings from a specialized problem context (Islam *et al.*, 2021). Moreover, data formats exclusive to a specific algorithm, such as the binarization of classic GAs, may cause conflict in streamlining heterogeneous data. In some cases, conjugation of incompatible operators results in a significant tradeoff between computation speed and accuracy (Deb & Srivastava, 2012).

1.3. Motivation behind proposing a hybrid of multiple algorithms

The no free lunch theorem for optimization (Wolpert & Macready, 1997) states that there is no such universally best algorithm for general purposes. Unlike other problems, building geometry design often introduces a large number of variables. Moreover, EPBD is associated with many engineering factors of building systems. Such decision-making input does not need to be as accurate as other engineering problems. However, ADO in the early design phase of EPBD practice may have to deal with many variables with different data scales. This aspect makes standard population-based or stochastic MHOAs, such as GA, SA, or similar hybrids, ill-suited to ADO. Meanwhile, most existing hybridization strategies have been proposed in the form of pairwise combinations, and the integration of multiple algorithms for more active data exchange has been addressed relatively less despite its immense potential to mitigate parametric fragility and operational dependence on the data structure and domain scale. Deng *et al.* (2012) found that a triple combination of GA, PSO, and ACO outperforms two-way hybrids and individual MHOAs. Liu *et al.* (Liu & Fu, 2014) accomplished machine-learning integration with the PSO-CS, developing a threefold hybrid algorithm of the support vector machine (SVM) CS-PSO (CS-PSO-SVM). More recently, Fathollahi-Fard *et al.* (2018) successfully validated a hybrid of RDA, GA, and SA and other threefold algorithm combinations. These achievements have motivated our attempts to better advance existing MHOAs and hybrids by considering the conjunction of four or more low-level algorithms for introducing further functional improvement, including reducing the hyperparameter sensitivity and algorithm complexity that enable fast convergence.

1.4. Research objectives and presentation structure

This study presents the algorithm design and performance of a novel quadruple variant that intersects MADS, TS, DE, and CS: the differential evolutionary cuckoo-search-integrated tabu-adaptive pattern search (DECS-TAPS). The study's main objectives were as follows: (i) to develop a high-efficient algorithm for large-scale and multivariate architectural design problems; (ii) to validate algorithm performance and utility; and (iii) to implement the proposed algorithms in EPBD. To this end, based on our prior knowledge from Yi *et al.* (2019), we decided to mix population and non-population-based MHOAs and hypothesized that (i) the use of a non-population-based search mechanism expedites diverse exploration and is computationally more efficient, and (ii) a sequential mixture of population-based search is advantageous in error-reduced exploitation in large-scale domains. We investigated the algorithm's performance with popular test functions and case-study architectural design problems. The remainder of this paper is organized as follows: The MHOAs used in DECS-TAPS are briefly reviewed in Section 2, and improved hybridization strategies are described in Section 3. In Sections 4.1 and 4.2, DECS-TAPS is validated by comparing other pairwise hybrids through benchmark problems, and the mechanical behavior of the algorithm is also discussed. EPBD applications of the proposed hybrid are presented in Section 4.3.

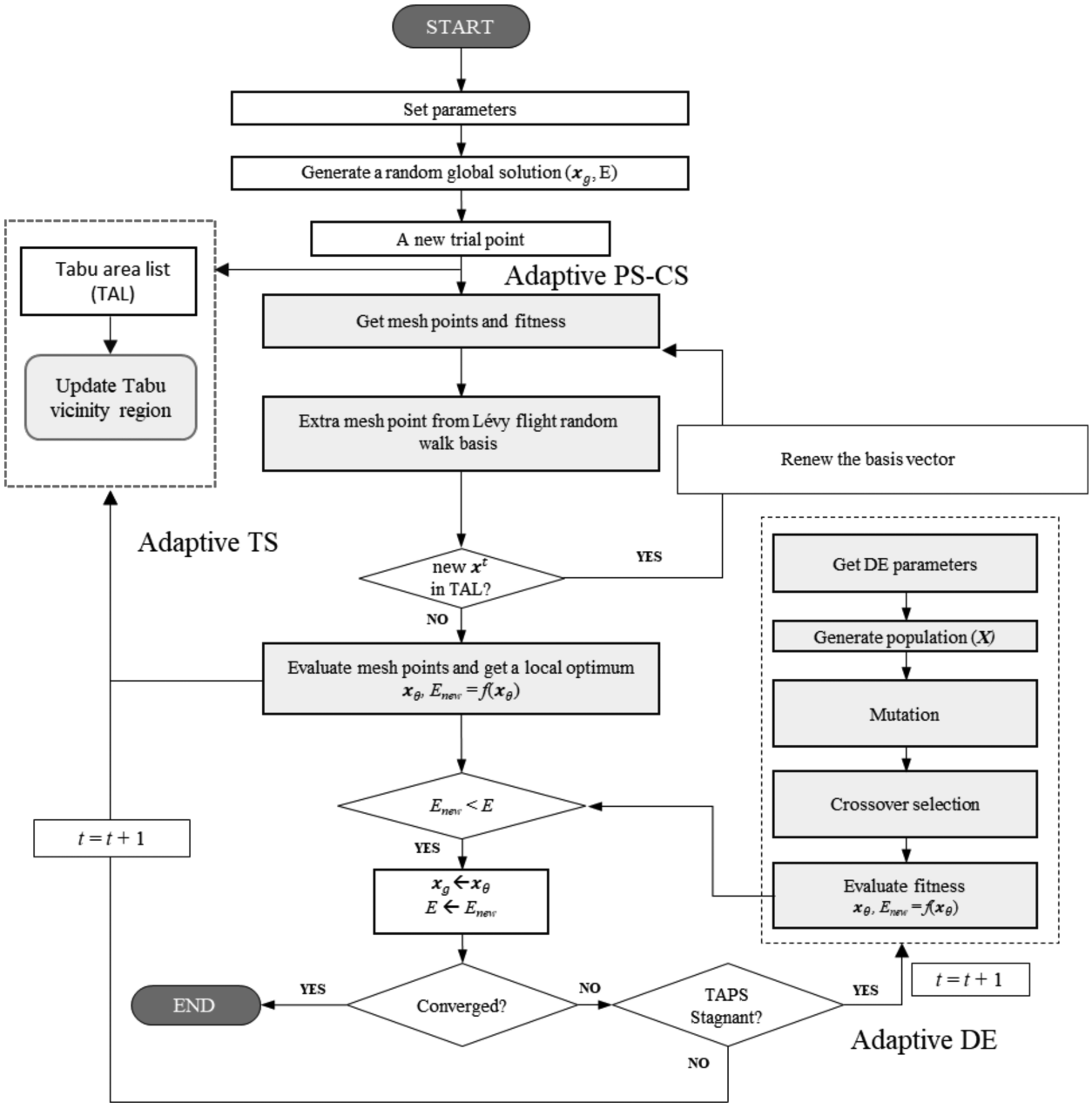


Figure 6: Algorithm flow chart (DECS-TAPS).

2. Methods and Material

Global optimization problems (GOPs) are generally represented by a triplet, (S, Ω, f), where f: $\boldsymbol{x} \subseteq S \rightarrow \mathbb{R}$ is an objective function (fitness) and S is a search space defined over a finite set of discrete variables, $x \in \boldsymbol{x}$. Ω is a set of constraints among the variables, by which S_Ω denotes the set of feasible solutions. Solving a GOP is to find at least one global optimal solution, $\boldsymbol{x}^* \subseteq S_\Omega$, which is expressed as

$$f(x^*) \leq f(x), \forall x \subseteq S_\Omega. \tag{1}$$

Swarm/population-based algorithms (PBAs), including DE, PSO, and ACO, extend the variable vectors $\boldsymbol{x}$ to a space of variable swarm $\boldsymbol{X} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m]^T$, $\forall \boldsymbol{x}_i \in \boldsymbol{X}$ $(i = 1,\ldots, m)$, and $\boldsymbol{x}_i = [x_{i1}, \ldots x_{in}]^T \in \mathbb{R}^n, \forall x_{ij} \in \boldsymbol{x}_i$ $(j = 1,\ldots, n)$. $\boldsymbol{x}_i$ is referred to as the i-th individual, where m and n are the population size and the number of the variables determined in a problem, respectively. At each iteration, the population is evolved toward a better solution (x'), such that

$$x' = arg\min f_\Omega(x), \forall x \subseteq S_\Omega. \tag{2}$$

2.1. Spatial TS

TS employs a rule of prohibition progress search process that encourages positive fusion with other methods (Li & Gao, 2016; Tayfur *et al.*, 2021). To prevent duplicated searches, spatial tabu memories restrict visiting a spatial domain in the vicinity of previous candidates and violating user-set regulations. The TS memorizing mechanism is rather a strategic scheme than an explicit parametric algorithm. In this study, we employed the tabu area list (TAL) that takes an optimal value ($\boldsymbol{x}'^t$) at t-th iteration depending on the TAL memory length $\xi \in \mathbb{N}_1$. In each trial vector expressed as $\boldsymbol{x}^t$, $t \in \mathbb{N}_1$, a tabu list with a size of ξ was updated, and a new candidate

Algorithm. DECS-TAPS metaheuristic

Set initial hyperparameter values ($\boldsymbol{d}$, l_{tb}, α, λ_{r_2}, n_p, c, r_{TR})
Set stagnant factor (SF) and tabu area list (TAL)
$\mu_F = 0.5,\ \mu_{CR} = 0.9$
Initialize random solution $\boldsymbol{x}$ and evaluate fitness value $E = f(\boldsymbol{x})$
Set initial global optimum $\boldsymbol{x}_g = \boldsymbol{x}$
Set iteration count $t = 1$, stagnation count (sc) = 0
while $t < t_{\max}$ and $E <$ fitness threshold and sc < SF **do**
  Update TAL
  Search neighborhood based on adaptive mesh bases and *Lévy* flight
  for $q = 1$ to $2j+1$ ($j = 1, \ldots, n$)
    $\boldsymbol{x}_q = \boldsymbol{x} + \beta\lambda_{r_1}\Delta^t\boldsymbol{d}_k$
    if $q = 2j+1$ then $\boldsymbol{x}_q = \boldsymbol{x} + \alpha \oplus \left(\frac{u}{|v|^{\lambda-1}}\right)\boldsymbol{d}_{k+1}$
    if $\boldsymbol{x}_q$ is in TAL then $\boldsymbol{d} = \lambda_{r_2}(\boldsymbol{x}^t - \bar{\boldsymbol{\tau}})$, $\bar{\boldsymbol{\tau}} = (\sum_{j=1}^{l_{tb}} \boldsymbol{\tau}_j)/l_{tb}$ ($j = 1, \ldots, l_{tb}$)
  Update r_{TR} and TAL (see Section 2.1)
  Update meshsize (see Section 2.2)
  Local minimum $\boldsymbol{x}_\theta = argmin\{f(\boldsymbol{x}_q)\}$ *and* $E_{new} = f(\boldsymbol{x}_\theta)$
  if $E_{new} < E$ then $\boldsymbol{x}_g = \boldsymbol{x}_\theta$, $E = E_{new}$, and $sc = 0$
  else $sc = sc + 1$
  $t = t + 1$
$t_{DE} = 0$, $\boldsymbol{X}$ = tabu list[:n_{p}]
while $t_{DE} < t_{\max}$ and $E <$ fitness threshold **do**
  $F \sim \mathcal{C}(\mu_F, 0.1)$, $CR \sim \mathcal{N}(\mu_{CR}, 0.1)$
  Randomly choose $\boldsymbol{x}_{r_1}, \boldsymbol{x}_{r_2}, \boldsymbol{x}_{r_3}$ from $\boldsymbol{X}$ ($\boldsymbol{x}_{r_1} \neq \boldsymbol{x}_{r_2} \neq \boldsymbol{x}_{r_3}$)
  for $i = 1$ to n_{p}
    $\boldsymbol{u}_i = \boldsymbol{x}_g + F(\boldsymbol{x}_i - \boldsymbol{x}_{r_1}) + F(\boldsymbol{x}_{r_2} - \boldsymbol{x}_{r_3})$
    if $r \sim U(0, 1) \leq CR$ then $\boldsymbol{x}_i = \boldsymbol{u}_i$
  Local minimum $\boldsymbol{x}_\theta = argmin\{f(\boldsymbol{x}_i)\}$ *and* $E_{new} = f(\boldsymbol{x}_\theta)$
  if $E_{new} < E$ then $\boldsymbol{x}_g = \boldsymbol{x}_\theta$, $E = E_{new}$
  Update μ_F, μ_{CR} (eq. (7))
  $t_{DE} = t_{DE} + 1$

Figure 7: Pseudo-code of the DECS-TAPS algorithm.

referring to TAL: $= \{\boldsymbol{\tau}_j\}_{j=1}^{\xi}$ was then generated to avoid the areas around evaluated vectors (termed tabu regions) (TRs) (Glover & Marti, 2006). The TRs were drawn from $\boldsymbol{\tau}_j$ with a radius r_{TR}. Given an updated TAL and a new trial point $\boldsymbol{x}^{t+1}$, it was overridden to have new search directions outside the TRs (Fig. 3) by λ_{r2} $(\boldsymbol{x}^{t+1} - \bar{\boldsymbol{\tau}})$, where $\bar{\boldsymbol{\tau}} = (\sum_{j=1}^{\xi} \boldsymbol{\tau}_j)/\xi$, and $\lambda_{r2} \in \mathbb{R}$ is a random number greater than $\max\{||\boldsymbol{x}^t - \boldsymbol{\tau}_j||\}_{j=1,\ldots,\xi} + r_{\mathrm{TR}}$.

2.2. Mesh adaptive pattern search

DS, also known as pattern search (PS), such as the Hooke–Jeeves algorithm (HJA) (Moser, 2009) is among the earliest optimization techniques. It is built on the non-stochastic, grid-based point-by-point navigation. Depending on the search space coordination and the definition of neighbor bases, it has some variants, such as depth-limited search and the Nelder–Mead/Amoeba method. DS/PS supports solving both non-linearly discontinuous and non-differentiable problems but often exhibits poor performance because it can get stuck in suboptimal regions by finding only immediate local neighbors. Advanced DS variants, such as MADS, are used to overcome this drawback by responsibly changing search areas (Audet & Dennis, 2006).

At each iteration of MADS, the PS strategy subdivides Ω into a k-dimensional set of mesh points. The mesh size parameter, $\boldsymbol{\Delta}^t \in \mathbb{R}^n$, determines the resolution of a PS grid. Based on the number of trials, each candidate creates a neighborhood over a mesh grid. For instance, if $\boldsymbol{x}^t$ is chosen as an incumbent solution at t-th iteration, PS generates several mesh vertices around $\boldsymbol{x}^t$ (Fig. 4), such that

$$x_q^t = x^t + \beta\Delta^t d_k (q = 1, \ldots, 2j), \tag{3}$$

where $\boldsymbol{d}_k$ is a spanning coordinate vector with $d_{kj} = 1$, if $j = k$ and $d_{kj} = 0$; otherwise, if $j \neq k$ and $\beta = 1$ ($q \leq j$) or -1 ($q > j$). Then, $f_\Omega(\boldsymbol{x}_i^t)$ is obtained per feasible point. A combination of MADS and TS suggested by Hedar and Fukushima (2006) contributes to improving the algorithm performance. The dynamic spanning of mesh vertices with flexible mesh size adaptation diversifies the neighbor-

Table 4: Test results (30 runs): standard non-hybrid algorithms.

ID	HJA (PS)	MADS	SA	GA	CE
T1	13 296 ± 9375.4 (10%, 81.080)	1516 ± 9784 (98%, 0.262)	14 752 ± 6045 (64%, 0.099)	11 120 ± 7396 (68%,0.097)	870 ± 146 (100%)
T2	18 796 ± 4740 (6%, 0.363)	14 009 ± 9076 (30%, 0.008)	19 817 ± 1468 (10%, 0.003)	19 547 ± 2481 (6%, 0.003)	16 562 ± 7356 (18%, 0.003)
T3	12 627 ± 9568 (22%, 0.021)	5710 ± 8881 (86%, 0.021)	16 057 ± 5525 (46%, 0.017)	17 674 ± 4738 (22%, 0.013)	1021 ± 221 (100%)
T4	12 598 ± 9541 (22%, 0.424)	18 762 ± 4705 (6%, 0.343)	16 335 ± 4977 (58%, <1E-4)	16 006 ± 5748 (44%, 0.001)	1597 ± 4652 (94%, 0.036)
T5	19 998 ± 7.98 (2%, >1E + 5)	18 027 ± 5919 (10%, >1E + 4)	19 976 ± 154 (4%, 469.4)	20 000 (0%, 955.1)	20 000 (0%, 759.9)
ID	**CMA-ES**	**PSO**	**CS**	**ACO**	**DE**
T1	485 ± 97 (100%)	17 119 ± 4772 (94%,<1E-5)	5366 ± 2725 (98%, 0.133)	3508 ± 632 (100%)	1525 ± 125 (100%)
T2	491 ± 103 (100%)	7679 ± 6148 (86%,<1E-6)	13 414 ± 7910 (42%, 0.002)	13 968 ± 6602 (64%, 0.001)	2471 ± 468 (100%)
T3	524 ± 75 (100%)	1224 ± 908 (100%)	5204 ± 2741 (98%, <1E-4)	3938 ± 3483 (96%, <1E-3)	1017 ± 112 (100%)
T4	602 ± 57 (100%)	3168 ± 4377 (98%,<1E-5)	3223 ± 4348 (94%, 0.032)	1310 ± 378 (100%)	616 ± 293 (100%)
T5	≥20 000 (0%, >1E + 5)	18 448 ± 5289 (10%, >1E + 4)	13 200 ± 7915 (48%, 3191.9)	9816 ± 5986 (82%, 2038.6)	2259 ± 280 (100%)

Note: Number of evaluation ± standard deviation (success rate percentage, mean error).

hood search and makes it resistant to changes in domain scales. Note that the poll size parameter Δ_p is introduced to update Δ such that $\Delta^{t(p)} = k\sqrt{\Delta^t}$. $\Delta^{t(p)}$ represents the stepwise exploration boundary around an incumbent vector. Equation (3) can then be rewritten as

$$\mathbf{x}_q^t = \mathbf{x}^t + \beta\lambda_{r1}\Delta^t \mathrm{d}_k, \tag{4}$$

where λ_{r1} is a random number from $\{\lambda_{r1} \mid 1 \le \lambda_{r1} \le \Delta^{t(p)}/\Delta^t,\ \lambda_{r1} \in \mathbb{N}\}$. If a solution is improved, $\Delta^{t+1} = 2\Delta^t$; otherwise, $\Delta^{(t+1)} = \Delta^t/2$. Note that Ω is normalized as $\{x \mid x \in [0, \text{(Loftness } et\ al., 2005)], x \in \mathbb{R}^n\}$, and every neighbor does not exceed the current exploration boundary.

2.3. Adaptive differential evolution

During the adaptive DE execution, a trial vector $\boldsymbol{u}_i^t$ is obtained by

$$\boldsymbol{u}_i^t = \mathbf{x}_g^t + F\left(\mathbf{x}_i^t - \mathbf{x}_{r_1}^t\right) + F\left(\mathbf{x}_{r_2}^t - \mathbf{x}_{r_3}^t\right), \tag{5}$$

where $\mathbf{x}_g^t$ and F represent the t-th best global solution and the scale factor, respectively. $\mathbf{x}_i^t$ is the current solution, and three random individuals ($\mathbf{x}_{r_1}^t$–$\mathbf{x}_{r_3}^t$) are independently chosen from the current population $\boldsymbol{X}$. Note that we considered the DE mutation operator managed with a single best target and two differential vectors containing a current solution (revised DE/best/2). It is understood that DE is crucially sensitive to mutation (Qin *et al.*, 2009). Among several vector generation methods, the classic DE/rand/1 is robust but inefficient. A greedier scheme, such as DE/best/1, increases the convergence rate, but its diversification is insufficient to avoid premature convergence in high-dimensional multimodal problems (Opara & Arabas, 2018). The modified DE/best/2 strikes a compromise between speed and solution quality in the proposed framework. $\boldsymbol{u}_i^t$ is accepted if a random number [0, 1] is less than or equal to the crossover rate (CR). F and CR are generally constant, but many studies suggest advantages of self-adaptive F and CR. Employing Zhang *et al.*'s proposal (Zhang & Sanderson, 2009) about JADE, DECS-TAPS determines F and CR probabilistically at each iteration such that

$$F^t \sim \mathcal{C}\left(\mu_F^t,\ 0.1\right)$$
$$\mathrm{CR}^t \sim \mathcal{N}\left(\mu_{\mathrm{CR}}^t,\ 0.1\right), \tag{6}$$

where $\mathcal{C}$ and $\mathcal{N}$ denote the Cauchy and normal distribution, respectively, with the mean of μ. Initially, $\mu_F^0 = 0.5$ and $\mu_{\mathrm{CR}}^0 = 0.9$. These values are updated at the end of each generation as follows:

$$\mu_F^t = (1-c)\ \mu_F^t + c\overline{S_F}$$
$$\mu_{\mathrm{CR}}^t = (1-c)\ \mu_{\mathrm{CR}}^t + c\overline{S_{\mathrm{CR}}}, \tag{7}$$

where c is a positive constant between 0 and 1 that prevents trivial evolution and was set to 0.5 in this study. $\overline{S_F}$ and $\overline{S_{\mathrm{CR}}}$ are the means of the parameter archive—s set of all successful F and CR values.

2.4. Cuckoo search

CS is similar to PSO in terms of the solution update, which is expressed as

$$x_i^{t+1} = x_i^t + \alpha \oplus \mathrm{L\acute{e}vy}(\lambda), \tag{8}$$

where $\mathbf{x}_i^{t+1}$ and $\boldsymbol{v}_i^{t+1}$ represent the updated (next) position and velocity at time $t + 1$, respectively. $\alpha > 0$ is the step size (or learning rate) that relates the Lévy flight to the search domain scale, and the product $\oplus$ denotes the entrywise multiplication. Usually, $\alpha = 1$ or $0.3 \le \alpha \le 1.99$ and $\mathrm{L\acute{e}vy} \sim u = t^{-\lambda}$ $(1 \le \lambda \le 3)$ (Mantegna, 1994; Yang & Suash, 2009). In the standard CS, the Lévy step length is characterized by the power-law behavior of stochastic distribution and can be computed as

$$\mathrm{L\acute{e}vy}\ (\lambda) = u/|v|^{\lambda^{-1}}, \tag{9}$$

where $u \sim \mathcal{N}(0, \sigma_u^2)$, $v \sim \mathcal{N}(0, \sigma_v^2)$, $\sigma_u = \left[\frac{\Gamma(1+\lambda)\sin(\frac{\pi\lambda}{2})}{\lambda\Gamma(\frac{(1+\lambda)}{2})2^{0.5(\lambda-1)}}\right]^{\lambda^{-1}}$, and $\sigma_v = 1$. $\mathcal{N}$ and Γ denote the normal and gamma distribution, respectively.

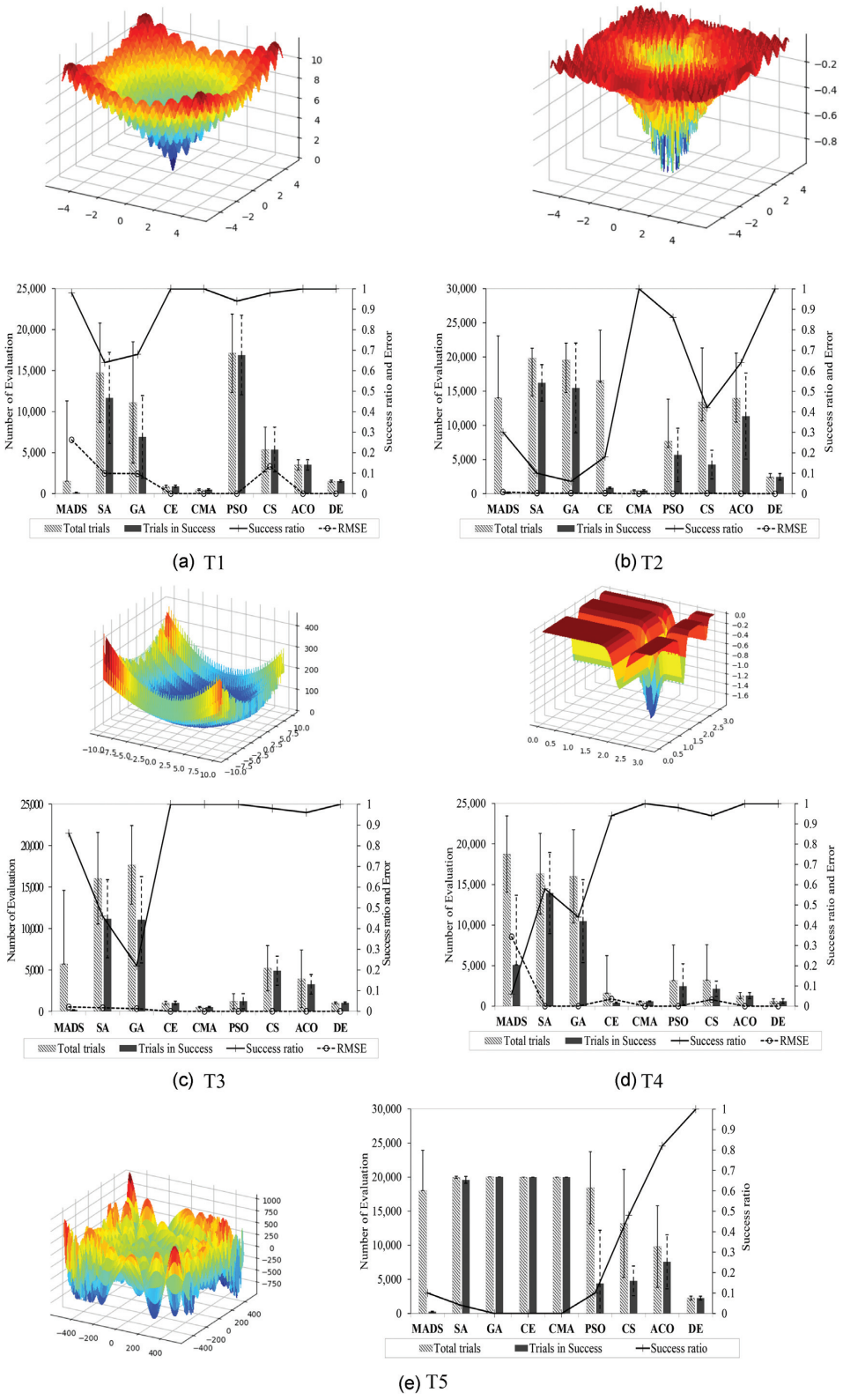
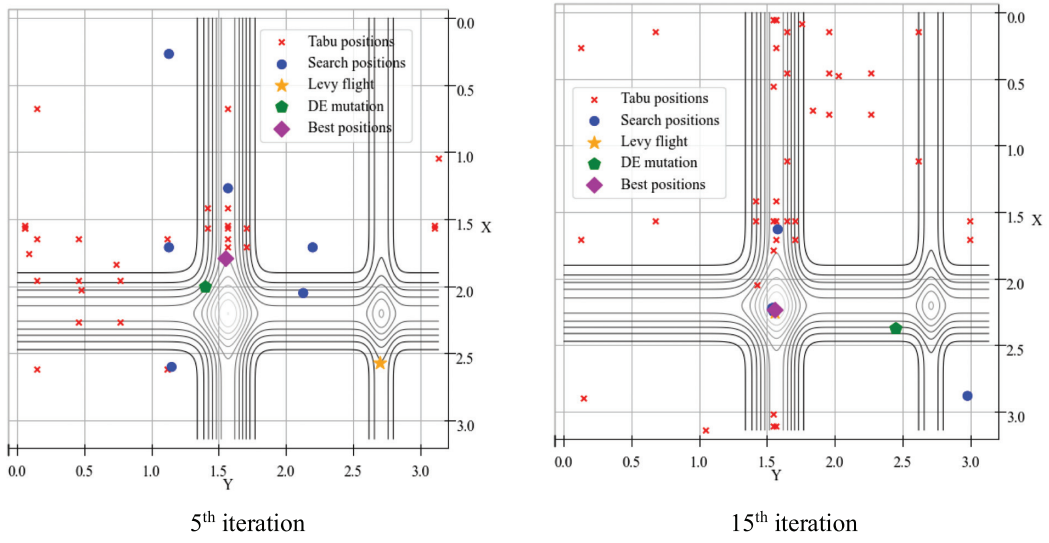
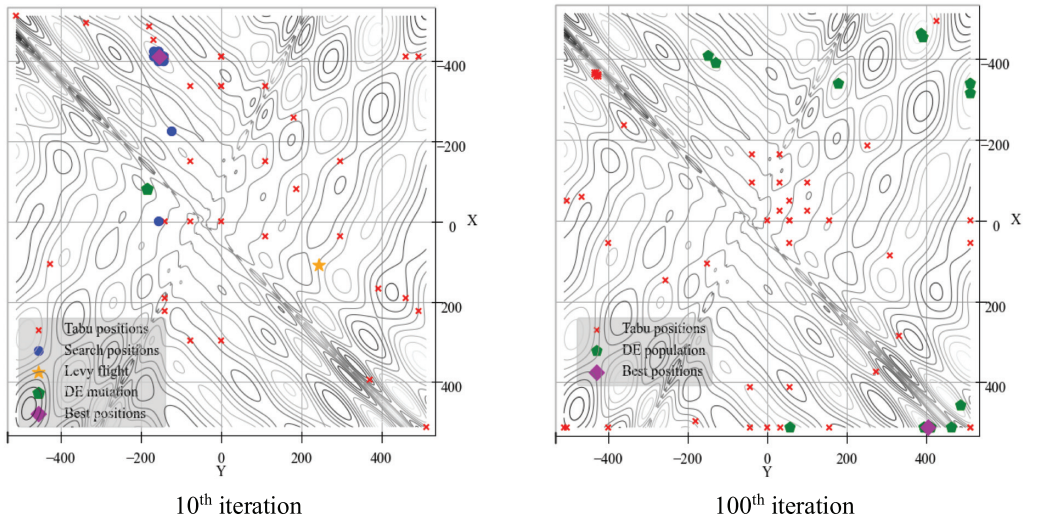


Figure 8: Benchmark functions and performance comparison of basic algorithms (T1–T4).



(a) T4 (Michalewicz, f_4): Separable multimodal function



(b) T5 (Eggholder, f_5): Inseperable multimodal and multifunnel function

Figure 9: DECS-TAPS: 2D search trajectories.

2.5. Test procedure and validation methods

As listed in Table 1, 14 problems were tested based on 12 complex functions selected from COCO—a black-box optimization platform (Hansen *et al.*, 2021) and the literature (Price *et al.*, 2005; Jamil & Yang, 2013). These functions included at least one in five different function categories of COCO. T1–5 were prepared to compare the performances of the 11 classic algorithms in 2D space, and T10–14 were employed to identify the dimensional impacts at an intermediate (n_d = 5–10) or relatively large scales ($n_d \geq 30$). Unimodal functions characterize how algorithms respond to changes in space dimensions. From Szynkiewicz's findings (Szynkiewicz, 2018), Discus, Bent Cigar, Griewank, and the high-multimodal Lunacek bi-Rastrigin functions were considered to compare DECS-TAPS with the CMA-ES hybrids. To indicate the convergence speed and accuracy, the average number of fitness evaluations and its deviation were measured by running 30 simulations per test. 31 existing hybrid algorithms were also tested to demonstrate DECS-TAPS performance. Table 2 shows the pairwise combinations of the major MHOAs, including the PS and TS, used in this study and their hybridization types (A, B, and C, shown in Fig. 2). Table 3 presents the parameter values of the standard algorithms investigated in this study.

Note that every algorithm presented in this work was coded in IronPython 2.7 by the authors, except for CMA-ES, for which we decided to use Hansen's Python open-source code of the bare CMA-ES (Source Code for Module barecmaes2, 2014). To investigate different algorithms on the same basis, each was implemented through a visual scripting component of the Rhino Grasshopper (GH, version 1.0.0007, Robert McNeel & Associates, USA)—an application programming interface in architecture. All the experiments were run on a PC of a 64-bit windows system with Intel(R) Core(TM) i7-7700 CPU (3.60GHz) and 32 GB RAM.

Table 5: Performance test results of hybrid algorithms (T6, $n_d = 5$).

Hybrids	Min	Max	μ	σ	RMSE(x^*)	RMSE(f)	γ-value
DECS-TAPS	**14**	**222**	**52**	**37.683**	**0**	**0**	**0**
TS-PS	132	2719	810	720.570	0	0	0
TS-CS	23 115	41 541	40 501	3837.122	0.239	1.857	3.76
TS-GA	20 040	20 040	20 040	0	0.575	12.398	12.42
TS-SA	9169	9169	9169	0	0.622	13.148	6.03
PS-DE	1581	20 445	5131	6865.351	0.117	2.935	0.75
PS-CS	12 329	48 041	40 572	11 916.958	0.023	0.179	0.36
PS-FA	26 962	27 027	27 012	17.143	0.812	14.981	20.23
PS-GA	26 449	26 540	26 505	24.240	0.468	8.638	11.45
PS-SA	32 089	32 089	32 089	0	1.222	35.779	57.41
DE-CMA-ES	256	256	256	0	0	0	0
PSO-DE	7681	34 081	29 446	9375.166	0.605	0.246	0.36
DE-CS	3212	61 040	7832	14 224.967	0.028	0.181	0.07
FA-DE	30 040	30 040	30 040	0	0.059	0.193	0.29
GA-DE	1120	60 040	40 648	27 425.389	0.018	0.016	0.03
SA-DE	2200	3080	2629	194.301	0.001	0	0
PSO-CMA-ES	8	504	487	89.035	0.036	0.146	0
PSO-ACO	8090	19 038	13 081	2294.274	0.002	0	0
PSO-CS	20 541	20 541	20 541	0	0.753	12.955	13.31
PSO-CE	19 953	21 034	20 959	256.761	0.547	0.068	0.07
PSO-GA	19 900	25 420	23 194	1422.169	0	0	0
PSO-SA	8524	9855	9703	223.494	0.277	0.091	0.04
ACO-CS	4590	91 040	74 035	34 014.389	0.585	6.760	25.02
ACO-GA	7320	11 120	9158	913.606	0.001	0	0
FA-CS	20 040	20 040	20 040	0	1.059	22.767	22.81
GA-CS	20 500	20 500	20 500	0.000	0.162	1.165	1.19
CS-SA	16 112	17 998	17 933	344.130	0.338	2.742	2.46
FA-CE	20 501	20 501	20 501	0	0.048	0.315	0.32
FA-GA	22 040	22 040	22 040	0	0.975	24.432	26.92
FA-SA	20 916	20 916	20 916	0	0.888	22.443	23.47
GA-CE	20 520	20 520	20 520	0	0.283	4.606	4.73
GA-SA	35 080	35 080	35 080	0	0.529	12.201	21.40

Note: Min/Max – minimum/maximum count of fitness evaluation, μ – mean of fitness evaluation count, and σ – standard deviation of fitness evaluation count.

Table 6: Performance test results of hybrid algorithms (T7, $n_d = 5$).

Hybrids	Min	Max	μ	σ	RMSE(x^*)	RMSE(f)	γ-value
DECS-TAPS	**656**	**9398**	**1693**	**2450.169**	**0.268**	**24.680**	**2.09**
TS-PS	119	4006	785	858.195	0.006	0	0
TS-CS	10 997	41 541	36 497	10 180.928	2.287	352.836	643.88
TS-GA	20 040	20 040	20 040	0	0.872	359.162	359.88
TS-SA	9169	9169	9169	0	1.162	425.450	195.05
PS-DE	1533	20 541	17 498	6412.820	2.295	387.698	339.20
PS-CS	7145	48 041	42 348	13 097.960	2.629	466.013	986.74
PS-FA	26 949	27 027	27 008	18.287	1.925	564.096	761.77
PS-GA	26 462	26 540	26 513	18.185	0.485	126.943	168.28
PS-SA	32 089	32 089	32 089	0	3.551	1450.958	2327.99
DE-CMA-ES	144	256	238	27.476	0.799	79.856	0.95
PSO-DE	34 081	34 081	34 081	0	2.825	472.757	805.60
DE-CS	2846	61 040	43 712	26 469.567	1.910	219.624	480.01
FA-DE	30 040	30 040	30 040	0	2.890	35 775.029	53 734.09
GA-DE	60 040	60 040	60 040	0	3.058	4157.467	12 480.72
SA-DE	2040	35 960	16 168	15 891.842	0.854	106.024	85.71
PSO-CMA-ES	504	504	504	0	3.086	1091.090	27.50
PSO-ACO	6319	80 540	35 659	34 185.819	1.133	206.367	367.94
PSO-CS	20 541	20 541	20 541	0	2.604	1335.709	1371.84
PSO-CE	17 609	21 033	20 816	781.278	2.895	542.655	564.81
PSO-GA	19 420	30 040	27 432	3552.005	1.673	166.384	228.21
PSO-SA	19 910	20 185	20 040	55.879	2.994	683.086	684.45
ACO-CS	2588	91 040	82 237	26 408.632	2.980	725.703	2983.99
ACO-GA	4800	50 240	16 699	16 864.613	0.547	59.896	50.01
FA-CS	20 040	20 040	20 040	0	3.180	3652.501	3659.81
GA-CS	3506	3506	3506	0	1.788	481.729	97.74
CS-SA	16 440	17 998	17 946	279.670	2.721	454.469	407.80
FA-CE	20 501	20 501	20 501	0	2.054	7606.007	7796.54
FA-GA	22 040	22 040	22 040	0	2.454	1860.052	2049.78
FA-SA	20 916	20 916	20 916	0	2.098	1441.033	1507.03
GA-CE	20 520	20 520	20 520	0	1.256	569.496	584.30
GA-SA	35 080	35 080	35 080	0	1.314	485.022	850.73

Table 7: Performance test results of hybrid algorithms (T8, $n_d = 5$).

Hybrids	Min	Max	μ	σ	RMSE(x^*)	RMSE(f)	γ-value
DECS-TAPS	**14**	**82**	**36**	**16.466**	**0**	**0**	**0**
TS-PS	834	6502	3264	1552.946	0.004	2.837E + 09	4.63E + 08
TS-CS	41 541	41 541	41 541	0	15.292	3.191E + 09	6.63E + 09
TS-GA	20 040	20 040	20 040	0	20.522	2.852E + 18	2.86E + 18
TS-SA	18 377	18 377	18 377	0	23.801	4.485E + 18	4.12E + 18
PS-DE	2853	20 313	19 154	4335.383	15.002	1.865E + 12	1.79E + 12
PS-CS	48 041	48 041	48 041	0	44.721	1.000E + 08	2.40E + 08
PS-FA	26 559	27 027	26 875	121.680	45.141	5.705E + 20	7.67E + 20
PS-GA	26 423	26 527	26 477	25.360	26.563	3.648E + 18	4.83E + 18
PS-SA	32 089	32 089	32 089	0	61.540	2.109E + 22	3.38E + 22
DE-CMA-ES	256	256	256	0	9.854	4.671E + 06	5.98E + 04
PSO-DE	34 081	34 081	34 081	0	19.950	4.743E + 09	8.08E + 09
DE-CS	8214	12 240	10 337	1152.927	0.011	0	0
FA-DE	30 040	30 040	30 040	0.000	21.167	2.467E + 09	3.70E + 09
GA-DE	3280	60 040	37 908	27 113.179	0.319	7.333E + 13	1.39E + 14
SA-DE	4760	35 960	30 792	11 556.121	9.549	3.291E + 06	5.07E + 06
PSO-CMA-ES	504	504	504	0	0.116	2.233E + 11	5.63E + 09
PSO-ACO	24 351	45 442	33 287	4665.621	0.009	0	0
PSO-CS	20 541	20 541	20 541	0	24.945	9.401E + 19	9.66E + 19
PSO-CE	20 977	21 020	21 006	10.873	20.414	5.310E + 08	5.58E + 08
PSO-GA	27 220	30 040	29 822	671.086	18.290	1.339E + 07	2.00E + 07
PSO-SA	20 385	20 635	20 539	65.410	25.913	3.109E + 08	3.19E + 08
ACO-CS	15 874	91 040	63 546	33 784.676	9.418	3.711E + 19	1.18E + 20
ACO-GA	18 320	27 800	22 365	2504.176	0.010	0	0
FA-CS	20 040	20 040	20 040	0	27.620	1.526E + 20	1.53E + 20
GA-CS	3506	3506	3506	0	14.979	3.517E + 17	4.47E + 14
CS-SA	17 998	17 998	17 998	0	17.666	8.805E + 11	7.92E + 11
FA-CE	20 501	20 501	20 501	0	9.363	7.572E + 17	7.76E + 17
FA-GA	22 040	22 040	22 040	0	29.243	1.493E + 20	1.65E + 20
FA-SA	20 916	20 916	20 916	0	22.877	2.918E + 19	3.05E + 19
GA-CE	20 520	20 520	20 520	0	23.442	5.956E + 18	6.11E + 18
GA-SA	35 080	35 080	35 080	0	23.078	2.120E + 18	3.72E + 18

3. Differential Evolutionary Cuckoo-Search-Integrated Tabu-Adaptive Pattern Search (DECS-TAPS)

A central idea of DECS-TAPS is to incorporate TS and the random walking mechanism (Yang & Suash, 2009) of CS into adaptive PS and DE by mixing hybridization schemes A and B so that diversification can be accelerated in early-phase exploration and adaptive DE prevents incomplete convergence. Although the mesh-based discrete move of candidates in TAPS enhances random search, it increases the risk of staying in local optima, whereby the converging robustness depends on the choice of the initial random solution. DE can improve solution quality, but the population-based mechanism consumes computation resources.

Accordingly, DECS-TAPS divides an optimization scheme into two sequential phases: (i) CS-integrated TAPS and (ii) adaptive DE. In the first phase, TAPS is performed to find successful candidates as quickly as possible over the entire domain. If it remains stagnant at a local optimum for some time, DE starts from the best local solution (Fig. 5). This strategy allows for DE-based exploitation with far more reduced computation and increased robustness against multimodal/multifunnel functions. Depending on the CS ratio $\gamma_{cs} \in (0, 1)$, we implemented the Lévy flight mechanism to form random walk mesh bases of TAPS to accelerate exploration. Since the four different algorithms (TS, PS, CS, and DE) are cooperatively synthesized, proper default parameter setting is crucial. From our preliminary testing, the TS memory (l_{tb}) was set to 25, and the phase switching threshold (the stagnation factor) was 50. The population size (n_p) was 20 ($0.5n_p$ in other MHOAs), and the step length was computed from equation (9) with λ of 1.5 and $\alpha = 1$, by following Mantegna's algorithm (Mantegna, 1994). Figures 5–7 illustrate the search mechanism, a pseudo-code, and a flow chart.

4. Results and Discussion

4.1. Benchmark function tests

4.1.1. *Comparison of standard algorithms*

Prior to testing the hybrid algorithms, we performed white-box optimization (T1–5) with various low-dimensional test functions (f_1–f_5) to briefly characterize the performance of the classic algorithms. The egg-holder function (f_5) was employed to identify the DECS-TAPS search mechanism in a very irregularly structured multimodal function. The maximum number of iterations per test run was set to 500 so that the maximum number of fitness evaluations did not exceed 20 000. The initial mesh size in MADS was 2.5 in T5 and 1 in the other tests. Table 4 and Fig. 8 show the test results with graphical representations of the test functions. HJA's poor accuracy (< 22%) indicates that it was not suitable even for well-structured functions (T1 and T2), and we discounted the performance comparison in Fig. 8a–e. It was also found that, in all the tests, the standard SA and GA slowly converged with weak robustness (high deviation and low accuracy). The PBAs (PSO, CS, ACO, and DE) were moderately performed than the non-PBAs. Upon comparing T1 and T3 with T2, the results suggested that the structural topology had a more significant impact than modality. We

Table 8: Performance test results of hybrid algorithms (T9, $n_d = 5$).

Hybrids	Min	Max	μ	σ	RMSE(x*)	RMSE(f)	γ-value
DECS-TAPS	**14**	**208**	**44**	**38.577**	**0**	**0**	**0**
TS-PS	535	6502	5480	1827.573	4.356	0.098	0.03
TS-CS	41 541	41 541	41 541	0	18.300	0.337	0.70
TS-GA	20 040	20 040	20 040	0	59.001	36.352	36.43
TS-SA	18 377	18 377	18 377	0	58.888	32.325	29.70
PS-DE	11 493	20 313	20 019	1583.238	5.922	0.003	0
PS-CS	48 041	48 041	48 041	0	16.767	0.221	0.53
PS-FA	26 195	27 014	26 728	183.893	181.835	2138.948	2858.44
PS-GA	26 410	26 540	26 479	28.511	65.391	46.112	61.05
PS-SA	32 089	32 089	32 089	0.000	377.765	37 043.205	59 433.97
DE-CMA-ES	72	256	234	51.491	3.331	0	0
PSO-DE	34 081	34 081	34 081	0	59.956	78.743	134.18
DE-CS	61 040	61 040	61 040	0	9.131	0.013	0.04
FA-DE	30 040	30 040	30 040	0	6.142	0.005	0.01
GA-DE	60 040	60 040	60 040	0	4.790	0.081	0.24
SA-DE	16 520	35 960	34 104	4740.718	2.783	0	0
PSO-CMA-ES	24	504	359	196.903	0.891	0.028	0
PSO-ACO	80 540	80 540	80 540	0	8.364	1.661	6.69
PSO-CS	20 541	20 541	20 541	0	75.950	289.060	296.88
PSO-CE	20 969	21 035	21 009	14.182	66.564	200.676	210.80
PSO-GA	30 040	30 040	30 040	0	41.955	13.919	20.91
PSO-SA	19 035	19 535	19 391	126.012	99.364	1300.104	1260.48
ACO-CS	91 040	91 040	91 040	0	90.402	242.577	1104.21
ACO-GA	50 120	50 240	50 128	24.000	5.113	0.017	0.04
FA-CS	20 040	20 040	20 040	0	50.096	215.333	215.76
GA-CS	3506	3506	3506	0	29.113	4.718	0.65
CS-SA	36 858	36 858	36 858	0	18.040	0.295	0.54
FA-CE	20 501	20 501	20 501	0	32.082	5.755	5.90
FA-GA	22 040	22 040	22 040	0	113.467	381.153	420.03
FA-SA	20 916	20 916	20 916	0	104.867	276.437	289.10
GA-CE	20 520	20 520	20 520	0	66.866	59.422	60.97
GA-SA	35 080	35 080	35 080	0	63.153	42.942	75.32

found that irregularity in objective distribution (f_2) led to a significant degradation of CS performance. PSO performed well in the asymmetrically shaped function (f_4, T4), but the PSO evolution exposed a risk for regularly clustered distribution with a relatively high-conditioned global optimum (T1). Meanwhile, we noticed that stochastic algorithms, such as CE and CMA-ES, were remarkably successful in all the tests except T5. However, CE was degraded by the crude morphology of a function. As addressed in Li *et al.* (2020), the standard CMA-ES was superior to others. Nevertheless, in Fig. 8e, we found that it was prone to the morphological complexity of an objective, like Benhamou *et al.*'s findings (Benhamou *et al.*, 2019). The performance of DE, although it offers simplicity in computational implementation, bears comparison with CMA-ES. This reveals that DE was highly competitive in terms of achieving accuracy for all tests and had a far lower number of evaluations in the multifunnel function (T5).

4.1.2. DECS-TAPS search behavior

In 4.1.1, the T1–T5 results suggest that stochastic or local-search algorithms are less robust (significant differences between total and successful trials). Nevertheless, PS is advantageous in reducing the converging time by limiting random movement in a mesh although it may degrade search resolution. Most basic algorithms suffer from premature convergence in large dimensional multimodal inseparable problems. By running 30 times each for T4 and T5, DECS-TAPS located a global optimum in much fewer iterations (no greater than 20 and 150 iterations). Figure 9a illustrates the searching trajectories of DECS-TAPS in the 2D space. Tracking the changes in the early and later phases of the iterations indicates that the combinatorial search of Lévy flight and TS-PS was effective in rapid exploration, and the extensive tabu list, TAL, prevented falling in local minima.

Figure 9b also shows that the DE particles evolved to exploit the concentrated areas and that diversity was maintained. CS works for diversification directed by random walks, preventing the pitfall of choosing the local minima. Rather than spreading random solutions for diversity, TRs guide exploration more efficiently.

4.1.3. Comparison of hybrid algorithms with DECS-TAPS

Tables 5–13 show the statistical results of T6–T14. For each algorithm, the iteration number was constrained to $n_p \times 500$ and 20 000. The accuracy of the final solution (x*) and fitness (f) was indicated by their root mean square error (RMSE) values after running 30 simulations. The γ-value was introduced to represent an adjusted fitness error augmented by the number of iterations such that

$$\gamma\text{-value} = \text{RMSE}\,(f) \times \mu/20\,000. \tag{10}$$

If no error occurs, the γ-value becomes zero. Otherwise, a greater γ-value of an algorithm indicates that it is relatively inefficient. Overall, in all the results, DECS-TAPS far outperformed other major pairwise hybrid algorithms. DECS-TAPS significantly reduced μ to about 98–99% on average compared to the popular PBAs. As PSO and ACO were superior to other traditional algorithms in the prior tests (Fig. 8), PSO-ACO was noticeably accurate except DE and CMA-ES. However, it should be noted that CS, FA,

Table 9: Performance test results of hybrid algorithms (T10, $n_d = 10$).

Hybrids	Min	Max	μ	σ	RMSE(x^*)	RMSE(f)	γ-value
DECS-TAPS	**24**	**238**	**73**	**52.512**	**0**	**0**	**0**
TS-PS	600	11 502	3329	3043.584	0.001	0	0
TS-CS	41 541	41 541	41 541	0	1.356	661.860	1374.72
TS-GA	20 040	20 040	20 040	0	1.853	5605.526	5616.74
TS-SA	18 377	18 377	18 377	0	1.817	4876.791	4481.04
PS-DE	11 001	20 727	18 744	3871.787	1.303	3201.317	3000.34
PS-CS	53 041	53 041	53 041	0	1.562	972.006	2577.81
PS-FA	31 925	32 017	31 992	29.002	1.872	4848.162	7755.05
PS-GA	31 448	31 540	31 502	27.480	1.705	3603.010	5675.11
PS-SA	55 009	55 009	55 009	0	2.695	11 749.122	32 315.37
DE-CMA-ES	250	250	250	0	0.804	232.710	2.91
PSO-DE	34 081	34 081	34 081	0	2.185	2384.905	4064.00
DE-CS	61 040	61 040	61 040	0	1.134	233.488	712.60
FA-DE	30 040	30 040	30 040	0	1.085	1240.261	1862.87
GA-DE	60 040	60 040	60 040	0	0.034	5.340	16.03
SA-DE	35 960	35 960	35 960	0	0.689	42.332	76.11
PSO-CMA-ES	500	500	500	0	0.412	1352.949	33.82
PSO-ACO	66 372	80 540	78 667	4077.219	0.506	252.337	992.53
PSO-CS	20 541	20 541	20 541	0	1.890	5427.375	5574.19
PSO-CE	21 022	21 036	21 031	4.409	2.208	1764.132	1855.04
PSO-GA	30 040	30 040	30 040	0	0.931	104.478	156.93
PSO-SA	20 040	20 200	20 147	44.331	2.167	2034.408	2049.31
ACO-CS	91 040	91 040	91 040	0.000	1.982	2100.222	9560.21
ACO-GA	50 120	50 200	50 140	34.641	0.702	467.669	1172.45
FA-CS	20 040	20 040	20 040	0	1.925	5836.625	5848.30
GA-CS	3506	3506	3506	0	1.519	4341.682	1607.06
CS-SA	36 858	36 858	36 858	0	1.295	560.102	1032.21
FA-CE	20 501	20 501	20 501	0	0.954	2228.502	2284.33
FA-GA	22 040	22 040	22 040	0	2.145	8886.097	9792.48
FA-SA	20 916	20 916	20 916	0	1.908	8054.235	8423.12
GA-CE	20 520	20 520	20 520	0	0.972	2219.913	2277.63
GA-SA	35 080	35 080	35 080	0	1.819	5850.945	10 262.56

and ACO hybrids increased μ by up to 2–4 times compared to the preset number since their operators required a large number of fitness evaluations. In particular, the ant-based production of solutions with the pheromone update and optional daemon actions caused slow convergence in ACO metaheuristics.

Similar to other studies (Hansen *et al.*, 2008; Krause *et al.*, 2016; Szynkiewicz, 2018) investigating the excellent performance of CMA-ES, our experiments also showed that CMA-ES hybrids were very successful. In T6, a PSO-CMA-ES test was completed with the smallest evaluation number, and in large-dimensional tests (T12–T14), DE-CMA-ES and PSO-CMA-ES converged even more quickly than DECS-TAPS (Tables 11–13). Nevertheless, T13 and T14 showed that their performance was significantly undermined by their functional complexity due to large-dimensional and multifunnel distribution with a weak global shape, while DECS-TAPS converged with 100% accuracy in every test.

4.1.4. *Performance analysis*

From the data in Tables 5–13, Fig. 10 shows the changes in the indicator values according to the different characteristics of the target functions (multimodal functions only), including the dimensional growth of search space, formal complexity, and condition number on the hybrids. In Fig. 10a and b, most PBAs reached the maximum iteration number even in low-dimensional problems, and the μ values tended to increase in PS hybrids. This reveals that they are prone to exploration in large space. The sharp increases of μ at $n_d = 30$ and 50 in PS-SA suggest that the SA operators are not well suited for escaping local optima. It is worth noting that GA-CS was occasionally better than PSO-ACO (T7, 8, and 11) or PSO-GA (T9 and 12) in some separable functions (f_6, f_8, f_9, and f_{10}). However, it was not always successful in finding real solutions. This, in effect, parallels the known finding (Forbes & Teli, 2006; Sutton *et al.*, 2006) that PSO is deficient in solving irregularly shaped multifunnel functions even if it works relatively well with multifunnel functions with regular shape structures (e.g., Rastrigin's) at low or intermediate dimensions (Sutton *et al.*, 2006) (Fig. 10c). Specifically, due to the PSO's "flying" mechanism (Salahi *et al.*, 2013), the hybridization of GA or ACO often results in mean-biased stagnation, thereby lacking search diversity.

The MHOA performance was critically affected by neighbor search methods. Ineffective integration of FA, GA, or SA became the worst choice in intricate functions (T6–T8). Despite the increased number of evaluations, PS-FA/-GA fell into local minima in complex and high-dimensional problems, whereas DE or CS showed moderate performance, as seen in DE-CS. However, the satisfactory performance of TS-PS (Fig. 10b) strongly suggests that adaptive random exploration well suited to the topology of a problem significantly improves the solution quality and efficiency.

Figure 10b also clearly shows that DE-/PSO-CMA-ES performed exceedingly well in low and moderate space dimensions, but unlike DECS-TAPS, their γ-values sharply increased in a larger space ($n_d = 50$). As Jin *et al.* (Jin *et al.*, 2020) point out, standard CMA-ES is prone to high functional complexity and premature stagnation in large-scale problems. Also, it was recently found that if the population size is large, CMA-ES often encounters certain termination

Table 10: Performance test results of hybrid algorithms (T11, $n_d = 30$).

Hybrids	Min	Max	μ	σ	RMSE(x^*)	RMSE(f)	γ-value
DECS-TAPS	**64**	**382**	**201**	**83.882**	**0**	**0**	**0**
TS-PS	821	31 502	5984	7340.337	0.001	1.203E-05	0
TS-CS	41 541	41 541	41 541	0	1.266	1.922E + 04	3.99E + 04
TS-GA	20 040	20 040	20 040	0	2.334	1.277E + 05	1.28E + 05
TS-SA	18 377	18 377	18 377	0	2.251	1.238E + 05	1.14E + 05
PS-DE	22 481	31 001	25 992	4090.485	2.014	5.088E + 04	6.61E + 04
PS-CS	73 041	73 041	73 041	0	1.324	8.349E + 03	3.05E + 04
PS-FA	51 851	51 977	51 918	36.130	2.526	1.298E + 05	3.37E + 05
PS-GA	51 162	51 540	51 388	101.122	2.211	1.156E + 05	2.97E + 05
PS-SA	146 689	146 689	146 689	0	2.643	9.911E + 04	7.27E + 05
DE-CMA-ES	252	252	252	0	1.156	1.498E + 04	1.89E + 02
PSO-DE	34 081	34 081	34 081	0	2.301	4.420E + 04	7.53E + 04
DE-CS	61 040	61 040	61 040	0	1.291	2.862E + 03	8.73E + 03
FA-DE	30 040	30 040	30 040	0	1.414	4.600E + 04	6.91E + 04
GA-DE	60 040	60 040	60 040	0	0.041	1.017E + 02	3.05E + 02
SA-DE	35 960	35 960	35 960	0	1.274	9.161E + 03	1.65E + 04
PSO-CMA-ES	504	504	504	0	0.502	4.808E + 04	1.21E + 03
PSO-ACO	80 540	80 540	80 540	0	1.331	3.804E + 04	1.53E + 05
PSO-CS	20 541	20 541	20 541	0	2.047	1.063E + 05	1.09E + 05
PSO-CE	20 976	21 039	21 028	17.796	2.635	6.801E + 04	7.15E + 04
PSO-GA	30 040	30 040	30 040	0	1.551	6.443E + 03	9.68E + 03
PSO-SA	19 900	20 250	20 150	100.585	2.473	7.316E + 04	7.37E + 04
ACO-CS	91 040	91 040	91 040	0	1.851	1.199E + 04	5.46E + 04
ACO-GA	50 120	50 120	50 120	0	1.244	4.415E + 04	1.11E + 05
FA-CS	20 040	20 040	20 040	0	2.201	1.196E + 05	1.20E + 05
GA-CS	3506	3506	3506	0	1.356	6.479E + 04	5.60E + 04
CS-SA	36 858	36 858	36 858	0	1.363	2.179E + 04	4.02E + 04
FA-CE	20 501	20 501	20 501	0	1.070	4.861E + 04	4.98E + 04
FA-GA	22 040	22 040	22 040	0	2.417	1.584E + 05	1.75E + 05
FA-SA	20 916	20 916	20 916	0	2.453	1.535E + 05	1.60E + 05
GA-CE	20 520	20 520	20 520	0	1.131	5.231E + 04	5.37E + 04
GA-SA	35 080	35 080	35 080	0	2.213	1.239E + 05	2.17E + 05

errors in cases when the active step-size update does not guarantee the positive definiteness of the covariance matrix (Akimoto & Hansen, 2020). Meanwhile, it is known that DE is weak at high-conditioned functions. Comparing Fig. 10c and d, we found that the morphology of fitness distribution affected the PSO hybrids, while the DE hybrids were relatively more sensitive to the condition number.

From these findings, we obtain Fig. 11, which graphically illustrates the effectiveness of the pairwise algorithm combinations based on the log-scale γ-values. PS-SA and FA-DE were the worst at $n_d = 5$. Figure 11b indicates that most combinations were desirable except FA-DE. In higher dimensions, pairing the individual algorithms was unsatisfactory in most cases, but TS-PS, GA-DE, DE-CMA-ES, and PSO-CMA-ES were better, while FA/SA/GA-centered algorithms degraded. Convergence depending on stochastic allocation of parameters may become unstably sluggish, particularly when dealing with high-conditioned multimodal objective functions with a large number of variables. In these results, the computation efficiency and robustness of DECS-TAPS, with little regard to changes in space dimension or function complexity, confirm that fine-tuned DS hybridized with well-designed random movement significantly contributes to reducing convergence time.

4.2. Parametric sensitivity analysis

Multiplicative hybrids that blend different phases and assorted operators may increase the number of hyperparameters, which need to be fine-tuned before execution. DECS-TAPS incorporated the hyperparameters of DE, CS, TS, and PS, as listed in Tables 3 and 14. To identify the parametric sensitivity of the operational factors, we performed a global sensitivity analysis of CS, DE, and DECS-TAPS at $n_d = 10$ and 30, employing the variance-based Sobol method and Python SALib 1.4.5. The number of iterations and Latin hypercube sampling (LHS) size were set to 3000 and 512, respectively, and simulations were repeated 30 times per test. In this analysis, the parametric uncertainty was measured by the total-order/-Sobol index (ST), first-order index (S1), and second-order index (S2). ST is a comprehensive indicator of the model output variance caused by parameter input, while S1 and S2 indicate influences from individual factors and interactions of the two factors, respectively. Table 14 and Fig. 12 show that STs in DECS-TAPS became lower than its important ingredients (CS and DE) in all the cases. In the original CS, r_{cs} and λ critically affected its performance, but their impact was notably mitigated in DECS-TAPS (Fig. 13). This suggests that dependence between the algorithm performance and the parameter values was weakened in the multihybrid algorithm, which alleviated hyperparameter tuning more than the standard MHOAs. However, this suggestion may not be confirmed unless an extended comparative investigation on other MHOAs is conducted.

4.3. Application to architectural design optimization and validation

4.3.1. *Multi-objective ADO*

To characterize the performance of the suggested algorithms in actual building design practice, we chose a simple shoebox

Table 11: Performance test results of hybrid algorithms (T12, $n_d = 50$).

Hybrids	Min	Max	μ	σ	RMSE(x^*)	RMSE(f)	γ-value
DECS-TAPS	**104**	**1662**	**380**	**300.227**	**0**	**0**	**0**
TS-PS	26 473	51 502	46 496	7889.540	0.998	3.293E + 04	7.66E + 04
TS-CS	41 541	41 541	41 541	0	46.844	1.407E + 10	2.92E + 10
TS-GA	20 040	20 040	20 040	0	426.409	8.320E + 13	8.34E + 13
TS-SA	18 377	18 377	18 377	0	425.034	8.176E + 13	7.51E + 13
PS-DE	22 653	22 653	22 653	0	167.685	2.122E + 12	2.40E + 12
PS-CS	93 041	93 041	93 041	0	52.013	2.037E + 10	9.48E + 10
PS-FA	64 933	71 628	69 671	2207.503	491.627	1.479E + 14	5.15E + 14
PS-GA	70 613	71 334	70 933	202.796	431.432	8.701E + 13	3.09E + 14
PS-SA	238 369	238 369	238 369	0	573.272	2.780E + 14	3.31E + 15
DE-CMA-ES	255	255	255	0	112.092	4.006E + 11	5.11E + 09
PSO-DE	34 081	34 081	34 081	0	330.531	4.466E + 13	7.61E + 13
DE-CS	61 040	61 040	61 040	0	9.346	3.493E + 07	1.07E + 08
FA-DE	30 040	30 040	30 040	0	24.491	1.684E + 09	2.53E + 09
GA-DE	60 040	60 040	60 040	0	7.527	8.376E + 06	2.51E + 07
SA-DE	35 960	35 960	35 960	0	120.217	6.018E + 11	1.08E + 12
PSO-CMA-ES	510	510	510	0	0.272	27.194	0.69
PSO-ACO	80 540	80 540	80 540	0	23.994	3.283E + 09	1.32E + 10
PSO-CS	20 541	20 541	20 541	0	430.945	9.252E + 13	9.50E + 13
PSO-CE	20 599	20 627	20 615	7.048	329.289	4.672E + 13	4.82E + 13
PSO-GA	30 040	30 040	30 040	0	232.090	9.521E + 12	1.43E + 13
PSO-SA	18 730	19 045	18 867	86.712	237.845	1.449E + 13	1.37E + 13
ACO-CS	91 040	91 040	91 040	0	249.507	1.238E + 13	5.64E + 13
ACO-GA	50 120	50 160	50 124	12.571	21.040	6.255E + 08	1.57E + 09
FA-CS	20 040	20 040	20 040	0	293.686	4.036E + 13	4.04E + 13
GA-CS	3506	3506	3506	0	124.673	6.715E + 11	7.92E + 10
CS-SA	36 858	36 858	36 858	0	57.191	2.813E + 10	5.18E + 10
FA-CE	20 501	20 501	20 501	0	179.799	2.682E + 12	2.75E + 12
FA-GA	22 040	22 040	22 040	0	460.723	1.144E + 14	1.26E + 14
FA-SA	20 916	20 916	20 916	0	449.222	1.025E + 14	1.07E + 14
GA-CE	20 520	20 520	20 520	0	212.247	5.154E + 12	5.29E + 12
GA-SA	35 080	35 080	35 080	0	433.905	8.893E + 13	1.56E + 14

building model (Fig. 14). For comparison, we selected the four best standard algorithms (GA, ACO, DE, and CMA-ES) from Section 4.1.1 and the eight best performative hybrids from Section 4.2, including DECS-TAPS.

The design goal was to find an optimal form of the building space defined by seven geometric variables x_1–x_7 to minimize annual energy usage and maximize indoor daylight up to a desirable level, 500 *lx* in this case. Every parameter was interdependent, x_1, $x_2 \in$[10, 20]m, x_4–$x_7 \in$[0.2, 0.9]. Note that x_3 was defined depending on the floor area to constrain the total space volume (1800 m^3). The annual energy use was represented as the energy use intensity (EUI, kWh/m^2 yr) and the daylight as the annual daylight autonomy (DA, %). DA accounts for the total percentage of annually occupied hours that receive more than the illumination threshold (500 *lx*). Using the min–max normalization method, the ranges of EUI and DA output were scaled in [0, (Loftness *et al.*, 2005)] and [−1, 0] so that the multi-objective fitness values ranged within [0, (Hensen & Lamberts, 2011)].

In the interest of building simulation time, the target functions were modeled using machine learning (ML). The random forest regression (RFR) technique from scikit-learn 1.1 was employed to model the EUI and DA. Two different datasets of 800 training and 200 test LHS samples each were generated by running EnergyPlus and Radiance simulation with default settings from the Climate Studio and the Honeybee GH plugins. The R^2 values of the final RFR models were 0.9698 and 0.9593 for EUI and DA, respectively.

Figure 15 depicts the resulting convergence plots of the test algorithms. The fitness evaluation number of an algorithm represents an average after running 30 repetitions. This shows that most of the hybrids performed better than the non-hybrid MHOAs. However, as seen in Table 15, GA-DE and PSO-CMA-ES showed relatively large errors (μ = 52.65, 43.41), while DE converged rapidly (11.87 s). PSO-GA, TS-PS, and DE-CMA-ES showed only small errors ($\mu \leq 0.05$). A gradual improving tendency toward convergence was found in SA-DE. As described in Section 4.1.3, ACO hybrids exhibited the slowest converging time. In the worst case, ACO-GA was finished after 1 min, whereas DECS-TAPS converged in only 1–15 s. As expected, DECS-TAPS showed remarkably better performance. In Fig. 15, the greater fitness variance of DECS-TAPS in the early process demonstrated the successful active exploration of the TS-based adaptive PS in the premature stage.

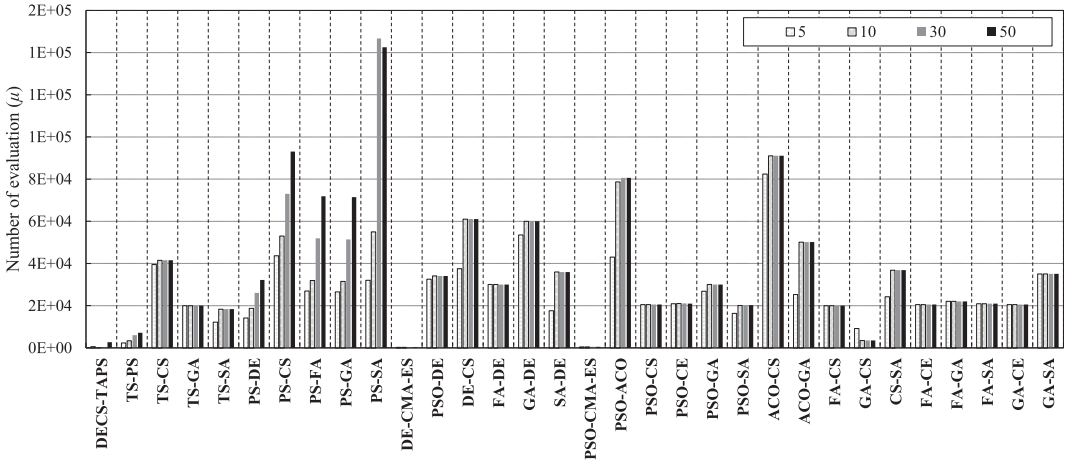
The convergence of multi-objective metaheuristics can also be characterized by the property of the Pareto optimal set (Pasha *et al.*, 2022), Accordingly, we analyzed the test algorithm performances with the recently developed metrics of multi-objective optimization: the number of Pareto front solutions (NPS), mean ideal distance (MID), and spread of non-dominance solutions (SNSs) (Fathollahi-Fard *et al.*, 2020; Pasha *et al.*, 2022). NPS and SNS relate to the diversification capacity and quality, respectively, while MID measures the closeness to the best solution and the capability of Pareto exploitation. In this test, MID and SNS were obtained after computing NPS, such as

Table 12: Performance test results of hybrid algorithms (T13, $n_d = 50$).

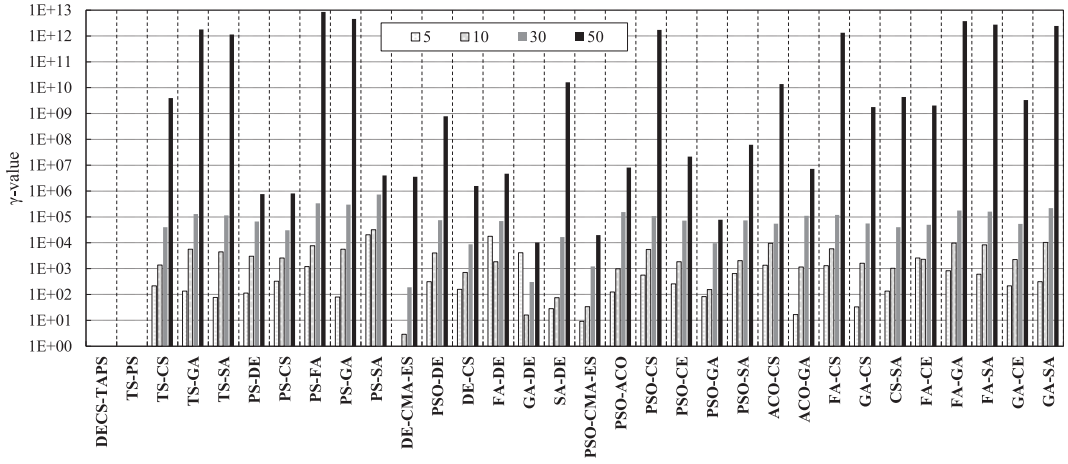
Hybrids	Min	Max	μ	σ	RMSE(x^*)	RMSE(f)	γ-value
DECS-TAPS	**518**	**21 998**	**3439**	**4636.066**	**0**	**0**	**0**
TS-PS	1856	51 502	7617	12 382.437	0.002	4.430E-06	0
TS-CS	41 541	41 541	41 541	0	1.328	3.822E + 09	7.94E + 09
TS-GA	20 040	20 040	20 040	0	3.412	3.547E + 12	3.55E + 12
TS-SA	18 377	18 377	18 377	0	3.422	2.486E + 12	2.28E + 12
PS-DE	36 729	38 667	37 762	729.206	0.917	6.336E + 05	1.20E + 06
PS-CS	93 041	93 041	93 041	0	0.805	2.268E + 05	1.06E + 06
PS-FA	71 319	71 937	71 845	190.817	3.587	2.722E + 12	9.78E + 12
PS-GA	71 231	71 540	71 448	102.362	3.308	1.402E + 12	5.01E + 12
PS-SA	238 369	238 369	238 369	0	1.021	5.403E + 05	6.44E + 06
DE-CMA-ES	255	255	255	0	1.158	3.731E + 08	4.76E + 06
PSO-DE	34 081	34 081	34 081	0	1.886	4.341E + 08	7.40E + 08
DE-CS	61 040	61 040	61 040	0	0.707	8.173E + 05	2.49E + 06
FA-DE	30 040	30 040	30 040	0	0.676	3.025E + 06	4.54E + 06
GA-DE	60 040	60 040	60 040	0	0.949	3.366E + 03	1.01E + 04
SA-DE	35 960	35 960	35 960	0	1.488	1.024E + 10	1.84E + 10
PSO-CMA-ES	510	510	510	0	1.038	1.004E + 06	2.56E + 04
PSO-ACO	80 540	80 540	80 540	0	0.996	3.813E + 06	1.54E + 07
PSO-CS	20 541	20 541	20 541	0	2.852	3.318E + 12	3.41E + 12
PSO-CE	20 842	20 998	20 930	55.870	1.410	4.051E + 07	4.24E + 07
PSO-GA	30 040	30 040	30 040	0	1.054	1.262E + 04	1.90E + 04
PSO-SA	19 810	20 445	20 205	213.334	1.263	1.215E + 08	1.23E + 08
ACO-CS	91 040	91 040	91 040	0	0.700	9.855E + 04	4.49E + 05
ACO-GA	50 120	50 200	50 133	26.667	1.052	5.538E + 06	1.39E + 07
FA-CS	20 040	20 040	20 040	0	2.765	2.659E + 12	2.66E + 12
GA-CS	3506	3506	3506	0	1.451	2.016E + 10	3.53E + 09
CS-SA	36 858	36 858	36 858	0	1.361	4.737E + 09	8.73E + 09
FA-CE	20 501	20 501	20 501	0	1.489	3.930E + 09	4.03E + 09
FA-GA	22 040	22 040	22 040	0	3.712	6.764E + 12	7.45E + 12
FA-SA	20 916	20 916	20 916	0	3.580	5.196E + 12	5.43E + 12
GA-CE	20 520	20 520	20 520	0	1.489	6.517E + 09	6.69E + 09
GA-SA	35 080	35 080	35 080	0	3.424	2.783E + 12	4.88E + 12

Table 13: Performance test results of hybrid algorithms (T14, $n_d = 50$).

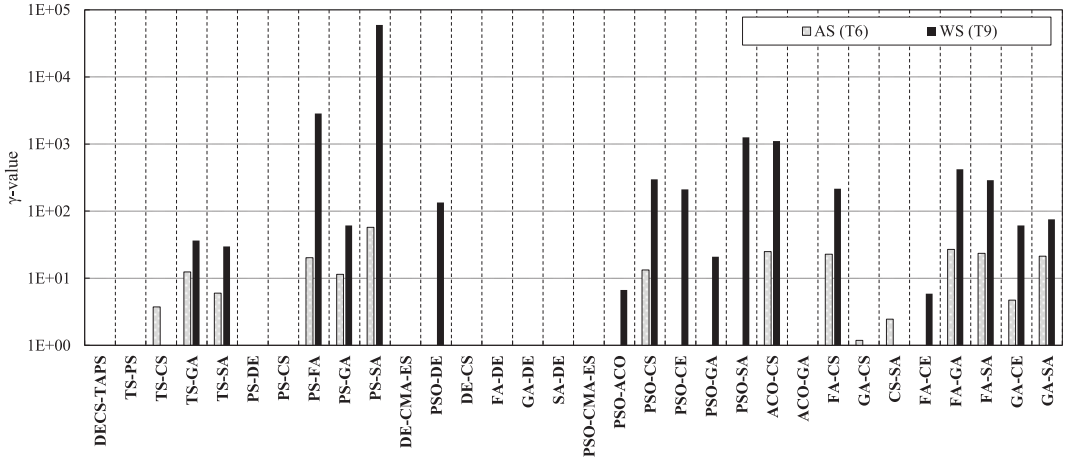
Hybrids	Min	Max	μ	σ	RMSE(x^*)	RMSE(f)	γ-value
DECS-TAPS	**872**	**5603**	**1996**	**1645.492**	**0**	**0**	**0**
TS-PS	2474	15 658	6738	5051.664	0.001	3.014E-06	1.02E-06
TS-CS	41 541	41 541	41 541	0	1.133	3.101E + 05	5.75E + 05
TS-GA	20 040	20 040	20 040	0	2.658	6.361E + 05	6.37E + 05
TS-SA	18 377	18 377	18 377	0	2.565	5.868E + 05	5.39E + 05
PS-DE	26 631	26 631	26 631	0	2.420	1.851E + 05	3.16E + 05
PS-CS	93 041	93 041	93 041	0	1.716	1.307E + 05	5.68E + 05
PS-FA	71 422	71 937	71 823	149.217	3.760	4.051E + 12	7.27E + 12
PS-GA	71 231	71 540	71 448	102.362	3.276	1.395E + 12	4.08E + 12
PS-SA	46 697	46 697	46 697	0	1.117	7.499E + 05	1.60E + 06
DE-CMA-ES	255	255	255	0	1.265	1.885E + 08	2.40E + 06
PSO-DE	34 081	34 081	34 081	0	1.798	4.815E + 08	8.20E + 08
DE-CS	61 040	61 040	61 040	0	0.677	2.174E + 05	6.64E + 05
FA-DE	30 040	30 040	30 040	0	0.674	3.189E + 06	4.79E + 06
GA-DE	60 040	60 040	60 040	0	0.949	3.395E + 03	1.02E + 04
SA-DE	35 960	35 960	35 960	0	1.454	7.798E + 09	1.40E + 10
PSO-CMA-ES	510	510	510	0	0.916	5.448E + 05	1.39E + 04
PSO-ACO	80 540	80 540	80 540	0	2.452	1.916E + 05	7.71E + 05
PSO-CS	20 541	20 541	20 541	0	2.092	5.368E + 05	5.51E + 05
PSO-CE	20 959	21 035	21 017	22.509	2.739	2.972E + 05	3.12E + 05
PSO-GA	30 040	30 040	30 040	0	1.871	9.007E + 04	1.35E + 05
PSO-SA	19 830	20 295	20 156	132.470	2.825	3.243E + 05	3.27E + 05
ACO-CS	91 040	91 040	91 040	0	0.764	5.992E + 09	2.73E + 10
ACO-GA	50 120	50 160	50 129	16.630	2.646	1.653E + 05	4.14E + 05
FA-CS	20 040	20 040	20 040	0	1.994	5.032E + 05	5.04E + 05
GA-CS	3506	3506	3506	0	1.515	4.722E + 05	4.33E + 05
CS-SA	36 858	36 858	36 858	0	1.541	2.925E + 05	5.39E + 05
FA-CE	20 501	20 501	20 501	0	1.668	2.797E + 05	2.87E + 05
FA-GA	22 040	22 040	22 040	0	2.734	7.278E + 05	8.02E + 05
FA-SA	20 916	20 916	20 916	0	2.696	7.141E + 05	7.47E + 05
GA-CE	20 520	20 520	20 520	0	1.457	3.440E + 05	3.53E + 05
GA-SA	35 080	35 080	35 080	0	2.705	6.043E + 05	1.06E + 06



(a) Space dimension: μ

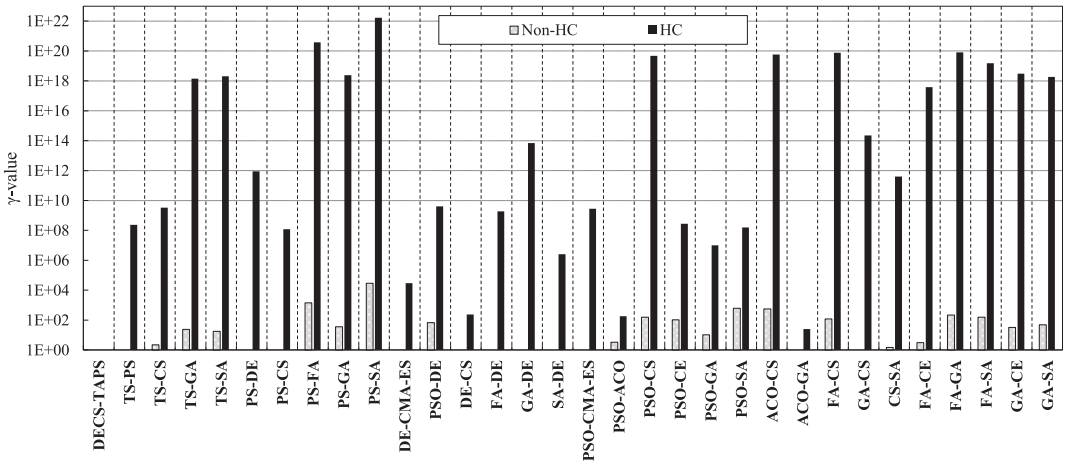


(b) Space dimension: γ-value



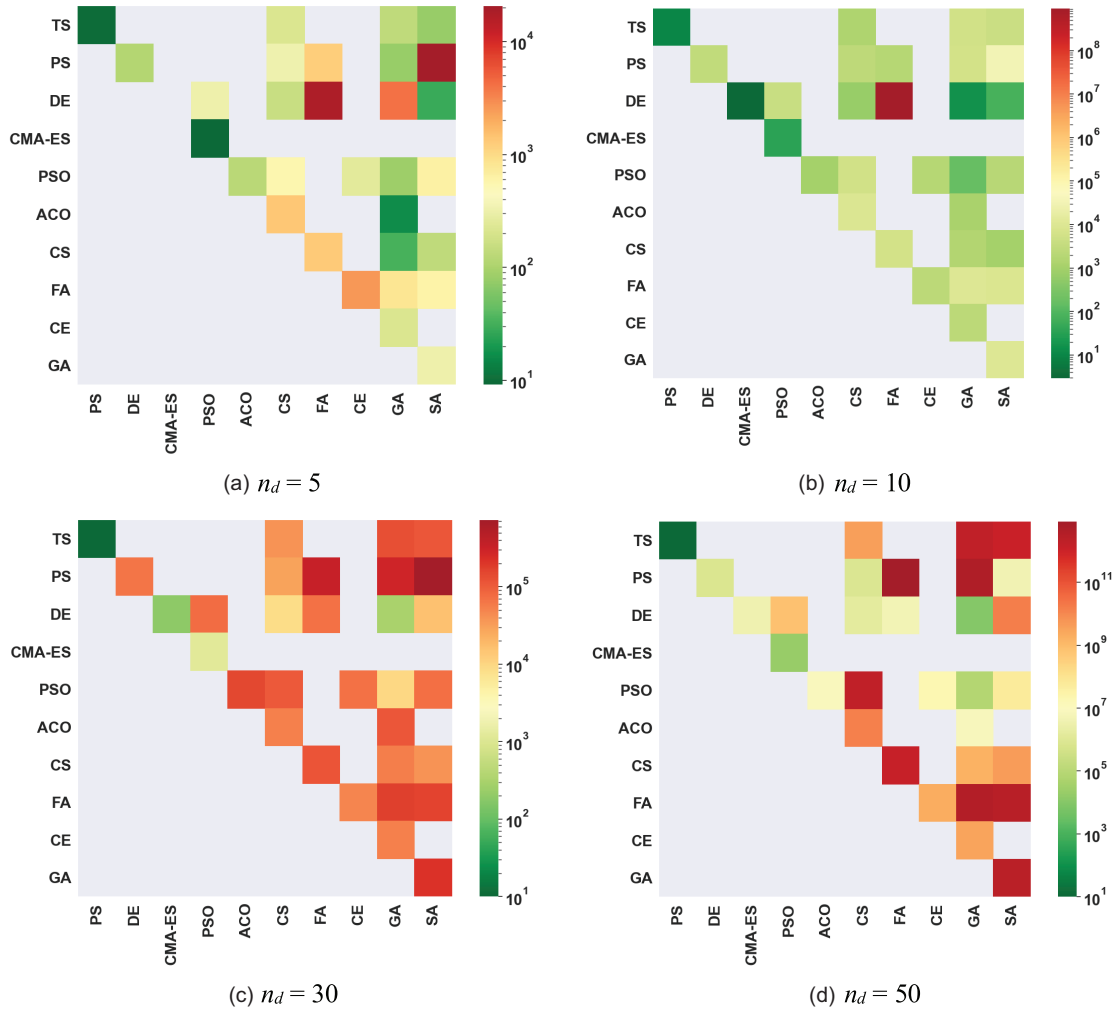
(c) Functional shape complexity: γ-value

Figure 10: Influence of dimension and function complexity on convergence quality.



(d) Fitness value conditining: γ-value

Figure 10 – *continued.*



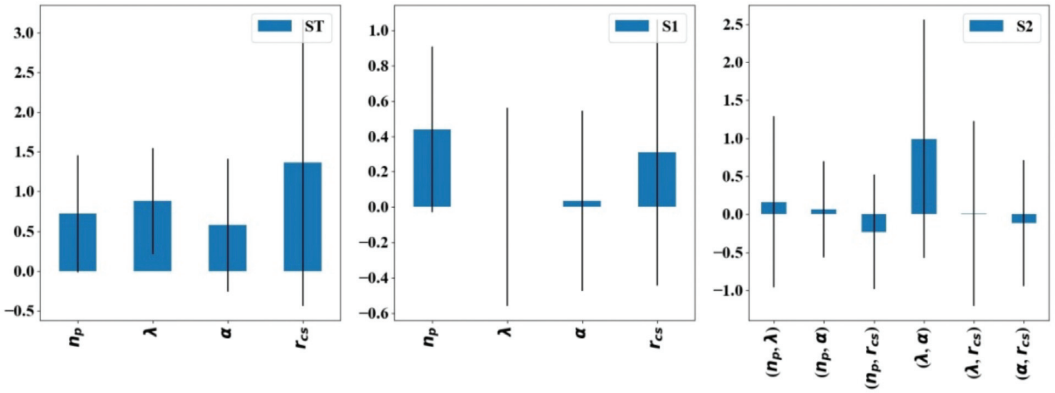
(a) $n_d = 5$ (b) $n_d = 10$

(c) $n_d = 30$ (d) $n_d = 50$

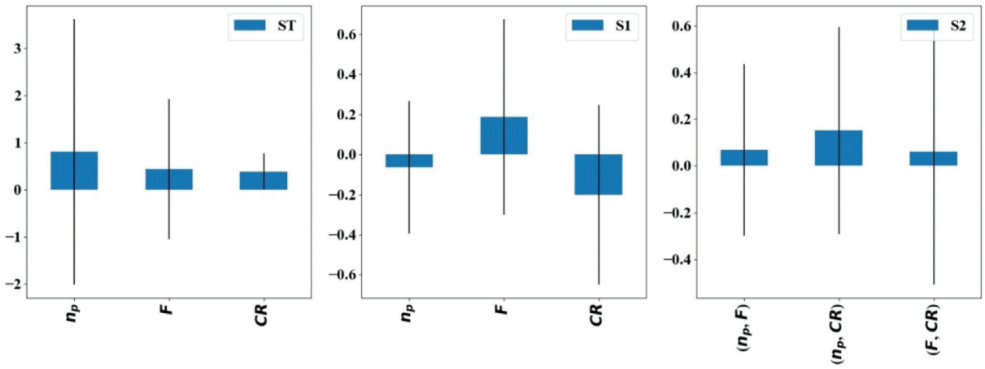
Figure 11: Heatmap: dimensional effect in pairwise hybridization (log-scale γ-values).

Table 14: Sensitivity analysis—average total-order indices.

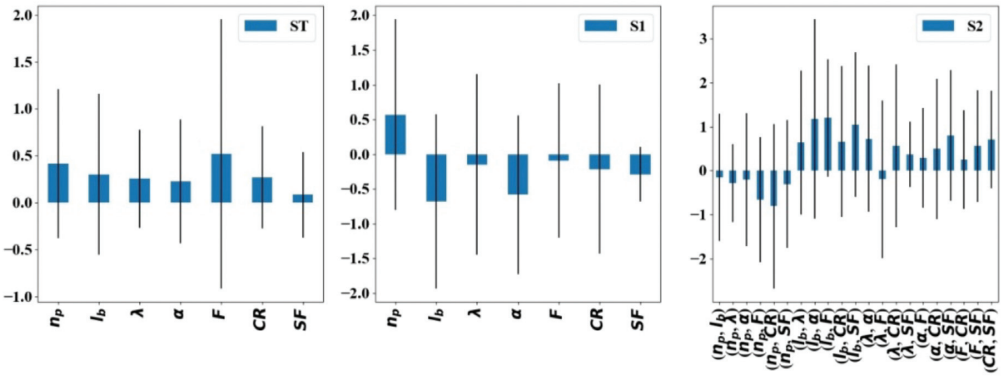
	n_p	l_{tb}	r_{cs}	λ	α	F	CR	sf
				$n_d = 10$				
CS	0.721		1.363	0.880	0.578			
DE	0.804					0.437	0.383	
DECS-TAPS	0.416	0.301		0.256	0.228	0.520	0.264	0.084
				$n_d = 30$				
CS	1.107		6.916	1.961	2.316			
DE	0.928					1.249	1.891	
DECS-TAPS	0.659	0.943		1.137	0.874	1.049	0.743	0.498



(a) CS



(b) DE



(c) DECS-TAPS

Figure 12: Parametric sensitivity ($n_d = 10$, ST/S1/S2: total/first-/second-order index).

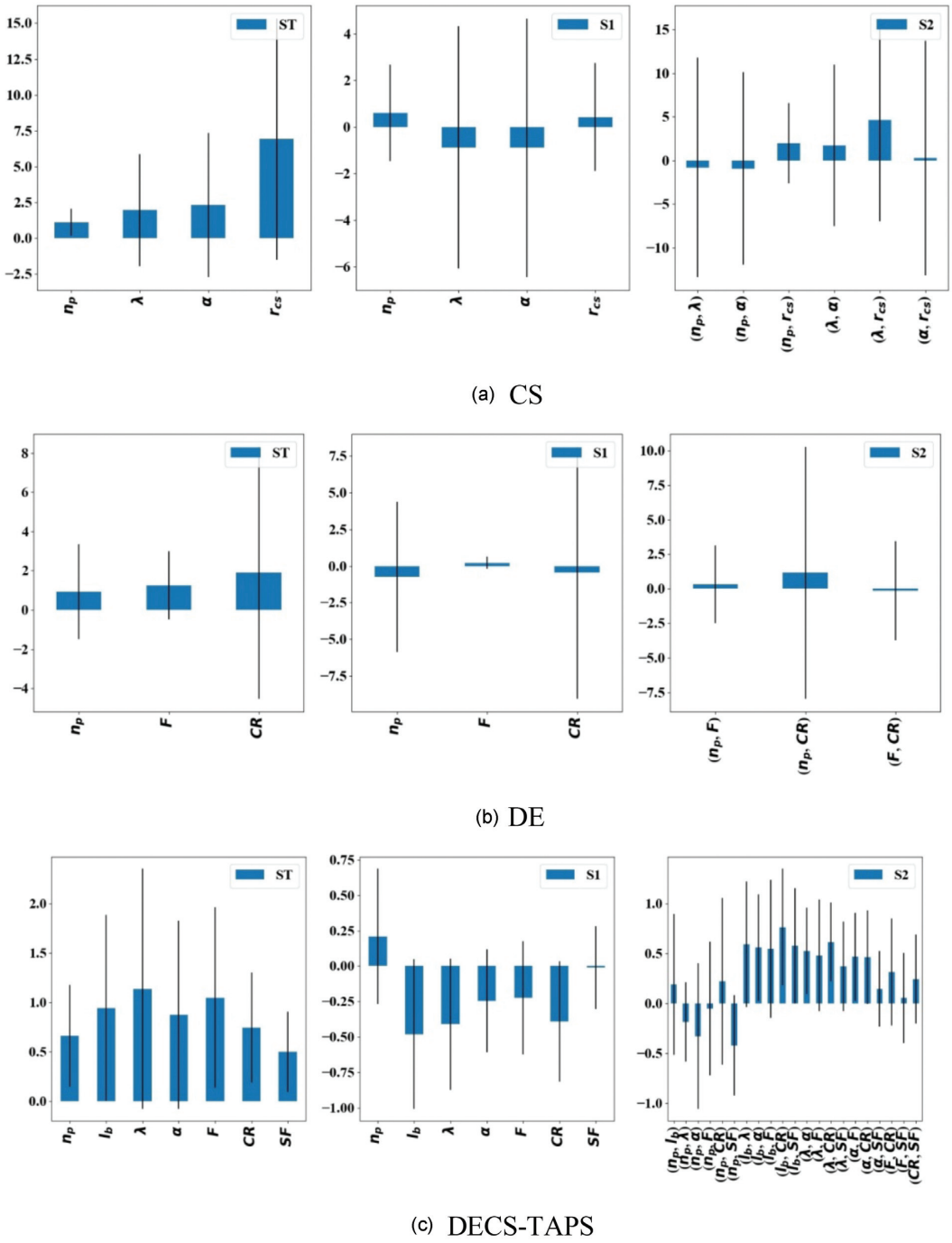


Figure 13: Parametric sensitivity ($n_d = 30$, ST/S1/S2: total/first-/second-order index).

$$\mathrm{MID} = \frac{1}{\mathrm{NPS}} \sum_{i=1}^{\mathrm{NPS}} \left(\sqrt{\sum_{j=1}^{2} \left(\frac{f_i^j - f_{\mathrm{best}}^j}{f_{\mathrm{max}}^j - f_{\mathrm{min}}^j} \right)^2} \right), \tag{11}$$

$$\mathrm{SNS} = \sqrt{\frac{1}{\mathrm{NPS} - 1} \sum_{i=1}^{\mathrm{NPS}} \left(\mathrm{MID} - \sum_{j=1}^{2} f_i^j \right)^2}, \tag{12}$$

where f_i^j and f_{best}^j are the i-th and ideal Pareto front solution value for the j-th objective, respectively. f_{min}^j and f_{max}^j denote the minimum and maximum values in all Pareto front solutions per objective, respectively. Note that the lower MID value indicates better performance and the higher is the better for NPS and SNS.

The assessment results (Table 16) show the larger NPS values (> 10) of DE, DE-CMA-ES, and PSO-GA. However, NPS polls a Pareto front set regardless of the population size or the iteration number. In our investigation, NPS was complemented by NPS*, NPS divided by the total number of entry evaluations, to estimate the likelihood of finding a Pareto solution in a one-time search. Nonetheless, NPS* results suggest that DECS-TAPS diversifies the Pareto front solutions very efficiently (NPS* = 36.27) than DE, CMA-ES

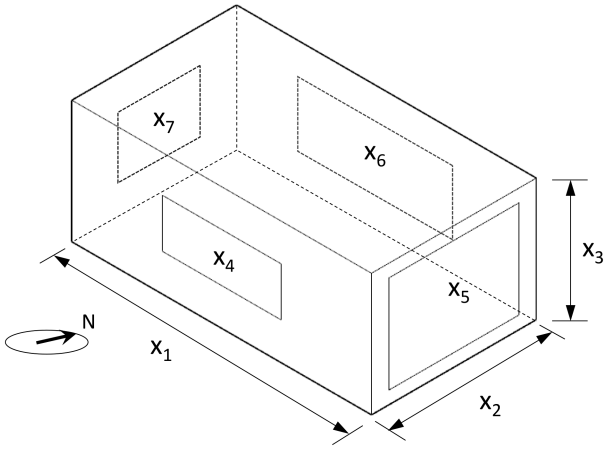


Figure 14: Shoebox test building geometry and design variables.

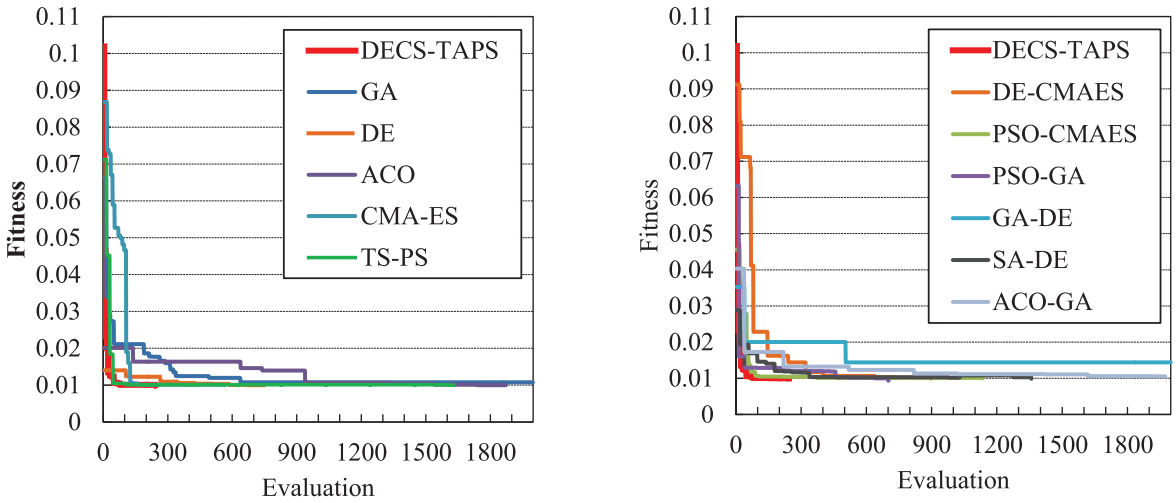


Figure 15: Convergence charts: algorithm comparison.

Table 15: Multi-objective ADO test results.

	GA	SA	DE	ACO	CMA-ES	DECS-TAPS	TS-PS	DE-CMAES	PSO-CMAES	PSO-GA	GA-DE	SA-DE	ACO-GA
						Iterations							
μ	**2000**	**2000**	**776**	**1935**	**563**	**246**	**1634**	**981**	**1134**	**703**	**2000**	**1360**	**1980**
Min	2000	2000	520	1340	99	78	77	410	36	100	2000	300	1620
Max	2000	2000	1480	2000	2000	1026	2000	1610	2000	2000	2000	2000	2000
σ	0.00	0.00	200.38	225.89	483.81	168.38	782.68	405.06	820.77	655.96	0.00	753.60	80.52
						RMSE(f), × E-6							
μ	**2.49**	**6.95**	**0.00**	**0.40**	**0.03**	**0.00**	**0.01**	**0.01**	**43.41**	**0.05**	**52.65**	**0.67**	**1.30**
Min	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.38	0.00	0.00
Max	7.60	19.61	0.00	3.83	0.21	0.00	0.02	0.15	1301.21	0.59	133.54	5.61	12.41
σ	2.69	6.67	0.00	0.95	0.06	0.00	0.01	0.03	237.56	0.15	46.40	1.40	3.24
						Time(s)							
μ	**31.11**	**38.40**	**11.87**	**28.64**	**12.98**	**3.42**	**23.83**	**16.33**	**36.04**	**10.33**	**27.91**	**21.38**	**41.47**
Min	25.23	24.42	7.17	18.11	1.83	1.01	1.09	5.76	0.68	1.43	27.01	4.08	23.60
Max	44.07	49.83	22.23	72.84	36.58	14.55	30.99	57.91	101.22	29.89	36.84	47.52	90.18
σ	8.30	12.05	3.72	10.82	11.02	2.39	11.42	10.73	34.93	9.42	1.78	12.28	21.19

Table 16: Assessment of Pareto optimal sets.

	GA	SA	DE	ACO	CMA-ES	DECS-TAPS	TS-PS	DE-CMAES	PSO-CMAES	PSO-GA	GA-DE	SA-DE	ACO-GA
						NPS							
μ	7.93	8.97	10.33	9.77	3.73	8.92	6.60	10.97	2.27	10.53	3.21	8.93	9
Min	4	6	6	5	2	5	3	7	2	5	2	4	3
Max	11	15	18	15	5	15	10	17	3	24	7	16	15
						NPS∗ (weighted NPS, $\times 1E\text{-}3$)							
μ	**3.97**	**4.48**	**13.32**	**5.05**	**6.63**	**36.27**	**4.04**	**11.18**	**2.00**	**14.98**	**1.60**	**6.57**	**4.55**
Min	2.00	3.00	7.73	2.58	3.55	20.33	1.84	7.14	1.76	7.11	1.00	2.94	1.52
Max	5.50	7.50	23.20	7.75	8.88	60.98	6.12	17.33	2.65	34.14	3.50	11.76	7.58
						MID							
μ	**0.66**	**0.99**	**0.46**	**0.53**	**0.64**	**0.45**	**0.42**	**0.43**	**2.93**	**0.39**	**0.69**	**0.51**	**0.52**
Min	0.39	0.36	0.30	0.29	0.46	0.30	0.26	0.31	0.52	0.20	0.48	0.31	0.33
Max	0.84	2.19	0.66	1.12	0.81	0.82	0.77	0.66	30.93	0.68	0.83	0.88	0.75
						SNS							
μ	**0.68**	**1.04**	**0.47**	**0.55**	**0.75**	**0.47**	**0.44**	**0.44**	**4.09**	**0.40**	**0.84**	**0.53**	**0.54**
Min	0.39	0.36	0.31	0.29	0.50	0.30	0.27	0.32	0.62	0.20	0.54	0.32	0.33
Max	0.87	2.29	0.68	1.21	0.98	0.86	0.83	0.69	43.72	0.72	0.98	1.00	0.90

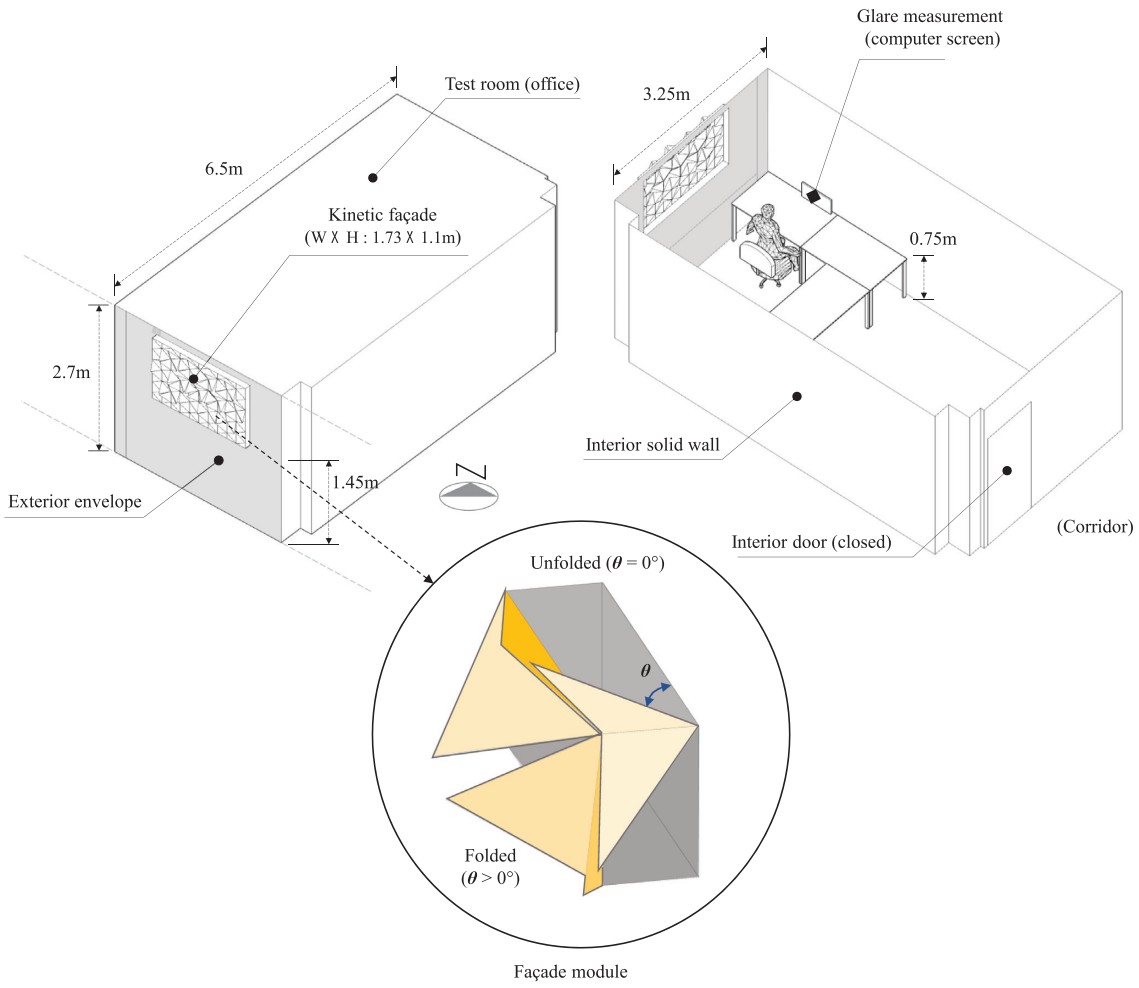


Figure 16: Experiment setup of dynamic ADO.

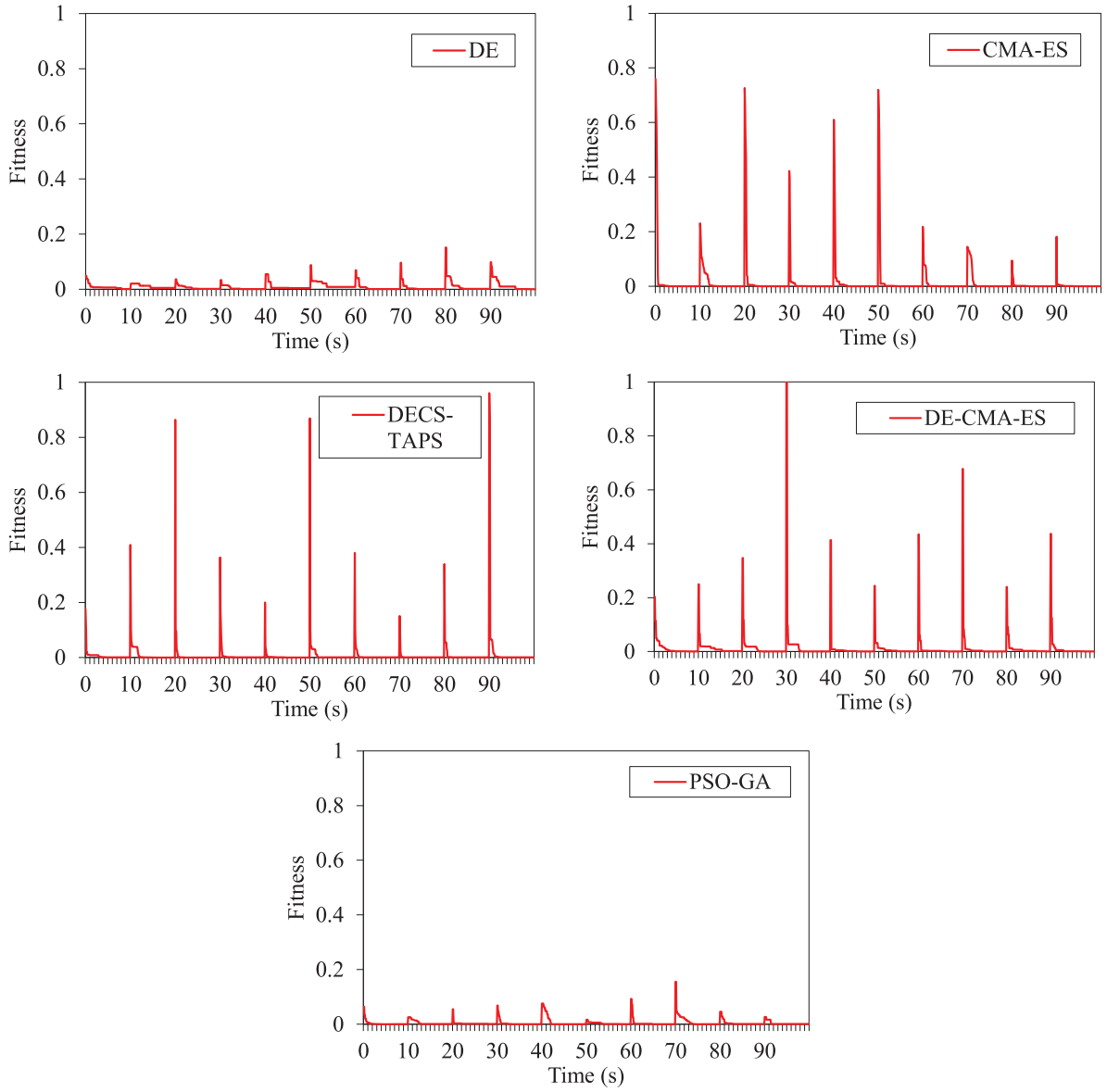


Figure 17: Convergence charts.

Table 17: Numeric results of kinetic building skin optimization.

		DE	CMA-ES	DECS-TAPS	DE-CMA-ES	PSO-GA
RMSE(f)	μ	2.60E-03	5.19E-05	7.93E-05	1.81E-04	4.89E-04
	Min	4.43E-04	1.17E-06	5.48E-06	2.04E-06	1.60E-04
	Max	8.74E-03	2.23E-04	1.89E-04	7.03E-04	1.00E-03
	σ	2.50E-03	8.48E-05	6.00E-05	2.35E-04	3.00E-04
Converging	μ	9.90	3.49	2.46	6.95	7.76
time (s) at	Min	9.00	2.20	1.00	3.40	2.00
$f \leq 0.0005$	Max	10.00	4.30	4.50	10.00	10.00
	σ	0.30	0.57	0.95	2.09	2.96

hybrids, and PSO-GA. In their intensification, PSO-GA slightly outperformed DECS-TAPS, but the DECS-TAPS's MID of 0.45 was as satisfactory as competing hybrid algorithms. DE and TS-PS were advantageous in multi-objective exploration and exploitation, respectively, and it explains the improved performance of their hybridization in DECS-TAPS. In SNS, PSO-CMA-ES was outstanding (SNS = 4.09), and SA (SNS = 1.04) showed a good spread quality. Despite the weak SNS of DECS-TAPS (SNS = 0.47), we suppose that

it was due to the intensive localization, as identified in Fig. 9, to increase accuracy during the latter stage of convergence. These results show that there is no single best algorithm satisfying every index. Nonetheless, we find that DECS-TAPS overwhelms other MHOAs in terms of rapid exploration.

4.3.2. *Time-sequential ADO: responsive building shading*

For advanced applications, we applied the five best algorithms in 4.3.1—DE, CMA-ES, DECS-TAPS, DE-CMA-ES, and PSO-GA—to the time-limited shape optimization of kinetic window shading. In EPBD, kinetic architectural design can improve environmental performance by dynamically morphing the building form. A critical concern in the construction of kinetic building systems is the optimal control of geometry change because highly rapid optimal decision making is required after sensing complex building information. As depicted in Fig. 16, we simulated a small office room with a kinetic shading skin [width (W) × height (H) = 1.73 × 1.1 m] over a front window.

13 kinetic modules comprising triangular movable panels populated the shading surface. Our design task was to create a desirable indoor work environment by minimizing both daylight glare on the computer screen and maximizing daylight illuminance by up to 500 lx on the desk plane area. No artificial lighting or building systems were considered in this experiment. The goal of this ADO was to find an optimal panel motion angle (θ) that minimizes the objective value under constantly varying solar position and outside illuminance. Note that RFR surrogate models were used in the same manner as Section 4.3.1 for simulating rapid daylight. By employing LHS and Radiance simulation, 1000 input–output data were used to create ML models of glare and illuminance, respectively. R^2 of the glare model was 0.7548 and that of the illuminance was 0.8371. The total test duration was 100 s. The time step was set to 10 s and the stopping criterion was set to $f(x^*) \leq 1E-4$.

Figure 17 and Table 17 show that the test algorithms except DE successfully converged in each time window. Our results reveal that although DECS-TAPS and DE-/PSO-CMA-ES underwent large fluctuations in the early phases, they quickly searched global optima in all the sequences. The CMA-ES's μ of RMSE(f) indicates that it can be slightly more accurate than DECS-TAPS. However, Table 17 shows that it risks taking relatively more time to process computation than DECS-TAPS.

5. Conclusions

The development of a high-performance optimization algorithm becomes an integral part of EPBD study in architecture because the use of a large number of geometric variables and exhaustive building performance simulation in EPBD practice calls for an advanced technique of mathematical optimization. Although GA or other standard EAs are widely employed in EPBD, the repetitive evaluation of populations often becomes the most prohibitive obstacle. In this study, by considering a multiple hybridization strategy, we sought to propose an alternative time-efficient hybrid algorithm by building on the knowledge of low-level (computationally uncomplicated) metaheuristic schemes.

DECS-TAPS was designed by adaptively integrating TS, CS, PS, and DE. This quadratic algorithmic synthesis, including a proper compromise between non-population-based search and DE, showed remarkable improvement in computation speed, accuracy, and robustness compared to the existing standard MHOAs and other pairwise hybrids. Our findings validated that the CS-enabled MADS mechanism reduced the exploration time in almost all types of complex problems. We also found that the quick early exploration followed by DE leads to robust exploitation toward convergence.

Compared to the well-established CMA-ES hybrids, a few large-dimensional test results showed that DECS-TAPS may not always ensure the best time-efficient performance, but it effectively improves solution quality. Additionally, reduced parametric sensitivity is another advantage. The global sensitivity analysis of DECS-TPAS suggests that an appropriate combination of the standard algorithms can prove competitive by relaxing the parametric dependence of the search mechanism.

Despite our efforts to prove the performative competence of DECS-TAPS through benchmark tests and architectural applications, many other complex problems remain untapped. These limitations will be addressed in a future study.

Authors' contributions

H.Y.: Funding acquisition, conceptualization, methodology, formal analysis, and writing—original draft. I.K.: Investigation and writing—review and editing.

Acknowledgments

This work was supported through the National Research Foundation of Korea (NRF) funded by the Korea government (MSIT) (Nos. NRF-2021R1C1C1003403 and NRF-2019R1A2C100913012) and Ajou University Research Grant (S-2021-G0001-00016).

Conflict of interest statement

None declared.

References

Abdel-Basset, M., Abdel-Fatah, L., & Sangaiah, A. K. (2018). Metaheuristic algorithms: A comprehensive review. In A. K. Sangaiah, M. Sheng, & Z. Zhang (Eds.), *Computational intelligence for multimedia big data on the cloud with engineering applications*(pp. 185–231). Academic Press.

Ahmed, R., Nazir, A., Mahadzir, S., Shorfuzzaman, M., & Islam, J. (2021). Niching grey wolf optimizer for multimodal optimization problems. *Applied Sciences*, **11**, 4795. https://doi.org/10.3390/app11114795.

Akimoto, Y., & Hansen, N. (2020). Diagonal acceleration for covariance matrix adaptation evolution strategies. *Evolutionary Computation*, **28**, 405–435. https://doi.org/10.1162/evco_a_00260.

Ali, A. F., & Tawhid, M. A. (2017). A hybrid particle swarm optimization and genetic algorithm with population partitioning for large scale optimization problems. *Ain Shams Engineering Journal*, **8**, 191–206. https://doi.org/10.1016/j.asej.2016.07.008.

Alkhateeb, F., & Abed-alguni, B. H. (2019). A hybrid cuckoo search and simulated annealing algorithm. *Journal of Intelligent Systems*, **28**, 683–698. https://doi.org/10.1515/jisys-2017-0268.

Amaran, S., Sahinidis, N. V., Sharda, B., & Bury, S. J. (2014). Simulation optimization: a review of algorithms and applications. *4OR*, **12**, 301–333. https://doi.org/10.1007/s10288-014-0275-2.

Araújo, T. F., & Uturbey, W. (2013). Performance assessment of PSO, DE and hybrid PSO–DE algorithms when applied to the dispatch of generation and demand. *International Journal of Electrical Power & Energy Systems*, **47**, 205–217. https://doi.org/10.1016/j.ijepes.2012.11.002.

Ataman, C., & Dino, İ. G. (2021). Performative design processes in architectural practices in turkey: Architects' perception. *Architectural Engineering and Design Management*, 1–15. https://doi.org/10.1080/17452007.2021.1995315.

Audet, C., & Dennis, J. E. (2006). Mesh adaptive direct search algorithms for constrained optimization. *SIAM Journal on Optimization*, **17**, 188–217. https://doi.org/10.1137/040603371.

Banks, A., Vincent, J., & Anyakoha, C. (2007). A review of particle swarm optimization. Part I: Background and development. *Natural Computing*, **6**, 467–484. https://doi.org/10.1007/s11047-007-9049-5.

Barrera, J., Álvarez-Bajo, O., Flores, J. J., & Coello, C. A. C. (2016). Limiting the velocity in the particle swarm optimization algorithm. *Computacion y Sistemas*, **20**, 635–645. https://doi.org/10.13053/cys-20-4-2505.

Benhamou, E., Atif, J., & Laraki, R. (2019). *A discrete version of CMA-ES*. Retreived from https://hal.archives-ouvertes.fr/hal-02011531v1/document.

Blue, J. A., & Bennett, K. P. (1998). Hybrid extreme point tabu search. *European Journal of Operational Research*, **106**, 676–688. https://doi.org/10.1016/S0377-2217(97)00297-X.

Bozorgi, S. M., & Yazdani, S. (2019). IWOA: An improved whale optimization algorithm for optimization problems. *Journal of Computational Design and Engineering*, **6**, 243–259. https://doi.org/10.1016/j.jcde.2019.02.002.

Brunetti, G. L. (2015). *Optimization as a design strategy. Considerations based on building simulation-assisted experiments about problem decomposition*. Retrieved from https://arxiv.org/abs/1407.5615.

Caraffini, F., Iacca, G., & Yaman, A. (2019). Improving (1+1) covariance matrix adaptation evolution strategy: a simple yet efficient approach. *AIP Conference Proceedings*, **2070**, 020004. https://doi.org/10.1063/1.5089971.

Ciccozzi, F., Feljan, J., Carlson, J., & Crnković, I. (2018). Architecture optimization: speed or accuracy? Both! *Software Quality Journal*, **26**, 661–684. https://doi.org/10.1007/s11219-016-9343-5.

Clerc, M., & Kennedy, J. (2002). The particle swarm—explosion, stability, and convergence in a multidimensional complex space. *IEEE Transactions on Evolutionary Computation* **6**, 58–73. https://doi.org/10.1109/4235.985692.

Deb, K., & Srivastava, S. (2012). A genetic algorithm based augmented lagrangian method for constrained optimization. *Computational Optimization and Applications*, **53**, 869–902. https://doi.org/10.1007/s10589-012-9468-9.

Deng, W., Chen, R., He, B., Liu, Y., Yin, L., & Guo, J. (2012). A novel two-stage hybrid swarm intelligence optimization algorithm and application. *Soft Computing*, **16**, 1707–1722. https://doi.org/10.1007/s00500-012-0855-z.

Devi, S. G., & Sabrigiriraj, M. (2019). A hybrid multi-objective firefly and simulated annealing based algorithm for big data classification. *Concurrency and Computation: Practice and Experience*, **31**, e4985.

Ding, J., Wang, Q., Zhang, Q., Ye, Q., & Ma, Y. (2019). A hybrid particle swarm optimization-cuckoo search algorithm and its engineering applications. *Mathematical Problems in Engineering*, **2019**, 5213759. https://doi.org/10.1155/2019/5213759.

Dino, I. G. (2016). An evolutionary approach for 3D architectural space layout design exploration. *Automation in Construction*, **69**, 131–150. https://doi.org/10.1016/j.autcon.2016.05.020.

Dorigo, M., Maniezzo, V., & Colorni, A. (1996). Ant system: optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics Part B (Cybernetics)*, **26**, 29–41. https://doi.org/10.1109/3477.484436.

Dorigo, M., Birattari, M., & Stutzle, T. (2006). Ant colony optimization. *IEEE Computational Intelligence Magazine*, **1**, 28–39. https://doi.org/10.1109/MCI.2006.329691.

Technical University of Denmark. (2018). *TopOpt plugin for Rhino and Grasshopper*. Available at: https://www.topopt.mek.dtu.dk/apps-and-software/topopt-plugin-for-rhino-and-grasshopper. Accessed on: June 12th, 2021.

Eberhart, R. C., & Shi, Y. (2000). Comparing inertia weights and constriction factors in particle swarm optimization. In *Proceedings of the 2000 Congress on Evolutionary Computation CEC00*(Cat No00TH8512). https://doi.org/10.1109/CEC.2000.870279.

El-mihoub, T., Hopgood, A., Nolle, L., & Alan, B. (2006). Hybrid genetic algorithms: A review. *Engineering Letters*, **3**. 124–137. http://irep.ntu.ac.uk/id/eprint/10082.

Fan, W., Ligui, L., Xing-shi, H., & Yan, W. (2011). Hybrid optimization algorithm of PSO and cuckoo search. In *2nd International Conference on Artificial Intelligence, Management Science and Electronic Commerce (AIMSEC)*. https://doi.org/10.1109/AIMSEC.2011.6010750.

Fathollahi-Fard, A. M., Hajiaghaei-Keshteli, M., & Mirjalili, S. (2018). Multi-objective stochastic closed-loop supply chain network design with social considerations. *Applied Soft Computing*, **71**, 505–525. https://doi.org/10.1016/j.asoc.2018.07.025.

Fathollahi-Fard, A. M., Hajiaghaei-Keshteli, M., & Tavakkoli-Moghaddam, R. (2020). Red deer algorithm (RDA): a new nature-inspired meta-heuristic. *Soft Computing*, **24**, 14637–14665. https://doi.org/10.1007/s00500-020-04812-z.

Fathollahi-Fard, A. M., Dulebenets, M. A., Hajiaghaei–Keshteli, M., Tavakkoli-Moghaddam, R., Safaeian, M., & Mirzahosseinian, H. (2021). Two hybrid meta-heuristic algorithms for a dual-channel closed-loop supply chain network design problem in the tire industry under uncertainty. *Advanced Engineering Informatics*, **50**, 101418. https://doi.org/10.1016/j.aei.2021.101418.

Ficarella, E., Lamberti, L., & Degertekin, S. O. (2021). Comparison of three novel hybrid metaheuristic algorithms for structural optimization problems. *Computers & Structures*, **244**, 106395. https://doi.org/10.1016/j.compstruc.2020.106395.

Flöry, S., Schmiedhofer, H., & Reis, M. (2018). Goat – Free optimization solver component for rhino's grasshopper. Retrieved from http://www.rechenraum.com/goat/.

Forbes, R., & Teli, M. N. (2006). Particle swarm optimization on multi-funnel functions. In *Computer aided optimum design in engineering XII*(Vol. **255**). https://doi.org/10.1.1.545.5837.

Garai, G., & Chaudhurii, B. B. (2013). A novel hybrid genetic algorithm with tabu search for optimizing multi-dimensional functions and point pattern recognition. *Information Sciences*, **221**, 28–48. https://doi.org/10.1016/j.ins.2012.09.012.

Glover, F., & Marti, R. (2006). Tabu search. In E. Alba, & R. Marti (Eds.), *Metaheuristic procedures for training neutral networks*. Springer.

Hansen, N., & Ostermeier, A. (1996). Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation. In *Proceedings of IEEE International Conference on Evolutionary Computation*. https://doi.org/10.1109/ICEC.1996.542381.

Hansen, N., Ros, R., Mauny, N., Schoenauer, M., & Auger, A. (2008). Impacts of invariance in search: when CMA-ES and PSO face ill-conditioned and non-separable problems. *Applied Soft Computing*, **11**, 5755–5769. https://doi.org/10.1016/j.asoc.2011.03.001.

Hansen, N., Auger, A., Ros, R., Mersmann, O., Tušar, T., & Brockhoff, D. (2021). COCO: A platform for comparing continuous optimizers in a black-box setting. *Optimization Methods and Software*, **36**, 114–144. https://doi.org/10.1080/10556788.2020.1808977.

Hasan, A., & Palonen, M. (2013). Mobo a new software for multi-objective building performance optimization. In *13th Conference of International Building Performance Simulation Association*. http://www.ibpsa.org/proceedings/bs2013/p_1489.pdf.

Hedar, A.-R., & Fukushima, M. (2004). Heuristic pattern search and its hybridization with simulated annealing for nonlinear global optimization. *Optimization Methods and Software*, **19**, 291–308. https://doi.org/10.1080/10556780310001645189.

Hedar, A.-R., & Fukushima, M. (2006). Tabu search directed by direct search methods for nonlinear global optimization. *European Journal of Operational Research*, **170**, 329–349. https://doi.org/10.1016/j.ejor.2004.05.033.

Hensen, J. L. M., & Lamberts, R. (Eds.). (2011). *Building performance simulation for design and operation*, ISBN 9781138392199. Spon Press.

Holland, J. H. (1975). *Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control, and artificial intelligence*, ISBN: 9780262275552. U Michigan Press. https://ieeexplore.ieee.org/servlet/opac?bknumber=6267401.

Il-Seok, O., Jin-Seon, L., & Byung-Ro, M. (2004). Hybrid genetic algorithms for feature selection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **26**, 1424–1437. https://doi.org/10.1109/TPAMI.2004.105.

Islam, M. R., Ali, S. M., Fathollahi-Fard, A. M., & Kabir, G. (2021). A novel particle swarm optimization-based grey model for the prediction of warehouse performance. *Journal of Computational Design and Engineering*, **8**, 705–727. https://doi.org/10.1093/jcde/qwab009.

Jamil, M., & Yang, X.-S. (2013). A literature survey of benchmark functions for global optimization problems. *International Journal of Mathematical Modelling and Numerical Optimisation*, **4**, 150–194. https://doi.org/10.1504/IJMMNO.2013.055204.

Javidrad, F., & Nazari, M. (2017). A new hybrid particle swarm and simulated annealing stochastic optimization method. *Applied Soft Computing*, **60**, 634–654. https://doi.org/10.1016/j.asoc.2017.07.023.

Jin, J., Yang, C., & Zhang, Y. (2020). An improved CMA-ES for solving large scale optimization problem. In *Advances in Swarm Intelligence*. Springer International Publishing. https://doi.org/10.1007/978-3-030-53956-6_34.

Juang, C. F. (2004). A hybrid of genetic algorithm and particle swarm optimization for recurrent network design. *IEEE Transactions on Systems, Man, and Cybernetics. Part B, Cybernetics*, **34**, 997–1006. https://doi.org/10.1109/TSMCB.2003.818557.

Kanagaraj, G., Ponnambalam, S. G., & Jawahar, N. (2013). A hybrid cuckoo search and genetic algorithm for reliability–redundancy allocation problems. *Computers & Industrial Engineering*, **66**, 1115–1124. https://doi.org/10.1016/j.cie.2013.08.003.

Kao, Y.-T., & Zahara, E. (2008). A hybrid genetic algorithm and particle swarm optimization for multimodal functions. *Applied Soft Computing*, **8**, 849–857. https://doi.org/10.1016/j.asoc.2007.07.002.

Katoch, S., Chauhan, S. S., & Kumar, V. (2021). A review on genetic algorithm: Past, present, and future. *Multimedia Tools and Applications*, **80**, 8091–8126. https://doi.org/10.1007/s11042-020-10139-6.

Kheiri, F. (2018). A review on optimization methods applied in energy-efficient building geometry and envelope design. *Renewable and Sustainable Energy Reviews*, **92**, 897–920. https://doi.org/10.1016/j.rser.2018.04.080.

Kıran, M. S., Gündüz, M., & Baykan, Ö. K. (2012). A novel hybrid algorithm based on particle swarm and ant colony optimization for finding the global minimum. *Applied Mathematics and Computation*, **219**, 1515–1521. https://doi.org/10.1016/j.amc.2012.06.078.

Kohler, N., & Moffatt, S. (2003). Life-cycle analysis of the built environment. *Industry and Environment*, **26**, 17–21.

Kolarevic, B., & Malkawi, A. (2005). *Performative architecture: Beyond instrumentality, ISBN:* 9780415700832. Spon Press.

Krause, O., Arbonès, D. R., & Igel, C. (2016). CMA-ES with optimal covariance update and storage complexity. In *NIPS 2016: The Thirtieth Annual Conference on Neural Information Processing Systems*. https://doi.org/10.5555/3157096.3157138.

Kroese, D. P., Porotsky, S., & Rubinstein, R. Y. (2006). The cross-entropy method for continuous multi-extremal optimization. *Methodology and Computing in Applied Probability*, **8**, 383–407. https://doi.org/10.1007/s11009-006-9753-0.

Kuo, R. J., & Han, Y. S. (2011). A hybrid of genetic algorithm and particle swarm optimization for solving bi-level linear programming problem—a case study on supply chain model. *Applied Mathematical Modelling*, **35**, 3905–3917. https://doi.org/10.1016/j.apm.2011.02.008.

Lanza Volpe, A. (2018). Building optimization: The adaptive façade. In *Advanced materials research*(Vol. **1149**, pp. 64–75). Trans Tech Publications, Ltd. https://doi.org/10.4028/www.scientific.net/amr.1149.64.

Li, X., & Gao, L. (2016). An effective hybrid genetic algorithm and tabu search for flexible job shop scheduling problem. *International Journal of Production Economics*, **174**, 93–110. https://doi.org/10.1016/j.ijpe.2016.01.016.

Li, W. D., Ong, S. K., & Nee, A. Y. C. (2002). Hybrid genetic algorithm and simulated annealing approach for the optimization of process plans for prismatic parts. *International Journal of Production Research*, **40**, 1899–1922. https://doi.org/10.1080/00207540110119991.

Li, G., Liu, P., Le, C., & Zhou, B. (2019). A novel hybrid meta-heuristic algorithm based on the cross-entropy method and firefly algorithm for global optimization. *Entropy*, **21**. 494–513 https://doi.org/10.3390/e21050494.

Li, S., Liu, L., & Peng, C. (2020). A review of performance-oriented architectural design and optimization in the context of sustainability: Dividends and challenges. *Sustainability*, **12**. 1427–1462 https://doi.org/10.3390/su12041427.

Liu, X., & Fu, H. (2014). PSO-Based support vector machine with cuckoo search technique for clinical disease diagnoses. *The Scientific World Journal*, **2014**, 548483. https://doi.org/10.1155/2014/548483.

Loftness, V., Lam, K. P., & Hartkopf, V. (2005). Education and environmental performance-based design: A carnegie mellon perspective. *Building Research & Information*, **33**, 196–203. https://doi.org/10.1080/0961321042000325336.

Lopez-Garcia, P., Onieva, E., Osaba, E., Masegosa, A. D., & Perallos, A. (2016). A hybrid method for short-term traffic congestion forecasting using genetic algorithms and cross entropy. *IEEE Transactions on Intelligent Transportation Systems*, **17**, 557–569. https://doi.org/10.1109/TITS.2015.2491365.

Mahi, M., Baykan, Ö. K., & Kodaz, H. (2015). A new hybrid method based on particle swarm optimization, ant colony optimization and 3-Opt algorithms for traveling salesman problem. *Applied Soft Computing*, **30**, 484–490. https://doi.org/10.1016/j.asoc.2015.01.068.

Mantegna, R. N. (1994). Fast, accurate algorithm for numerical simulation of levy stable stochastic processes. *Physical Review E*, **49**, 4677–4683. https://doi.org/10.1103/PhysRevE.49.4677.

Mirjalili, S., & Lewis, A. (2016). The whale optimization algorithm. *Advances in Engineering Software*, **95**, 51–67. https://doi.org/10.1016/j.advengsoft.2016.01.008.

Mohamed, A. W. (2017). Solving large-scale global optimization problems using enhanced adaptive differential evolution algorithm.

Complex & Intelligent Systems **3**, 205–231. https://doi.org/10.1007/s40747-017-0041-0.

Mohamed, A. W., Hadi, A. A., Fattouh, A. M., & Jambi, K. M. (2017). LSHADE with semi-parameter adaptation hybrid with CMA-ES for solving CEC 2017 benchmark problems. In *2017 IEEE Congress on Evolutionary Computation (CEC)*(pp. 145–152). https://doi.org/10.1109/CEC.2017.7969307.

Mori, N., Kita, H., & Nishikawa, Y. (1996). Adaptation to a changing environment by means of the thermodynamical genetic algorithm. In H.-M. Voigt, W. Ebeling, I. Rechenberg, & H.-P Schwefel. (Eds.), *Parallel problem solving from nature, Lecture Notes in Computer Science*(Vol. **1141**, pp. 513–522). Springer.

Moser, I. (2009). Hooke-Jeeves revisited 2009 IEEE Congress on Evolutionary Computation, IEEE 2009 IEEE Congress on Evolutionary Computation, Trondheim, Norway 2670–2676. https://doi.org/10.1109/CEC.2009.4983277.

Nancharaiah, B., & Mohan, B. C. (2014). Hybrid optimization using ant colony optimization and cuckoo search in MANET routing. In *2014 International Conference on Communication and Signal Processing*. https://doi.org/10.1109/ICCSP.2014.6950142.

Nassef, A. O., Hegazi, H. A., & Metwalli, S. M. (2000). A hybrid genetic-direct search algorithm for the shape optimization of solid C-frame cross-sections. In *Proceedings of the ASME Design Engineering Technical Conferences*(pp. 131–139). https://doi.org/10.1115/DETC2000/DAC-14295.

Nithya, B., & Jeyachidra, J. (2021). Hybrid approach of genetic algorithm and differential evolution in WSN localization. In *2021 International Conference on Innovative Computing, Intelligent Communication and Smart Electrical Systems (ICSES)*(pp. 1–6). https://doi.org/10.1109/ICSES52305.2021.9633962.

Opara, K., & Arabas, J. (2018). Comparison of mutation strategies in differential evolution—a probabilistic perspective. *Swarm and Evolutionary Computation*, **39**, 53–69. https://doi.org/10.1016/j.swevo.2017.12.007.

Osman, I. H., & Christofides, N. (1994). Capacitated clustering problems by hybrid simulated annealing and tabu search. *International Transactions in Operational Research*, **1**, 317–336. https://doi.org/10.1016/0969-6016(94)90032-9.

Pandian, V. (2011). Hybrid mesh adaptive direct search and genetic algorithms techniques for industrial production systems. *Archives of Control Sciences*, **21**, 299–312. https://doi.org/10.2478/v10170-010-0045-0.

Pasha, J., Nwodu, A. L., Fathollahi-Fard, A. M., Tian, G., Li, Z., Wang, H., & Dulebenets, M. A. (2022). Exact and metaheuristic algorithms for the vehicle routing problem with a factory-in-a-box in multi-objective settings. *Advanced Engineering Informatics*, **52**, 101623. https://doi.org/10.1016/j.aei.2022.101623.

Ponsich, A., & Coello Coello, C. A. (2013). A hybrid differential evolution—tabu search algorithm for the solution of job-shop scheduling problems. *Applied Soft Computing*, **13**, 462–474. https://doi.org/10.1016/j.asoc.2012.07.034.

Price, K., Storn, R. M., & Lampinen, J. A. (2005). *Differential evolution: A practical approach to global optimization*. Springer-Verlag. https://doi.org/10.1007/3-540-31306-0.

Qin, A. K., Huang, V. L., & Suganthan, P. N. (2009). Differential evolution algorithm with strategy adaptation for global numerical optimization. *IEEE Transactions on Evolutionary Computation*, **13**, 398–417. https://doi.org/10.1109/TEVC.2008.927706.

Rahmani, A., & MirHassani, S. A. (2014). A hybrid firefly-genetic algorithm for the capacitated facility location problem. *Information Sciences*, **283**, 70–78. https://doi.org/10.1016/j.ins.2014.06.002.

Rahmani, R., Yusof, R., Seyedmahmoudian, M., & Mekhilef, S. (2013). Hybrid technique of ant colony and particle swarm optimization for short term wind energy forecasting. *Journal of Wind Engineering and Industrial Aerodynamics*, **123**, 163–170. https://doi.org/10.1016/j.jweia.2013.10.004.

Sahu, R. K., Panda, S., & Chandra Sekhar, G. T. (2015a). A novel hybrid PSO-PS optimized fuzzy PI controller for AGC in multi area interconnected power systems. *International Journal of Electrical Power & Energy Systems*, **64**, 880–893. https://doi.org/10.1016/j.ijepes.2014.08.021.

Sahu, R. K., Panda, S., & Padhan, S. (2015b). A hybrid firefly algorithm and pattern search technique for automatic generation control of multi area power systems. *International Journal of Electrical Power & Energy Systems*, **64**, 9–23. https://doi.org/10.1016/j.ijepes.2014.07.013.

Salahi, M., Jamalian, A., & Taati, A. (2013). Global minimization of multi-funnel functions using particle swarm optimization. *Neural Computing and Applications*, **23**, 2101–2106. https://doi.org/10.1007/s00521-012-1158-0.

Sha, D. Y., & Hsu, C.-Y. (2006). A hybrid particle swarm optimization for job shop scheduling problem. *Computers & Industrial Engineering*, **51**, 791–808. https://doi.org/10.1016/j.cie.2006.09.002.

Shehab, M., Khader, A. T., & Al-Betar, M. A. (2017). A survey on applications and variants of the cuckoo search algorithm. *Applied Soft Computing*, **61**, 1041–1059. https://doi.org/10.1016/j.asoc.2017.02.034.

Shi, X. (2010). Performance-based and performance-driven architectural design and optimization. *Frontiers of Architecture and Civil Engineering in China*, **4**, 512–518. https://doi.org/10.1007/s11709-010-0090-6.

Shi, X., Tian, Z., Chen, W., Si, B., & Jin, X. (2016). A review on building energy efficient design optimization rom the perspective of architects. *Renewable and Sustainable Energy Reviews*, **65**, 872–884. https://doi.org/10.1016/j.rser.2016.07.050.

Shieh, H.-L., Kuo, C.-C., & Chiang, C.-M. (2011). Modified particle swarm optimization algorithm with simulated annealing behavior and its numerical verification. *Applied Mathematics and Computation*, **218**, 4365–4383. https://doi.org/10.1016/j.amc.2011.10.012.

Soke, A., & Bingul, Z. (2006). Hybrid genetic algorithm and simulated annealing for two-dimensional non-guillotine rectangular packing problems. *Engineering Applications of Artificial Intelligence*, **19**, 557–567. https://doi.org/10.1016/j.engappai.2005.12.003.

Song, J., & Sun, S. (2021). Research on architectural form optimization method based on environmental performance-driven design. In *Proceedings of the 2020 DigitalFUTURES*(pp. 217–228). https://doi.org/10.1007/978-981-33-4400-6_21.

Source Code for Module barecmaes2. (2014). http://www.cmap.polytechnique.fr/~nikolaus.hansen/html-pythonbarecma2/frames.html. Accessed on: 9/11/2021.

Su, Z., & Yan, W. (2015). A fast genetic algorithm for solving architectural design optimization problems. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, **29**, 457–469. https://doi.org/10.1017/S089006041500044X.

Sutton, A. M., Whitley, D., Lunacek, M., & Howe, A. (2006). PSO and multi-funnel landscapes: How cooperation might limit exploration. In *Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation*(pp., 75–82). Association for Computing Machinery. https://doi.org/10.1145/1143997.1144008.

Szynkiewicz, P. (2018). Comparative study of PSO and CMA-ES algorithms on Black-box optimization benchmarks. *Journal of Telecommunications and Information Technology*, **4**, 5–17. https://doi.org/10.26636/jtit.2018.127418.

Talbi, E. G. A. (2002). Taxonomy of hybrid metaheuristics. *Journal of Heuristics*, **8**, 541–564. https://doi.org/10.1023/A:1016540724870

Tawhid, M. A., & Ali, A. F. (2016). Direct search firefly algorithm for solving global optimization problems. *Applied Mathematics and Information Sciences*, **10**, 841–860. http://dx.doi.org/10.18576/amis/100304.

Tayfur, B., Yilmaz, H., & Daloğlu, A. T. (2021). Hybrid tabu search algorithm for weight optimization of planar steel frames. *Engineering Optimization*, **53**, 1369–1383. https://doi.org/10.1080/0305215X.2020.1793977.

Terki, A., & Boubertakh, H. (2021). A new hybrid binary-real coded cuckoo search and tabu search algorithm for solving the unit-commitment problem. *International Journal of Energy Optimization and Engineering*, **10**, 104–119. https://doi.org/10.4018/IJEOE.2021040105.

Trivedi, A., Srinivasan, D., Biswas, S., & Reindl, T. (2016). A genetic algorithm—differential evolution based hybrid framework: Case study on unit commitment scheduling problem. *Information Sciences*, **354**, 275–300. https://doi.org/10.1016/j.ins.2016.03.023.

Vasant, P., & Barsoum, N. (2010). Hybrid pattern search and simulated annealing for fuzzy production planning problems. *Computers & Mathematics with Applications*, **60**, 1058–1067. https://doi.org/10.1016/j.camwa.2010.03.063.

Waibel, C., Wortmann, T., Evins, R., & Carmeliet, J. (2019). Building energy optimization: An extensive benchmark of global search algorithms. *Energy and Buildings*, **187**, 218–240. https://doi.org/10.1016/j.enbuild.2019.01.048.

Wang, G., Guo, L., Duan, H., Wang, H., Liu, L., & Shao, M. (2012). A hybrid metaheuristic DE/CS algorithm for UCAV three-dimension path planning. *The Scientific World Journal*, **2012**, 583973. https://doi.org/10.1100/2012/583973.

Wang, X., Zhao, H., Wei, Z., Liang, Y., Li, Y., & Han, T. (2019). A novel CMA-ES with an eigen coordinates framework for parameter identification in photovoltaic models. In *2019 IEEE Congress on Evolutionary Computation (CEC)*. https://doi.org/10.1109/CEC.2019.8789900.

Wang, P., Bai, J., & Meng, J. (2020). A hybrid genetic ant colony optimization algorithm with an embedded cloud model for continuous optimization. *Journal of Information Processing Systems*, **16**, 1169–1182. https://doi.org/10.3745/JIPS.01.0059.

Wetter, M. (2018). GenOpt: Generic optimization program, 2018. Retrieved from https://simulationresearch.lbl.gov/GO/.

Wolpert, D. H., & Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, **1**, 67–82. https://doi.org/10.1109/4235.585893.

Wortmann, T. (2017). Opossum: Introducing and evaluating a model-based optimization tool for grasshopper. Available at: http://papers.cumincad.org/data/works/att/caadria2017_124.pdf.

Wortmann, T. (2019a). Architectural design optimization—results from a user survey. *KnE Social Sciences*, **3**, 473–483. https://doi.org/10.18502/kss.v3i27.5550.

Wortmann, T. (2019b). Genetic evolution vs. function approximation: Benchmarking algorithms for architectural design optimization. *Journal of Computational Design and Engineering*, **6**, 414–428. https://doi.org/10.1016/j.jcde.2018.09.001.

Xia, W.-j., & Wu, Z.-m. (2006). A hybrid particle swarm optimization approach for the job-shop scheduling problem. *The International Journal of Advanced Manufacturing Technology*, **29**, 360–366. https://doi.org/10.1007/s00170-005-2513-4.

Xiaobing, Y., Zhenjie, L., XueJing, W., & Xuming, W. (2021). A hybrid differential evolution and simulated annealing algorithm for global optimization. *Journal of Intelligent & Fuzzy Systems*, **41**, 1375–1391.

Xu, J., & Zhang, J. (2014). Exploration-exploitation tradeoffs in metaheuristics: Survey and analysis. In *Proceedings of the 33rd Chinese Control Conference*. https://doi.org/10.1109/ChiCC.2014.6896450.

Xu, P., Luo, W., Lin, X., Qiao, Y., & Zhu, T. (2019). Hybrid of PSO and CMA-ES for global optimization. In *2019 IEEE Congress on Evolutionary Computation (CEC)* (pp. 27–33). https://doi.org/10.1109/CEC.2019.8789912.

Yang, X.-S. (2009). Firefly algorithms for multimodal optimization. In *Stochastic algorithms: Foundations and applications*. Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-04944-6_14.

Yang, X., & Suash, D. (2009). Cuckoo search via Lévy flights. In *2009 World Congress on Nature & Biologically Inspired Computing (NaBIC)*. https://doi.org/10.1109/NABIC.2009.5393690.

Yi, H. (2016). User-driven automation for optimal thermal-zone layout during space programming phases. *Architectural Science Review*, **59**, 279–306. https://doi.org/10.1080/00038628.2015.1021747.

Yi, H. (2020). Visualized co-simulation of adaptive human behavior and dynamic building performance: an agent-based model (ABM) and artificial intelligence (AI) approach for smart architectural design. *Sustainability*, **12**. 6672–6689 https://doi.org/10.3390/su12166672.

Yi, H., & Yi, Y. K. (2014). Performance based architectural design optimization: Automated 3D space layout using simulated annealing. In *Proceedings of 2014 ASHRAE/IBPSA-USA Building Simulation Conference*(pp. 292–299). https://experts.illinois.edu/en/publications/performance-based-architectural-design-optimization-automated-3d-.

Yi, H., Kim, M.-J., Kim, Y., Kim, S.-S., & Lee, K.-I. (2019). Rapid simulation of optimally responsive façade during schematic design phases: Use of a new hybrid metaheuristic algorithm. *Sustainability*, **11**, 2681. https://doi.org/10.3390/su11092681.

Zhang, J., & Sanderson, A. C. (2009). JADE: Adaptive differential evolution with optional external archive. *IEEE Transactions on Evolutionary Computation*, **13**, 945–958. https://doi.org/10.1109/TEVC.2009.2014613.

Zhang, L., Gao, L., & Li, X. (2013). A hybrid genetic algorithm and tabu search for a multi-objective dynamic job shop scheduling problem. *International Journal of Production Research*, **51**, 3516–3531. https://doi.org/10.1080/00207543.2012.751509.

Zhang, L., Liu, L., Yang, X.-S., & Dai, Y. (2016). A novel hybrid firefly algorithm for global optimization. *PLoS One*, **11**, e0163230. https://doi.org/10.1371/journal.pone.0163230.

Zhang, Y., Yu, Y., Zhang, S., Luo, Y., & Zhang, L. (2019). Ant colony optimization for cuckoo search algorithm for permutation flow shop scheduling problem. *Systems Science & Control Engineering*, **7**, 20–27. https://doi.org/10.1080/21642583.2018.1555063.

Zhu, W. (2011). Massively parallel differential evolution—pattern search optimization with graphics hardware acceleration: an investigation on bound constrained optimization problems. *Journal of Global Optimization*, **50**, 417–437.

Appendix

This appendix presents a new add-on architectural design optimizer developed in this study.

Zebroid

The developed algorithms were implemented in a parametric design interface using Rhino GH. DECS-TAS especially, the most robust and efficient algorithm in this work, was compiled as a GH add-on component entitled "*Zebroid*" for public distribution (Fig. A1). A tutorial video, examples, and a *Zebroid1.0.1.ghpy file* are publicly accessible at https://drive.google.com/drive/folders/15PWok_KYRIlRF3VfgWxjuGvoWnL7UGcv?usp = sharing. The Zebroid optimizer can be installed through the Food4Rhino at https://www.food4rhino.com/en/app/zebroid.

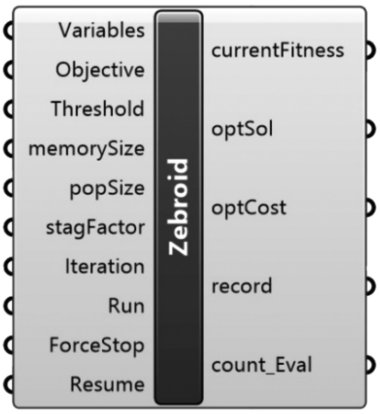


Figure A1: Zebroid: add-on GH multihybrid optimizer.