A selection hyperheuristic guided by Thompson Sampling for numerical optimization

Marcella Scoczynski
UTFPR
Brazil
marcella@utfpr.edu.br

Diego Oliva
Universidad de Guadalajara
Mexico
diego.oliva@cucei.udg.mx

Erick Rodríguez-Esparza
University of Deusto
Spain
erick.rodriguez@deusto.es

Myriam Delgado
UTFPR
Brazil
myriamdelg@utfpr.edu.br

Ricardo Lüders
UTFPR
Brazil
luders@utfpr.edu.br

Mohamed El Yafrani
Aalborg University
Denmark
mey@mp.aau.dk

Luiz Ledo
UTFPR
Brazil
luizledo@utfpr.edu.br

Mohamed Abd Elaziz
Zagazig University
Egypt
abd_el_aziz_m@yahoo.com

Marco Peréz-Cisnero
Universidad de Guadalajara
Mexico
marco.perez@cucei.udg.mx

ABSTRACT

Selection hyper-heuristics have been increasingly and successfully applied to numerical and discrete optimization problems. This paper proposes HHTS, a hyper-heuristic (HH) based on the Thompson Sampling (TS) mechanism to select combinations of low-level heuristics aiming to provide solutions for various continuous single-objective optimization benchmarks. Thompson Sampling is modeled in the present paper as a Beta Bernoulli sampler considering the increase/decrease of diversity among population individuals to measure the success/failure during the search. In the experiments, HHTS (a generic evolutionary algorithm generated by TS) is compared with five well-known evolutionary algorithms. Results indicate that, despite requiring less computational effort, HHTS's performance is similar or better than the other algorithm for most instances and in 50% of the cases it is capable of achieving the global optimum.

KEYWORDS

Hyperheuristics, Thompson Sampling, Evolutionary Algorithms

1 INTRODUCTION

In spite of successfully solving complex and large scale optimization problems, metaheuristics might present some difficulties when applied to real-world problems due to the existence of multiple parameters and high sensibility to their configuration.

Many approaches have been proposed to tackle this issue such as parameter tuning [3, 6, 33] and parameter control [25]. With parameters being defined *a priori* in the first case and during the execution in the second one. Algorithm selection [52] is a mechanism that selects methods from a portfolio and is usually based on machine learning techniques to identify the most suitable algorithm. The combination of metaheuristics in a multiagent perspective [34] is another interesting approach. The hyper-heuristics [31] which generate, select or combine heuristic components to automatically design more efficient heuristic approaches.

Hyper-heuristics are considered search methods that operate on a high-level of abstraction aiming to automate the process of combining, generating and selecting low-level components of a given heuristic [16]. The high-level strategy (HLS) usually involves two main components, (i) the selection mechanism which chooses an element from the pool of heuristics or combine elements from the pool of heuristics components, and (ii) the move acceptance which decides whether to accept or discard the generated solution [49]. The low-level heuristics (LLHs) are defined for a particular optimization framework and problem domain and consist of operators working directly on the problem-solving space. In [31], an ant colony optimization (ACO)-based hyper-heuristic solves the intercell scheduling problem. The approach selects rules to schedule parts and machines in a combinatorial process. In another interesting work [34], the authors propose a MultiAGent Metaheuristic Architecture (MAGMA) framework and show that a hybrid algorithm can be designed as a combination of existing components from the MAGMA architecture.

Although hyper-heuristics are considered robust approaches, the selection mechanism influences quite hard their performance. Particularly, due to the exploration *versus* exploitation (EvE) trade-off: the strategy must find out if there is a heuristic better than the current one (exploration); nevertheless, it must select the best heuristic performance as much as possible (exploitation) [20]. EvE is a classic dilemma in reinforcement learning (RL). Many interesting tasks from decision-making can be formulated as RL problems. In such cases, interactions between agents and the environment, which might be dynamic, stochastic and partial known, guide the

GECCO '21 Companion, July 10–14, 2021, Lille, France

ACM ISBN 978-1-4503-8351-6/21/07…$15.00
https://doi.org/10.1145/3449726.3463140

search for a stock selection approach that optimizes some measure of long-term performance [20].

The stochastic problem of the Multi-Armed Bandit (MAB) is the simplest model of compensation for the EvE dilemma [5] and it is formulated as the problem of a player whose choices rely on different slot machines in a casino (an armed bandit). The MAB model presents a decision-maker who repeatedly chooses between multiple discrete options so that each option produces a probabilistic reward. Over time, the decision maker builds an understanding of the distribution of rewards for each option. The goal is to maximize the expected reward, the option that offers a higher average reward should be sought and then exploited.

Thompson Sampling (TS) is an algorithm based on Bayesian inference that has been proposed to support online decision problems [11]. Its actions are taken in a sequential way aiming at (i) exploiting what is known to maximize current performance and (ii) accumulating new information to improve future performance. Providing an alternative that more intelligently allocates the scanning effort, and proved to be almost optimal [1], TS is adopted in various computationally efficient problems.

This paper proposes a hyper-heuristic (HH) based on TS for choosing one from multiple arms of the MAB problem, i.e., combinations of heuristic components in the HH context. The synergy between HH and MAB has been previously explored for example in [2, 19, 22], when it is used to solve combinatorial optimization problems. Authors in [29] also use TS to support the selection mechanism in a HH, yet in another application context (Max-SAT competition). Besides a different application domain (numerical optimization) is introduced.

In the present paper we consider TS for Bernoulli bandits with Beta(α, β) as *prior* distribution of Bernoulli means. We consider TS as it is more efficient to address exploration and exploitation balance compared with other approaches [51]. Moroever, it has the ability to self-correct [23] due to the Bayesian inference. Moreover, heuristic components are addressed as evolutionary operators: 6 crossover types, 5 mutation operators and 3 selection mechanisms. The experiments are performed using 23 instances of the CEC'05 benchmark functions. In the next section we present the basic concepts necessary to understand the proposed approach.

2 BACKGROUND

2.1 Multi-armed bandit (MAB) problem

The multi-armed bandit problem backs to the sequential design of experiments [39] and can be defined as follows. Given a slot machine, at each time, one of the arms is selected. Each arm, when is selected for playing, returns a random reward based on an unknown probability distribution [1]. In classic MAB formulations, the random rewards obtained from playing an arm repeatedly are independent and identically distributed for each arm. The goal is to maximize the expected accumulated reward in a given time horizon by using a proper selection approach [19].

MAB is often applied to study the exploration/exploitation tradeoff in sequential decision-making problems [1]. According to [7], in such problems, an agent is expected to select the best action (arm) among several alternatives at each time instant. These problems are frequently encountered in various practical applications, from clinical trials, to recommender systems, and anomaly detection.

There are many strategies to decide which arm should be selected at each time. Epsilon Greedy, Upper Confidence Bound (UCB) and Thompson Sampling (detailed in the next section) are examples of algorithms proposed to solve MABs.

2.2 Thompson Sampling

Thompson Sampling (TS) is a randomized probability matching algorithm [1] which samples arms according to the Bayesian posterior probability that each arm is optimal. The beta-Bernoulli bandit problem [40] assumes that rewards are either 0 or 1 drawn from a Bernoulli distribution with parameter $\theta \sim Beta(\alpha, \beta)$ as conjugate prior distribution according to Eq. (1).

$$Beta(\alpha, \beta) : p(\theta; \alpha, \beta) = \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha)\,\Gamma(\beta)} \theta^{\alpha-1}(1-\theta)^{\beta-1} \quad (1)$$

with mean $\alpha / (\alpha+\beta)$ and function gamma Γ. High values of α and β concentrate the distribution around its mean. After a Bernoulli trial observation, the posterior distribution is updated to $Beta(\alpha + 1, \beta)$ or $Beta(\alpha, \beta + 1)$ depending on the success or failure of the trial, respectively.

The TS algorithm starts by assuming that each arm has a prior $Beta(1, 1)$ which is the uniform distribution on $(0, 1)$. After playing arm s at trial t, the result can be either success (reward = 1) or failure (reward = 0) and the total number of successes $S_s(t)$ or failures $F_s(t)$ of the selected arm s is updated accordingly. Thereafter, the algorithm updates the posterior distribution $\theta_s \sim Beta(S_s+1, F_s+1)$, samples θ_i from $Beta(S_i + 1; F_i + 1)$ of each arm $i = 1, ..., K$, and chooses the arm $s \leftarrow \text{argmax}_i(\theta_i)$.

2.3 Related approaches

Many algorithms have been developed for MAB problems, such as Upper Confidence Bound (UCB) [26, 28], Epsilon Greedy [44] and Thompson Sampling [11]. Several studies have shown the efficiency of Thompson sampling [41], and in some cases TS outperforms more complex alternatives such as UCB [11]. TS has been often applied to solve several real-world problems [11, 45]. Recently, in addition to its important contribution to reinforcement learning, MAB has been also applied in the context of optimization. More specifically, in selection mechanisms of hyper-heuristics [2, 19, 22, 29, 30, 36, 50].

Similarly to the present work, in [2] the selection task of a HH is interpreted as a player and the low-level heuristics as the arms of the slot machines. In that work, the authors propose for the first time an adaptive TS mechanism for a selection HH and evaluate its performance on a large number of hyper-heuristics within the 2011 Cross-domain HeuristicSearch (CHeSC) competition [35]. Their proposed approach outperforms even the state-of-the-art algorithms for Personnel Scheduling, Permutation Flow-shop and the Travelling Salesman problem.

In [19], the authors also present a MAB selection strategy. However, differently from other works, the selection mechanism is based not only on the performance of an operator but also on the number of times it has been used. The approach is also evaluated on the CHeSC2011 Competition.

The authors in [22] combine the MOEA/D framework with a selection hyperheuristic to solve many-objective problems. They

propose the Contextual Multi-Armed Bandit (MAB) to find the low level heuristic (Differential Evolution mutation strategy) to apply for each individual solution during MOEA/D execution.

Recently, the work presented in [29] proposes a hyper-heuristic called the Multilevel Synergy Thompson Sampling Hyper-Heuristic that includes both the probabilistic learning mechanism (as a adaptive learning selection hyper-heuristic) and the multilevel paradigm to generate smaller sub-problems from large problem instances hierarchically, addressing Max-SAT competition.

3 THE PROPOSED APPROACH

This section presents the proposed approach encompassing a MAB problem (formulated as the selection of heuristic components in a hyper-heuristic) which must be solved by the TS algorithm used as a selection mechanism of a HH. The TS-based selection mechanism aims at online selecting combinations of operators for crossover, mutation and selection to be used when searching for optimal solutions of numerical optimization problems. Our approach will be referred to as a Hyper-Heuristic based on Thompson Sampling (HHTS) in the remainder of the paper.

3.1 Heuristic components

In this subsection, it is described the portfolio of heuristic components used in HHTS. The crossover is one of the basic operators in evolutionary algorithms. In the proposed approach six different crossover operators are applied to the selected pairs of parents: Single point crossover [38]; Blend crossover [17]; Simulated binary crossover - SBX[15]; Unimodal normal distribution crossover - UNDX [37]; Parent-centric recombination - PCX [15]; and the Opposition-based learning - OBL [48].

Another basic operator is mutation. We consider five mutation operators: Gaussian mutation [24]; Michalewicz' mutation [18]; Differential evolution (DE) mutation [42]; Random mutation [8]; and the Particle swarm optimization operator [27].

The proposed approach considers three selection methods: Roulette Wheel [32]; Tournament [46]; and Truncation [46]. Notice that the operators addressed in this paper are considered due to their adoption in a large number of evolutionary algorithms. In addition, their performance has been studied and analyzed extensively in previous works both theoretically and empirically using benchmark functions and real-world problems.

3.2 HHTS Framework

Within the HHTS context, an arm is equivalent to a particular combination of different LLHs. Considering for example $M = 3$ LLHs, and *CROSS*, *MUT* and *SEL* as acronyms representing crossover, mutation and selection operators respectively, we would have $\mathbf{a} = (a_1, \ldots, a_M)$ encoding an arm, i.e., a particular combination of these operators. In this case, each element $a_m \in \{1, \ldots, L_m, \}$, $m = 1, \ldots, M$, is associated with one specific operator, and L_m denotes the total of options addressed for the particular operator considered in the m-th position of the vector. Therefore, assuming the encoding sequence as crossover, mutation and selection we would have: $L_1 = |CROSS| = 6$, $L_2 = |MUT| = 5$ and $L_3 = |SEL| = 3$, with $|.|$ defined as the sets' cardinalities whose values are established according to the alternatives described in Section 3.1. Figure 1 illustrates the framework which consists of two main modules: high-level and problem domain. The high-level domain is associated with the HHTS components, with $\{\mathbf{a^1}, \cdots, \mathbf{a^K}\}$ identifying the set of all possible arms (i.e., all combinations of the addressed LLHs), and $\mathbf{a^i} = (a_1^i, \ldots, a_M^i)$ encoding a particular combination. The problem domain represents the problem features and parameters (benchmark function).

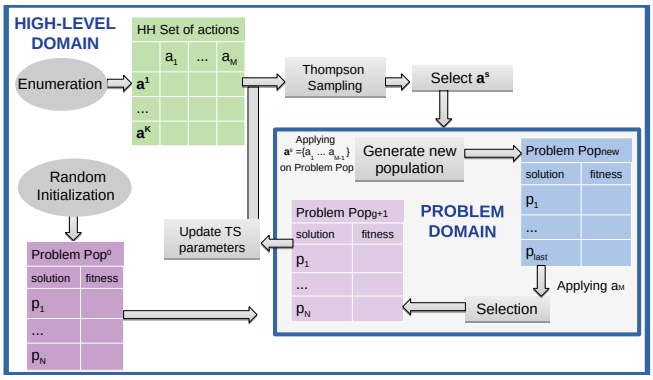


Figure 1: HHTS Framework.

In the high-level domain, the actions (arms) are enumerated in block *Enumeration*. Therefore K represents the total of arms, i.e., all possible combinations using the addressed LLHs (block *HH Set of actions*). In *Thompson Sampling* block, the TS mechanism is considered to identify which arm (combination of LLHs) should be applied at a given iteration.

The first $M-1$ sub-actions $(a_1^s, a_2^s, ..., a_{M-1}^s)$ (i.e., LLHs associated with reproduction operators and encoded in a selected action or arm $\mathbf{a^s}$) are then sequentially applied, in block *Generate new population*, to each problem solution $\mathbf{p_j}$, $j \in 1, ..., N$, from *population Pop*, which has been randomly initialized.

In block *Problem* Pop_{new}, the new problem solutions obtained from the LLHs associated with reproduction operators constitute the new population. It is noticed that Pop_{new} can present a different cardinality at each stage, depending on the applied operator. The last sub-action (a_M^s) encoded in the sequence represents the LLH associated with the selection technique, and it must be applied in block *Selection* to the entire population Pop_{new} to select the best N solutions that will compose the next population Pop^{g+1}.

The entire iterative process continues until a stop criterion is satisfied. Usually used as the stopping criterion, the maximum number of problem function evaluations Max_{eval} is also adopted in this work. Finally, the best problem solution is selected from the population Pop^{g+1} and provided as the algorithm output.

In the proposed approach, TS considers each execution of an arm (a combination of LLHs) as a Bernoulli trial with output $\{0, 1\}$, with 1 meaning that the applied arm generates a better population Pop_{g+1} with respect to a given metric and search point, and 0 denotes the opposite case. The distribution $Beta(\alpha, \beta)$ is commonly used as prior distribution of Bernoulli distribution for the Thompson sampling [47]. Given that $Beta(\alpha, \beta)$ distribution is the prior conjugate of Bernoulli distribution, the framework computes the posteriori distribution of θ_i updated by the count of successes and failures of Bernoulli trials. [40].

The arms perform in a different way according to the regions of the search space [2] and TS chooses an arm according to the Bayesian posterior probability of that arm is optimal [47]. In this paper, Thompson sampling is modeled as a beta-Bernoulli sampler, but it can be generalized beyond the Bernoulli bandit problem [40]. The TS mechanism adopted in the proposed approach is described in Algorithm 1.

Algorithm 1 TS for the Bernoulli bandit in HHTS

1: K: total number of arms or actions (LLH combinations in this case)
2: $Max_{Eval} \leftarrow$ maximum number of function evaluation
3: **for** $i = 1...K$ **do**
4: $S_i \leftarrow 0$
5: $F_i \leftarrow 0$
6: **end for**
7: $g \leftarrow 0$
8: **repeat**
9: **for** each $i = 1...K$ **do**
10: sample $\theta_i \sim Beta(S_i + 1, F_i + 1)$
11: **end for**
12: $s \leftarrow \text{argmax}_i(\theta_i)$
13: Apply arm s (combination of LLHs) to Pop_g
14: $g \leftarrow g + 1$
15: **if** Pop_{g+1} is better than Pop_g **then**
16: $r \leftarrow 1$
17: $S_s \leftarrow S_s + 1$
18: **else**
19: $r \leftarrow 0$
20: $F_s \leftarrow F_s + 1$
21: **end if**
22: **if** Move Acceptance criterion is not met **then**
23: $Pop_{g+1} \leftarrow Pop_g$
24: **end if**
25: **until** Max_{eval}

It initially assumes that each arm i has prior $Beta(1, 1)$ and $S_i = F_i = 0$ for $i = 1, ..., K$. HHTS then iterates by sampling from posterior distributions (line 10), selecting an arm s according to the highest probability (line 12) and playing it (line 13). After playing arm s at iteration g, and observing success ($r = 1$) or failure ($r = 0$) based on which population (Pop_{g+1} or Pop_g) is better to the problem, TS updates the distribution parameters S_s and F_s. Moreover, it accepts the new population in case of success or disregards it in case of failure (line 23).

To check the acceptance criterion (i.e., to test informing whether the modified population is better than the current one), this paper considers the True Diversity normalized metric (D^n_{TD}) [14] which is calculated as using Eq. 2.

$$D^n_{TD} = \frac{\frac{1}{C}\sqrt{\sum_{c=1}^{C}(\overline{p_c^2} - (\overline{p_c})^2)}}{N_{MDF}} \tag{2}$$

D^n_{TD} represents the average standard deviation calculated for each element of a C dimensional solution $\mathbf{p} = (p_1, ..., p_C)$, with $\mathbf{p}$ representing an individual from the population Pop with N individuals, and $\overline{p_c} = \frac{1}{N}\sum_{j=1}^{N} p_{j,c}$, $\overline{p_c^2} = \frac{1}{N}\sum_{j=1}^{N} p_{j,c}^2$. N_{MDF} is a normalization factor that helps to determine the maximum diversity obtained until the current iteration. Such value is computed as the maximum diversity in the initialization procedure.

4 EXPERIMENTS AND RESULTS

This section aims to compare the results of HHTS with those provided by Particle Swarm Optimization (PSO) [27], Differential Evolution (DE) [42], Genetic Algorithm (GA) [21], Differential Search (DS) [12] and Covariance Matrix Adaptation and Evolution Strategies (CMAES) [4] to solve CEC'05 benchmark of continuous functions F1-F23 [43].

We use these functions since they present known global optimum values (f_{min}) that are shown in Table 1 together with equation, type, dimension and lower and upper bounds. In [9], a hyper-heuristic generates mutation operators for evolutionary programming to solve these benchmarks as well. More details about those functions can be found in [43].

The proposed approach (HHTS) has different parameters that must be configured and the values considered in this paper are: the solution size is $M = 3$; the set of crossover operators is $CROSS$ = {Blend, Multi-Point, OBL, PCX, SBX, Single Point, UNDX} and mutation MUT = {Gaussian, Michalewicz, DE, PSO operator, Random}; the selection operators SEL = {Truncation, Tournament, Roulette}; the set of actions (all possible combinations) is $K = (|CROSS| + |MUT|)^{M-1} * |SEL|$ and finally the maximum number of problem function evaluation Max_{eval} is 25000.

In the problem domain, the population size is set to $N = 50$ for all the algorithms considered in comparisons. For the statistical analysis, the optimization methods performed 35 independent runs for each function from the benchmark set. The PSO, GA, DE, DS and CMAES internal parameters have been set up according to the original references aiming to maintain each algorithm performance.

4.1 Comparative Analysis

Figures 2 and 3 show the average best score of objective values over all the 35 runs in the optimization process that is performed until the stop criteria is achieved (maximum number of function evaluations).

We consider one curve for each algorithm according to the colors shown in the legend of Fig. 3l (with HHTS in dark blue). The curves are in linear-log scales but curves for F16 which are in linear scales due to positive and negative values. We can clearly observe that HHTS uses fewer function evaluations to reach the minimum values, providing the best performance in terms of convergence rate.

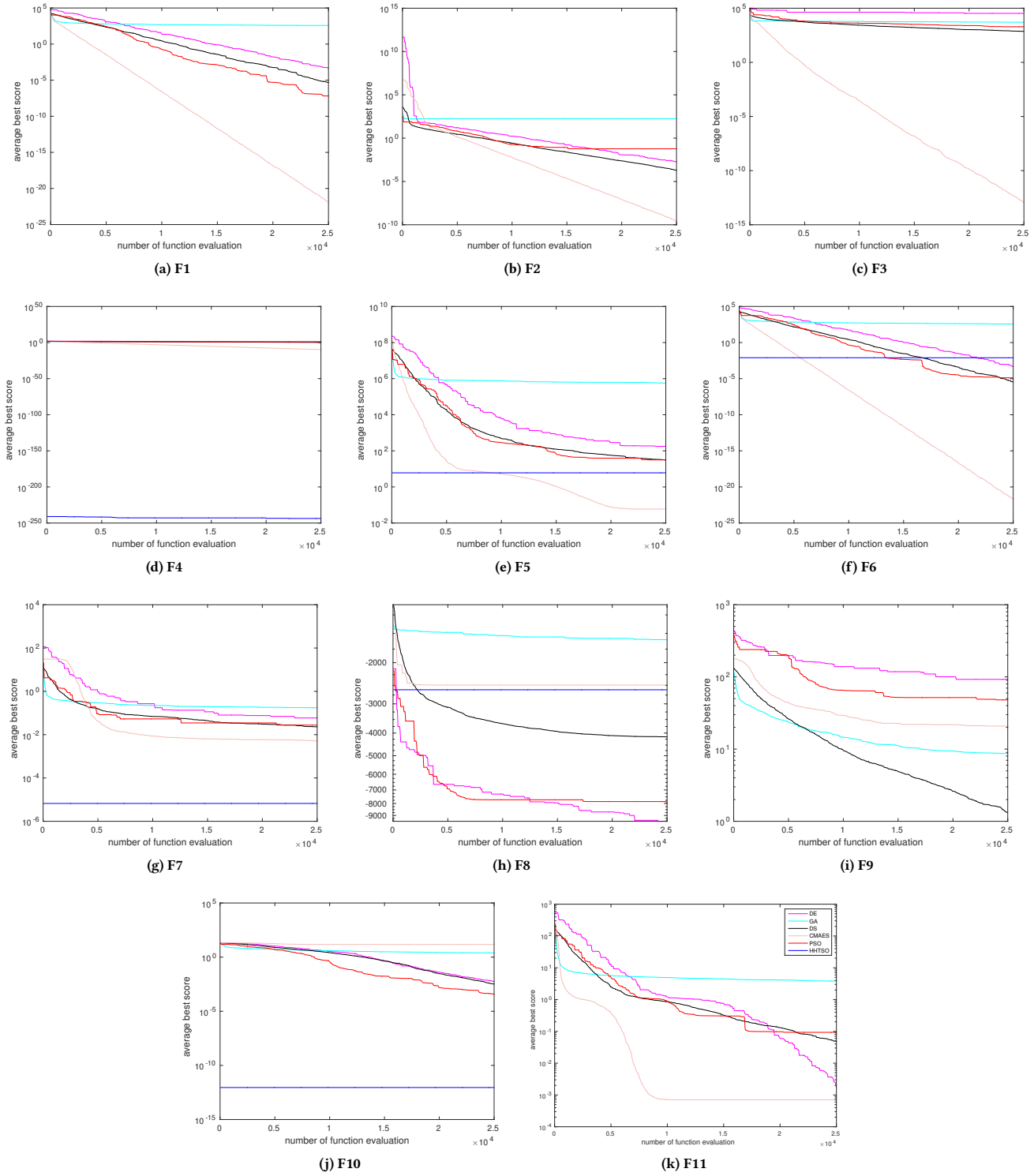
For F1, F2, F3, F9, F11, F14, F16, F19, note that HHTS strongly decays to the minimum values. This fact also occurs when f_{min} is reached with a minimum number of iterations and fitness function evaluations. In these cases, the dark blue curve of HHTS is difficult to identify due to the fast convergence.

Since the information obtained shows a non-normal distribution (based on the Shapiro-Wilk normality test [13]), it is necessary to employ the Kruskal-Wallis [10] and Dunn-Sidak's post-hoc tests. Such methods consider and compare the HHTS and the rest of the methods. In this way they are used the objective function values generated by the 35 independent experiments. Notice that the tests used employs confidence level of 95% ($\alpha = 5\%$). The average of the objective function values for each test function is presented in Table 2. In such table the values in bold corresponds to the global best, while the colored results are the ones that have no statistically significant differences with the global best values.

We notice in Table 2 that HHTS is similar or better than the other methods for all cases but 4 instances (2 unimodal and 2 multimodal). Moreover, in some cases the proposed method has no statistically significant differences in specific when it is compared to DE and

Table 1: Benchmark functions.

ID	Equation	Lower	Upper	Dimension	Type	fmin
F1	$f(x) = \sum_{i=1}^{n} x_i^2$	-100	100	10	Unimodal	0
F2	$f(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	-10	10	10	Unimodal	0
F3	$f(x) = \sum_{i=1}^{n} \left(\sum_{j-1}^{i} x_i \right)^2$	-100	100	10	Unimodal	0
F4	$f(x) = \max_i \{\lvert x_i \rvert, 1 \le i \le n\}$	-100	100	10	Unimodal	0
F5	$f(x) = \sum_{i=1}^{n-1} \left[100\left(x_{i+1} - x_i^2\right)^2 + (x_i - 1)^2 \right]$	-30	30	10	Unimodal	0
F6	$f(x) = \sum_{i=1}^{n} \left([x_i + 0.5] \right)^2$	-100	100	10	Unimodal	0
F7	$f(x) = \sum_{i=1}^{n} i \cdot x_i^4 + random\,[0, 1]$	-1.28	1.28	10	Unimodal	0
F8	$f(x) = \sum_{i=1}^{n} -x_i \sin\left(\sqrt{x_i}\right)$	-500	500	10	Multimodal	-415*5
F9	$f(x) = \sum_{i=1}^{n} \left[x_i^2 - 10\cos(2\pi x_i) + 10 \right]$	-5.12	5.12	10	Multimodal	0
F10	$f(x) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right) + 20 + e$	-32	32	10	Multimodal	0
F11	$f(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	-600	600	10	Multimodal	0
F12	$f(x) = \frac{\pi}{n}\left\{10\sin^2(\pi y_1) + \sum_{i=1}^{n-1}(y_i - 1)^2\left[1 + 10\sin^2(\pi y_{i+1})\right] + (y_n - 1)^2\right\}$ $\sum_{i=1}^{n} u(x_i, 10, 100, 4)$ $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m, & x_i > a \\ 0, & -a \le x_i \le a \\ k(-x_i - a)^m, & x_i < -a \end{cases}$	-50	50	10	Multimodal	0
F13	$f(x) = 0.1\left\{\sin^2(3\pi x_i) + \sum_{i=1}^{n}(x_i - 1)^2\left[1 + \sin^2(3\pi x_i + 1)\right] + (x_n - 1)^2\left[1 + \sin^2(2\pi x_n)\right]\right\} + \sum_{i=1}^{n} u(x_i, 5, 100, 4)$	-50	50	10	Multimodal	0
F14	$f(x) = \left(\frac{1}{500} + \sum_{j=1}^{25}\frac{1}{j + \sum_{i=1}^{2}(x_i - a_{ij})^6}\right) - 1$	-65.536	65.536	2	Multimodal	0
F15	$f(x) = \left(\sum_{i=1}^{11}\left[a_i - \frac{x_1(b_i^2 + b_i x_2)}{b_i^2 + b_i x_3 + x_4}\right]^2\right)$	-5	5	4	Multimodal	0.0003
F16	$f(x) = \left(4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - x_2^2 + 4x_2^4\right)$	-5	5	2	Multimodal	-1.0316
F17	$[f(x) = \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos x_1 + 10]$	-5	5	2	Multimodal	0.398
F18	$[f(x) = \left[1 + (x_1 + x_2 + 1)^2\left(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2\right)\right] \times \left[30 + (2x_1 - 3x_2) \times \left(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 26x_2^2\right)\right]$	-2	2	2	Multimodal	3
F19	$f(x) = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{3} a_{ij}\left(x_j - p_{ij}\right)^2\right)$	1	3	3	Multimodal	-3.86
F20	$f(x) = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{6} a_{ij}\left(x_j - p_{ij}\right)^2\right)$	-0	1	6	Multimodal	-3.32
F21	$f(x) = -\sum_{i=1}^{5}\left[(X - a_i)(X - a_i)^T + c_i\right]^{-1}$	-0	10	4	Multimodal	-10.1532
F22	$f(x) = -\sum_{i=1}^{7}\left[(X - a_i)(X - a_i)^T + c_i\right]^{-1}$	-0	10	4	Multimodal	-10.4028
F23	$f(x) = -\sum_{i=1}^{10}\left[(X - a_i)(X - a_i)^T + c_i\right]^{-1}$	-0	10	4	Multimodal	-10.5363



(a) F1 (b) F2 (c) F3

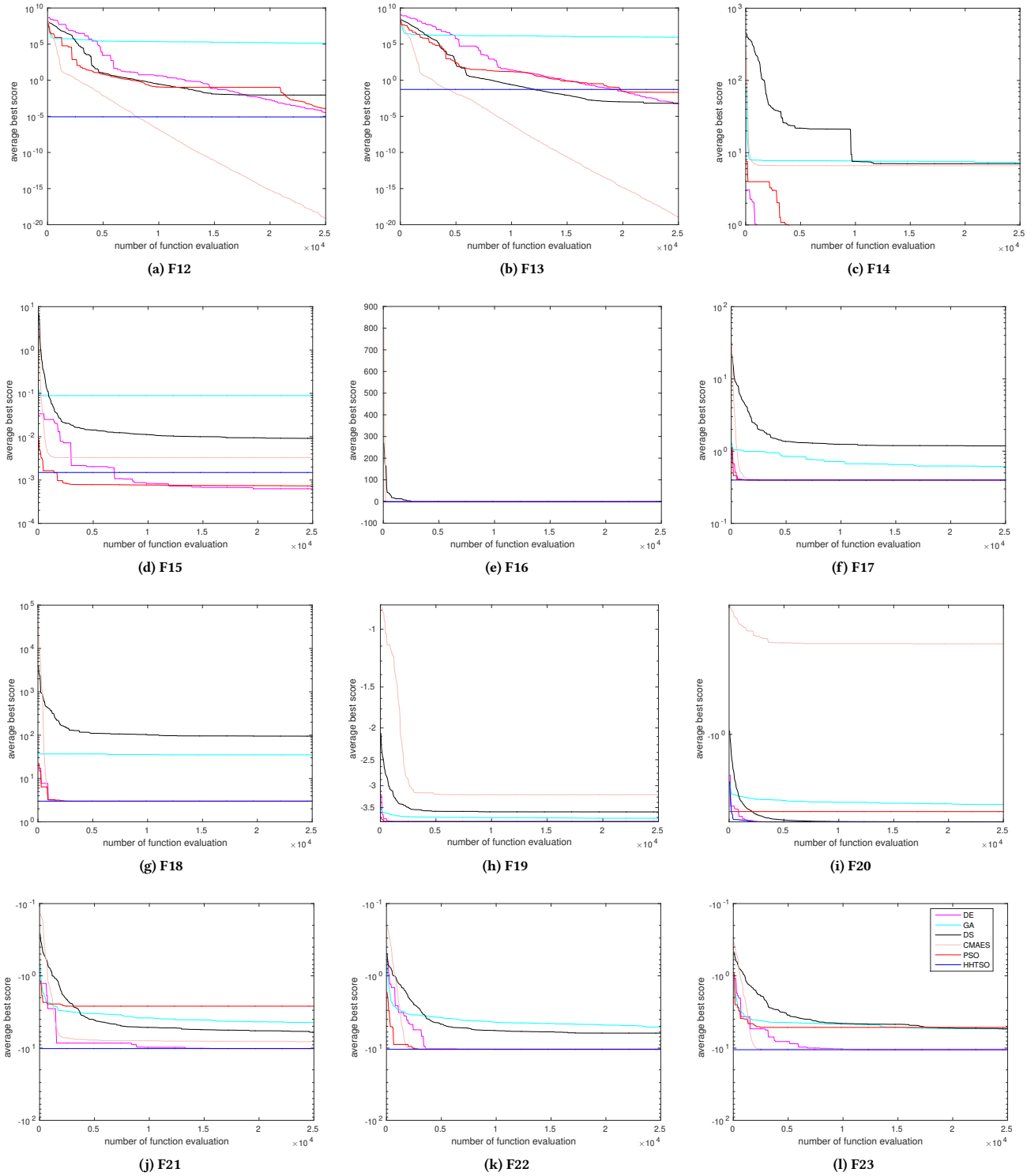
(d) F4 (e) F5 (f) F6

(g) F7 (h) F8 (i) F9

(j) F10 (k) F11

Figure 2: Convergence curve of F1-F11 for average best scores over the 35 runs using semilog scale for all algorithms.

PSO. It is important to remark that for half of the instances, i.e., for F1, F2, F3, F9, F11, F16, F17, F18, F19, F21, F22 and F23, the proposed method can achieve the global optimal f_{min} value.



(a) F12 (b) F13 (c) F14

(d) F15 (e) F16 (f) F17

(g) F18 (h) F19 (i) F20

(j) F21 (k) F22 (l) F23

Figure 3: Convergence curve of F12-F23 for average best scores over the 35 runs using semilog scale (with exception for F16), for all algorithms.

Table 2: Average objective values over 35 executions for each function.

Instance	HHTS	PSO	DE	GA	DS	CMAES
F1	**0.000**	$1.548E-09$	$2.654E-12$	378.0	$4.865E-06$	$1.202E-22$
F2	**0.000**	0.3346	$3.643E-08$	177.2	$0.1986E-03$	$2.783E-10$
F3	**0.000**	1523	$2527E+01$	5086	731.0	$1.107E-13$
F4	**2.134E−244**	7.688	1.962	9.689	1.767	$1.885E-10$
F5	6.065	247.7	47.01	$5718E+02$	30.51	**59.73E−03**
F6	$7.834E-03$	$3.336E-10$	$2.692E-12$	367.6	$3.661E-06$	**1.917E−22**
F7	**0.006668E−03**	$49.54E-03$	$28.24E-03$	$249.5E-03$	$23.15E-03$	$5.160E-03$
F8	−2612	−8112	**−1241E+01**	−1596	−4151	−2376
F9	**0.000**	39.42	58.33	8.668	1.331	20.73
F10	**9.104E−13**	$38.46E-03$	$4.770E-07$	2.420	$3.270E-03$	14.67
F11	**0.000**	$33.92E-03$	$1.626E-09$	3.822	$48.69E-03$	$0.7043E-03$
F12	$0.008258E-03$	$32.60E-03$	$3.992E-13$	$1344E+02$	$8.887E-03$	**8.052E−20**
F13	$0.5282E-03$	$72.64E-03$	$1.628E-12$	$9027E+02$	$0.6518E-03$	**1.200E−19**
F14	**0.9980**	1.252	1.026	7.301	7.042	6.610
F15	**1.494E−03**	$2.383E-03$	$1.289E-03$	$88.73E-03$	$9.132E-03$	$3.310E-03$
F16	**−1.032**	−1.032	−1.032	−0.5766	−0.9.150	−1.032
F17	0.3979	0.3979	0.3979	0.6110	1.182	0.3979
F18	3.000	3.000	3.000	35.03	94.88	3.000
F19	−3.863	−3.863	−3.863	−3.774	−3.862	−3.197
F20	−2.888	−3.269	−3.069	−2.619	−3.316	−0.2885
F21	**−10.15**	−7.078	−9.339	−4.434	−6.091	−8.156
F22	**−10.40**	−9.093	−10.23	−5.207	−6.211	−10.21
F23	**−10.54**	−7.906	−10.32	−5.475	−5.421	−10.52

These results corroborate with the comparison shown in Figures 2 and 3, especially for functions F1, F2, F3, F9, F11, F16, F17, F18, F19, F21, F22 and F23 where f_{min} is achieved.

5 CONCLUSIONS

This article proposes a hyper-heuristic approach based on the Thompson Sampling mechanism to select combinations of low-level-heuristics applied to candidate solutions for different continuous benchmarks. Thompson Sampling is a classical algorithm adopted to solve multi-armed bandit problems, and it is applied in a hyper-heuristic context here, due to its efficiency in addressing the trade-off between exploration and exploitation while selecting appropriate heuristic components.

Modeled in the proposal as a sampler of Beta distributions with means adjusted by Bayesian inference using observations of Bernoulli distributions, Thompson Sampling considers the increase/decrease of diversity among population individuals to measure the success/failure during the search. Diversity has also used as the move acceptance criterion in the proposed hyper-heuristic. The low-level-heuristics have been addressed in the paper as evolutionary operators: 6 different crossover types, 5 mutation operators, and 3 selection mechanisms.

The experimental study was performed over 23 instances of CEC'05 benchmark dataset. The proposed approach HHTS was compared with 5 traditional implementations of Particle Swarm Optimization, Genetic Algorithm, Differential Evolution, Differential Search and Covariance Matrix Adaptation with Evolutionary Strategy. The reported results indicated that HHTS performed similarly or better than the other algorithms in most of instances (19 out of 23 total). HHTS attained the global optimum value for 12 instances. Furthermore, all the results were achieved with less computational effort compared with the other algorithms, which is mainly due to HHTS' ability to balance between exploration and exploitation.

Future works will include extending the approach to use other MAB formulations such as restless and contextual bandits. Moreover, we intend to expand the framework and analyze its performance for combinatorial and multi-objective optimization problems.

REFERENCES

[1] Shipra Agrawal and Navin Goyal. 2012. Analysis of thompson sampling for the multi-armed bandit problem. In *Conference on Learning Theory*. 39–1.

[2] Fawaz Alanazi. 2016. Adaptive Thompson sampling for hyper-heuristics. In *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*. IEEE, 1–8.

[3] Carlos Ansótegui, Yuri Malitsky, Horst Samulowitz, Meinolf Sellmann, and Kevin Tierney. 2015. Model-based genetic algorithms for algorithm configuration. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*.

[4] Anne Auger and Nikolaus Hansen. 2012. Tutorial CMA-ES: evolution strategies and covariance matrix adaptation.. In *GECCO (Companion)*. 827–848.

[5] Allan Axelrod and Girish Chowdhary. 2015. The Explore–Exploit Dilemma in Nonstationary Decision Making under Uncertainty. In *Handling Uncertainty and Networked Structure in Robot Control*. Springer, 29–52.

[6] Thomas Bartz-Beielstein, Oliver Flasch, Patrick Koch, Wolfgang Konen, et al. 2010. SPOT: A toolbox for interactive and automatic tuning in the R environment. In *Proceedings*, Vol. 20. 264–273.

[7] Djallel Bouneffouf and Irina Rish. 2019. A survey on practical applications of multi-armed and contextual bandits. *arXiv preprint arXiv:1904.10040* (2019).

[8] Ilhem BoussaïD, Julien Lepagnot, and Patrick Siarry. 2013. A survey on optimization metaheuristics. *Information sciences* 237 (2013), 82–117.

[9] Edmund K. Burke, M. Hyde, G. Kendall, and J. Woodward. 2010. A Genetic Programming Hyper-Heuristic Approach for Evolving 2-D Strip Packing Heuristics. *IEEE Transactions on Evolutionary Computation* 14, 6 (Dec 2010), 942–958.

[10] G. Casella and R. L. Berger. 2001. *Statistical Inference* (second ed.). Duxbury, USA.

[11] Olivier Chapelle and Lihong Li. 2011. An Empirical Evaluation of Thompson Sampling. In *Advances in Neural Information Processing Systems 24*, J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger (Eds.). 2249–2257.

[12] Pinar Civicioglu. 2012. Transforming geocentric cartesian coordinates to geodetic coordinates by using differential search algorithm. *Computers & Geosciences* 46 (2012), 229–247.

[13] W.J. Conover. 1999. *Practical Nonparametric Statistics* (third ed.). Wiley.

[14] Guillaume Corriveau, Raynald Guilbault, Antoine Tahan, and Robert Sabourin. 2012. Review and study of genotypic diversity measures for real-coded representations. *IEEE transactions on evolutionary computation* 16, 5 (2012), 695–710.

[15] Kalyanmoy Deb, Dhiraj Joshi, and Ashish Anand. 2002. Real-coded evolutionary algorithms with parent-centric recombination. In *Proceedings of the 2002 Congress on Evolutionary Computation. CEC'02 (Cat. No. 02TH8600)*, Vol. 1. IEEE, 61–66.

[16] John H Drake, Ahmed Kheiri, Ender Özcan, and Edmund K Burke. 2020. Recent advances in selection hyper-heuristics. *European Journal of Operational Research* 285, 2 (2020), 405–428.

[17] Larry J Eshelman and J David Schaffer. 1993. Real-coded genetic algorithms and interval-schemata. In *Foundations of genetic algorithms*. Vol. 2. Elsevier, 187–202.

[18] Susana C Esquivel and CA Coello Coello. 2003. On the use of particle swarm optimization with multimodal functions. In *The 2003 Congress on Evolutionary Computation, 2003. CEC'03.*, Vol. 2. IEEE, 1130–1136.

[19] Alexandre Ferreira, Richard Gonçalves, and Aurora Pozo. 2017. A Multi-Armed Bandit selection strategy for Hyper-heuristics. 525–532. https://doi.org/10.1109/CEC.2017.7969356

[20] Alexandre Silvestre Ferreira, Richard Aderbal Gonçalves, and Aurora Pozo. 2017. A Multi-Armed Bandit selection strategy for Hyper-heuristics. In *2017 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 525–532.

[21] David E Goldberg. 2006. *Genetic algorithms*. Pearson Education India.

[22] R. Gonçalves, C. Almeida, R. Lüders, and M. Delgado. 2018. A New Hyper-Heuristic Based on a Contextual Multi-Armed Bandit for Many-Objective Optimization. In *2018 IEEE Congress on Evolutionary Computation (CEC)*. 1–8.

[23] Ole-Christoffer Granmo. 2010. Solving two-armed bernoulli bandit problems using a bayesian learning automaton. *International Journal of Intelligent Computing and Cybernetics* 3, 2 (2010), 207–234.

[24] Natsuki Higashi and Hitoshi Iba. 2003. Particle swarm optimization with Gaussian mutation. In *Proceedings of the 2003 IEEE Swarm Intelligence Symposium. SIS'03 (Cat. No. 03EX706)*. IEEE, 72–79.

[25] G. Karafotias, M. Hoogendoorn, and A. E. Eiben. 2015. Parameter Control in Evolutionary Algorithms: Trends and Challenges. *IEEE Transactions on Evolutionary Computation* 19, 2 (2015), 167–187. https://doi.org/10.1109/TEVC.2014.2308294

[26] Emilie Kaufmann, Olivier Cappé, and Aurélien Garivier. 2012. On Bayesian upper confidence bounds for bandit problems. In *Artificial intelligence and statistics*. 592–600.

[27] James Kennedy. 2011. Particle swarm optimization. In *Encyclopedia of machine learning*. Springer, 760–766.

[28] Tze Leung Lai and Herbert Robbins. 1985. Asymptotically efficient adaptive allocation rules. *Advances in applied mathematics* 6, 1 (1985), 4–22.

[29] Mourad Lassouaoui, Dalila Boughaci, and Belaid Benhamou. 2019. A multilevel synergy Thompson sampling hyper-heuristic for solving Max-SAT. *Intelligent Decision Technologies* Preprint (2019), 1–18.

[30] Mourad Lassouaoui, Dalila Boughaci, and Belaid Benhamou. 2020. A synergy Thompson sampling hyper-heuristic for the feature selection problem. *Computational Intelligence* (2020).

[31] Dongni Li, Miao Li, Xianwen Meng, and Yunna Tian. 2014. A hyperheuristic approach for intercell scheduling with single processing machines and batch processing machines. *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 45, 2 (2014), 315–325.

[32] Adam Lipowski and Dorota Lipowska. 2012. Roulette-wheel selection via stochastic acceptance. *Physica A: Statistical Mechanics and its Applications* 391, 6 (2012), 2193–2196.

[33] Manuel López-Ibáñez, Jérémie Dubois-Lacoste, Leslie Pérez Cáceres, Mauro Birattari, and Thomas Stützle. 2016. The irace package: Iterated racing for automatic algorithm configuration. *Operations Research Perspectives* 3 (2016), 43–58.

[34] Michela Milano and Andrea Roli. 2004. MAGMA: a multiagent architecture for metaheuristics. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 34, 2 (2004), 925–941.

[35] Gabriela Ochoa, Matthew Hyde, Tim Curtois, Jose A Vazquez-Rodriguez, James Walker, Michel Gendreau, Graham Kendall, Barry McCollum, Andrew J Parkes, Sanja Petrovic, et al. 2012. Hyflex: A benchmark framework for cross-domain heuristic search. In *European Conference on Evolutionary Computation in Combinatorial Optimization*. Springer, 136–147.

[36] Diego Oliva and Marcella SR Martins. 2019. A Bayesian based Hyper-Heuristic approach for global optimization. In *2019 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 1766–1773.

[37] I Ono and S Kobayashi. 1999. A real-coded genetic algorithm for function optimization using unimodal normal distribution. In *Proceedings of International Conference on Genetic Algorithms*. 246–253.

[38] Riccardo Poli and William B Langdon. 1998. Schema theory for genetic programming with one-point crossover and point mutation. *Evolutionary Computation* 6, 3 (1998), 231–252.

[39] Herbert Robbins. 1952. Some aspects of the sequential design of experiments. *Bull. Amer. Math. Soc.* 58, 5 (1952), 527–535.

[40] Daniel J Russo, Benjamin Van Roy, Abbas Kazerouni, Ian Osband, Zheng Wen, et al. 2018. A tutorial on thompson sampling. *Foundations and Trends® in Machine Learning* 11, 1 (2018), 1–96.

[41] Steven L Scott. 2010. A modern Bayesian look at the multi-armed bandit. *Applied Stochastic Models in Business and Industry* 26, 6 (2010), 639–658.

[42] Rainer Storn and Kenneth Price. 1997. Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *Journal of global optimization* 11, 4 (1997), 341–359.

[43] Ponnuthurai N Suganthan, Nikolaus Hansen, Jing J Liang, Kalyanmoy Deb, Ying-Ping Chen, Anne Auger, and Santosh Tiwari. 2005. Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization. *KanGAL report* 5 (2005).

[44] Richard S Sutton, Andrew G Barto, et al. 1998. *Introduction to reinforcement learning*. Vol. 2. MIT press Cambridge.

[45] Liang Tang, Romer Rosales, Ajit Singh, and Deepak Agarwal. 2013. Automatic ad format selection via contextual bandits. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*. ACM, 1587–1594.

[46] Dirk Thierens and David Goldberg. 1994. Convergence models of genetic algorithm selection schemes. In *International Conference on Parallel Problem Solving from Nature*. Springer, 119–129.

[47] William R Thompson. 1933. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika* 25, 3/4 (1933), 285–294.

[48] Hamid R Tizhoosh. 2005. Opposition-based learning: a new scheme for machine intelligence. In *Computational intelligence for modelling, control and automation, 2005 and international conference on intelligent agents, web technologies and internet commerce, international conference on*, Vol. 1. IEEE, 695–701.

[49] Ayad Turky, Nasser R Sabar, Simon Dunstall, and Andy Song. 2018. Hyper-heuristic Based Local Search for Combinatorial Optimisation Problems. In *Australasian Joint Conference on Artificial Intelligence*. Springer, 312–317.

[50] Stefan AG van der Stockt and Andries P Engelbrecht. 2018. Analysis of selection hyper-heuristics for population-based meta-heuristics in real-valued dynamic optimization. *Swarm and evolutionary computation* 43 (2018), 127–146.

[51] Yingce Xia, Haifang Li, Tao Qin, Nenghai Yu, and Tie-Yan Liu. 2015. Thompson sampling for budgeted multi-armed bandits. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*.

[52] Lin Xu, Frank Hutter, Holger H Hoos, and Kevin Leyton-Brown. 2008. SATzilla: portfolio-based algorithm selection for SAT. *Journal of artificial intelligence research* 32 (2008), 565–606.