Vol. 36, No. 4 (2026) | DOI: 10.37190/ord/231140

OPEN ACCESS

Operations Research and Decisions

www.ord.pwr.edu.pl

OPERATIONS RESEARCH AND DECISIONS QUARTERLY

Entropy-based Reduction Operator for Heuristic Binary Optimization

Nedjmeddine Kantour*[1] Khadidja Chaabane[1] Sadek Bouroubi[1]

[1]*LIFORCE Laboratory, USTHB, Faculty of Mathematics, Department of Operations Research, P.B. 32 El-alia, 16111, Bab Ezzouar, Algiers, Algeria.*

Corresponding author, email address: nkantour@usthb.dz

Abstract

In this paper, we propose a reduction operator addressing discrete optimization problems, more precisely, for optimization problems with binary represented decisions. This operator may be integrated within various metaheuristics in order to reduce the dimension of the target problem. This is, by performing an iterative supervision over the browsed admissible space throughout the search process. We thus use the information entropy concept to measure the current uncertainty towards decision variables current values provided by an iterative heuristic search algorithm. Hence, it allows to explore the decision space more efficiently. Furthermore, we present two frameworks as effective applications of this operator: a pre-process for a hybrid approach and an interactive heuristic approach. Finally, we use the Knapsack Problem (KP) as a test problem, where we assess the impact of the proposed reduction operator over two classical search algorithms: Local Search (LS) and Genetic Algorithm (GA). More precisely, on their efficiency and the quality of the produced solutions. We further compare it against established variable-fixing approaches from the literature, on both the Knapsack Problem and standard Quadratic Unconstrained Binary Optimization (QUBO) benchmark instances. The obtained results indicate that the proposed reduction compares favorably with these approaches, and that it generalizes across constrained and unconstrained binary optimization problems.

Keywords: *Binary Optimization, Heuristic Algorithms, Search Space Reduction, Information Entropy, Knapsack Problem, Quadratic Unconstrained Binary Optimization.*

1. Introduction

Decision-making problems are often modeled as optimization problems under constraints. Optimization is an expanding field of applied mathematics and computer science, which has asserted its role as an essential decision-making tool since its inception in the 1940s. One category of such problems that attracted many researchers in the last decades is binary optimization, i.e., optimization problems with binary represented decisions. That is due to their wide range of applications in both theory and real-world

Received 19 April 2026, accepted 15 August 2026, published online 15 August 2026
ISSN 2391-6060 (Online)/

applications. We mention for example: Subset-sum problem [21], dominating set [17], protein folding [5], facility location [11], feature selection [46], scheduling [41], cryptanalysis [42], image compression [8], network analysis [27], etc. Binary optimization problems are usually difficult to solve, thus, heuristic approaches are naturally adopted for practical applications. Moreover, these approaches are either problem-specific heuristics that are limited to local optima search, or the so called metaheuristics which are theoretically adaptable to any optimization problem and overcome the limitation of problem-specific heuristics.

A binary optimization problem with n decision variables has an exponential search space of size 2^n, a combinatorial explosion that renders exact approaches intractable for large n [37]. Even metaheuristics with asymptotic convergence guarantees require prohibitively many iterations for large n, making search efficiency a central challenge. While traditional strategies such as intensification and diversification address this challenge at the algorithmic level, a complementary and comparatively less explored strategy is online variable fixing (or dynamic variable elimination): reducing n to $n' < n$ during the search by permanently fixing variables whose values can be anticipated with high certainty. This paper studies an information-theoretic mechanism for such reduction that applies to binary optimization problems and to iterative heuristic search algorithms in general.

1.1. Related work

Information entropy has been employed in the design of metaheuristics in a variety of ways, most commonly as a measure of the uncertainty or diversity present in the search. Within these studies, we distinguish three main roles. The first is diagnostic: the entropy of the population, or of the recently visited solutions, quantifies diversity and helps detect stagnation in local optima, and the resulting measurement is used to trigger diversification. This role appears across several paradigms, including genetic algorithms [13, 20], simplified swarm optimization [29], ant colony optimization [16], evolutionary feature selection [44], and more general bio-inspired diversification schemes [34]. The second role is parameter and operator self-tuning, where an entropy measure is used to adjust algorithmic settings during the search, for example in task scheduling [52] and in entropy-based probabilistic initialization for rule mining [40]. The third role is probabilistic model-building, of which the Cross-Entropy (CE) method [4] can be considered as the main representative: a sampling distribution is updated iteratively by minimizing its Kullback–Leibler divergence to an elite set, with applications to combinatorial [7, 28] and continuous [6, 54] problems, and related relative-entropy criteria have been used within swarm methods [31]. In these three roles, entropy is used to iteratively guide the search rather than to fix variables, so that the dimension of the decision space remains unchanged.

A separate line of research addresses large search spaces directly, by eliminating or fixing variables. Exact and problem-specific pre-processing fixes variables before or during a branch-and-bound search by exploiting the linear structure of the LP relaxation, as in reduced-cost fixing [38] and the tests developed for the knapsack problem [1, 25, 32]. Closer to our setting are interactive variable-fixing strategies that operate inside a heuristic search. The Backbone method [47, 53], developed for the Quadratic Unconstrained Binary Optimization (QUBO) problem, approximates the backbone of an instance, that is the set of variables that take the same value in every optimal solution, by extracting the assignments shared by a pool of recent high-quality solutions and fixing them to intensify the search around promising regions. The sample-persistence variable-fixing (SPVAR) mechanism [22, 23], developed for QUBO models solved by quantum annealers and Monte Carlo samplers, instead tracks the historical selection frequency of each assignment and fixes a variable once that frequency exceeds an empirical threshold, reducing the search space size over successive sampling rounds. Both are effective within their domains, but they decide what to fix only from the structural overlap of elite solutions or from empirical selection frequencies, rather than from a formal measure of how uncertain a variable still is.

In summary, the entropy-based metaheuristics of the first group exploit entropy to steer the search but do not modify the decision space, whereas the variable-fixing methods of the second reduce the decision space on the basis of structural overlap or selection frequency, rather than from a measured reduction of uncertainty. To our knowledge, no existing method uses a per-variable entropy for interactive dimensionality reduction. Furthermore, the distinction from the Cross-Entropy method is particularly relevant: the Cross-Entropy method evaluates entropy at the population level and updates an n-dimensional sampling distribution towards better solutions, so the search keeps operating over all n variables. In contrast, the operator proposed here evaluates the empirical entropy of each individual variable along the search as a measurement of uncertainty, then fixes the variable and removes it from the problem.

We present two applications of this operator. First, a hybrid framework in which the operator preprocesses and shrinks the search space before an exact method is applied; second, an interactive approach in which the operator is embedded within an iterative heuristic search algorithm for real-time search space reduction. This paper is organized as follows: in the rest of this section, we recall some essential concepts of binary optimization, the Knapsack Problem, and heuristic approaches (i.e., classification, comparison, and convergence). In Section 2, we establish the link between heuristic approaches and entropy for uncertainty evaluation. In Section 3, we present the suggested reduction operator and its incorporation framework. In Section 4, we present the experimental study carried out for assessing this operator. Finally, we end the paper with the conclusion and our perspectives.

1.2. Background

An optimization problem consists of finding a solution $x^\star$ in a feasible space $\Omega \subseteq \mathbb{R}^n$ such that $f(x^\star) \geq f(x)$ for all $x \in \Omega$, where f is the objective function. Formally:

$$\begin{cases} \max\ f(x), \\ s.t., \quad x \in \Omega, \end{cases}$$

In this paper, we address binary optimization problems, where $\Omega \subseteq \{0,1\}^n$. As a support problem, we consider the well-known Knapsack Problem (KP), a linear binary optimization problem where all coefficients and right-hand bounds are positive.

1.2.1. Knapsack Problem

Given n items having a characteristic $\forall j \in \{1,\ldots,n\}, w_j \in \mathbb{Z}^+$ (weight, volume, etc.), and their associated profits $\forall j \in \{1,\ldots,n\}, c_j \in \mathbb{Z}^+$, we want to select items as to maximize the profit, while not exceeding the knapsack capacity W. The Knapsack Problem (KP) can be formulated as follows:

$$\begin{cases} max \;\; Z(x) = \sum_{j=1}^{n} c_j x_j, \\ s.t., \;\; \sum_{j=1}^{n} w_j \, x_j \leq W, \\ \qquad\quad x_j \in \{0,1\}, \; \forall j \in \{1,\ldots,n\}, \end{cases}$$

where n is the number of items, x_j denotes the j^{th} binary decision variable, Z stands for the objective function. In the case where the items have multiple characteristics, it is simply called the Multidimensional Knapsack Problem (MKP). The KP is proven to be NP-hard [24]. Several exact methods for solving these problems, we can cite for instance an early application given by Lorie and Savage [30], and the dynamic programming and the branch-and-bound approaches [25, 32]. However, for practical reasons, this class of problem is often tackled using heuristic methods, which favors time efficiency over the guarantee of optimality.

1.2.2. Quadratic Unconstrained Binary Optimization (QUBO)

The Quadratic Unconstrained Binary Optimization (QUBO) problem, also known as the Unconstrained Binary Quadratic Programming (UBQP) problem, is a general binary optimization model that arises in combinatorial optimization, machine learning, and quantum computing. It seeks to maximize a quadratic objective function of binary variables without any constraints. Formally, it can be stated as:

$$\max \;\; Z(x) = \sum_{i=1}^{n} \sum_{j=1}^{n} q_{ij} x_i x_j,$$
$$s.t., \;\; x_i \in \{0,1\}, \; \forall i \in \{1,\ldots,n\},$$

where $Q = (q_{ij})$ is an $n \times n$ matrix of real coefficients, and x is a vector of binary decision variables. The QUBO model is known for its generality: many constrained NP-hard combinatorial optimization problems, such as graph coloring, maximum cut, capital budgeting, and clustering, can be reformulated as QUBO by moving their constraints into the objective function through quadratic penalty terms [26]. In addition, QUBO is the primary input format for quantum annealers and modern quantum-inspired hardware [22]. Solving large-scale QUBO instances is computationally difficult, which makes this problem suitable for assessing the general applicability of heuristic reduction mechanisms such as the proposed E/R operator.

1.2.3. Classification and convergence of heuristic approaches

Heuristic approaches can be divided into two main categories: constructive (greedy) heuristics and ameliorative heuristics based on iterative stochastic transformations. Metaheuristics build on the latter by adopting non-deterministic mechanisms to explore broad solution spaces, and are commonly subdivided into single-solution (trajectory) methods and population-based methods [43, 45]. The No-Free-Lunch theorem [50, 51] states that, averaged over all possible problems, no search algorithm performs better than any other, so that effective performance on a given problem depends on the problem-specific knowledge the algorithm incorporates. Regarding convergence, a metaheuristic is said to converge asymptotically when the best-found solution x^t at iteration t satisfies $\lim_{t\to\infty} f(x^t) = \max_{x\in\Omega} f(x)$. In practice, however, the search often stagnates after a finite number of iterations, owing to limitations of its operators, parameters, or problem-specific adaptations. We interpret this practical behaviour as a form of partial convergence: along the search, a subset of the decision variables progressively settles at the values they take in high-quality solutions. In the following section, we present a means of exploiting this partial convergence by anticipating the stagnation values of individual decision variables.

2. Uncertainty evaluation within Heuristic Algorithms

When considering a solution's elementary component values (i.e., a decision variable) within the best-found solution or across a generation in a population-based metaheuristics, it functions as information sources, forming an information stream along the iterative search process. As the search procedure progresses towards a solution, ideally approaching optimality, a particular value within the information stream or the decision variable emerges as significantly prevalent. Consequently, this decision variable tends to stagnate around this dominant value, indicating a trend towards convergence. Hence, the application of certainty measures to evaluate the information stream generated by the guided search process facilitates the anticipation of convergence for this component. In the rest of this section, we explain how one can harvest information from an iterative heuristic search algorithm and how one can exploit it to predict components values.

2.1. Entropy of a binary source

The entropy is stated as follows [15]:

$$\sum_{x\in A} -\mathbb{P}(x)\log\mathbb{P}(x).$$

Take the case of a binary source (A comprising the two symbols 0 and 1), such that:

$$\mathbb{P}(0) = p \text{ and } \mathbb{P}(1) = 1 - p.$$

The entropy of a binary source is:

$$-p\log_2(p) - (1-p)\log_2(1-p).$$

These formulations express the measure of uncertainty or randomness within the binary source, providing insights into the distribution and predictability of symbols within the source.

2.2. Uncertainty within heuristic convergence

If we project the entropy concept on the distribution of each binary decision variable throughout the iterative process of a heuristic search algorithm, it would allow us to evaluate the uncertainty towards a given variable value. This is, by quantifying its uncertainty using the entropy. Let $x = (x_1, x_2, \ldots, x_n), x_i \in$

$\{0, 1\}$ decision variables of a binary optimization problem. For the entropy calculation, we consider for each decision variable x_i as a source of information, where the values taken by x_i in the iterative process of a heuristic search algorithm are considered as an information stream and used for entropy calculation. The entropy of the i^{th} variable $H_i = H(x_i)$ is defined as follows:

$$H(x_i) = -\mathbb{P}(x_i = 1)\log_2(\mathbb{P}(x_i = 1)) - \mathbb{P}(x_i = 0)\log_2(\mathbb{P}(x_i = 0)).$$

Hence, considering the fact that heuristic search gradually enhances the quality of the produced solution, we use the information entropy using this binary information stream to measure the uncertainty within the search process towards a component value. In Figure 1, we present an illustrative example of the evolution of the entropy values of two decision variables using their values in the best found solution along an iterative search process of a heuristic algorithm.

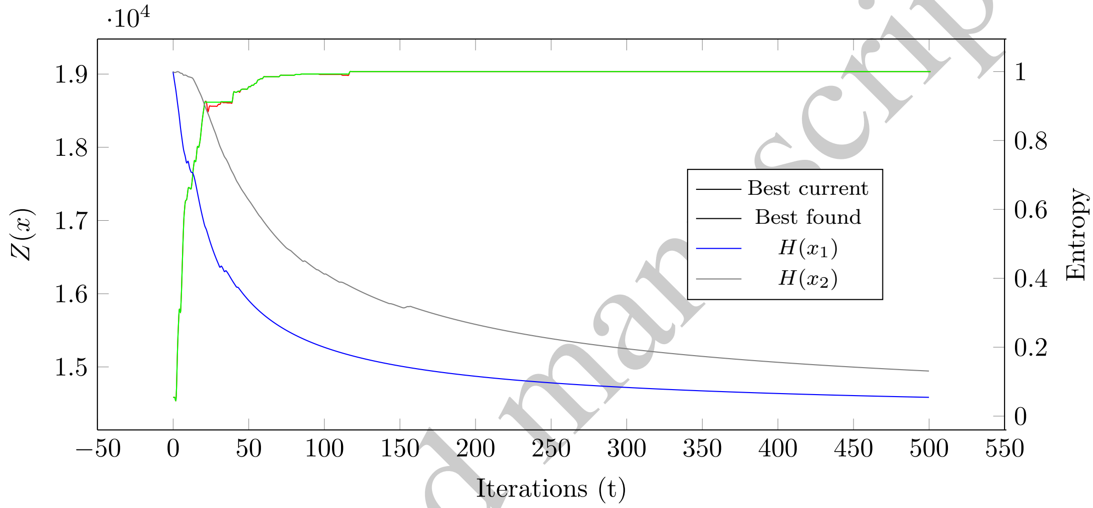

Figure 1. Illustrative example of the entropy evolution, as an uncertainty measure, along an iterative convergence search process.

One can observe a correlation between the entropy values and the practical convergence of the search algorithm. This can be interpreted as follows: for a given decision variable, say x_1, the uncertainty towards its current value (i.e., $H(x_1)$) is declining along the search process. Hence, in theory, one can prematurely fix the value of x_1 and precedes the search process without degrading the targeted value.

3. Entropy based reduction operator

As it is mentioned earlier, one can assess the uncertainty amid an iterative search algorithm towards solutions values using information entropy, we suggest an entropy based reduction operator, we denote it by E/R operator, that can be used as a pre-process for solvers and/or incorporated within iterative heuristic search algorithm. This latter is essentially based on:

- periodic calculation of the entropy at the current generation,
- fix the values of variables that reach a given uncertainty threshold α (i.e., a sufficiently low entropy).

For a given parameter α the reduction operator proceed as follow: for each decision variable $x_i, i = \overline{1, n}$, if the entropy reaches the predefined uncertainty threshold α, i.e., if $H(x_i) \leq \alpha$, then

$$x_i = \begin{cases} 1, & \text{if } \mathbb{P}(x_i = 1) > \dfrac{1}{2}, \\ 0, & \text{otherwise.} \end{cases}$$

The uncertainty threshold α is the entropy value below which the uncertainty about a decision variable is considered low enough to fix its value.

The uncertainty threshold α has a direct probabilistic reading. Let $p = \mathbb{P}(x_i = 1)$ and $h(p) = -p\log_2 p - (1-p)\log_2(1-p)$ for the binary entropy function, the fixing condition $H(x_i) \leq \alpha$ requires the frequency of the majority value to reach at least $h^{-1}(\alpha)$ (see, Figure 2); α thus expresses the confidence demanded before a variable is fixed. Furthermore, the function h is symmetric about $p = 0.5$, where uncertainty is maximal, and becomes zero at the two certain values $p = 0$ and $p = 1$, where its slope diverges. This steepness near the extremes is what makes the criterion meaningful for a heuristic search: as a decision variable converges towards one value, its observed entropy collapses sharply, so a small α fixes a variable only once the search has committed strongly to that value, whereas a larger α fixes it earlier, on weaker frequency. Furthermore, near the center (i.e., $p = 0.5$) where the slope is relatively flat, small changes in entropy values require large changes in the corresponding frequency. In practice, $H(x_i)$ is estimated from the frequencies observed over a finite number of iterations. The choice of α and of the window length is studied in the experimental analysis (Sections 4 and 4.3.3).

3.1. Illustrative example

Let $\alpha = 0.2$ be the uncertainty threshold. Any value of $\mathbb{P}(x_i = 1)$ that maps to an entropy value smaller than α triggers the reduction operator. Figure 2 presents an illustrative example of the range of $x_i = 1$ appearance frequencies that corresponds to the uncertainty threshold $\alpha = 0.2$. Hence, this threshold is attained when the appearance frequency of $x_i = 1$ is within the corresponding ranges $[0, 0.03]$ and $[0.97, 1]$, highlighted in red.

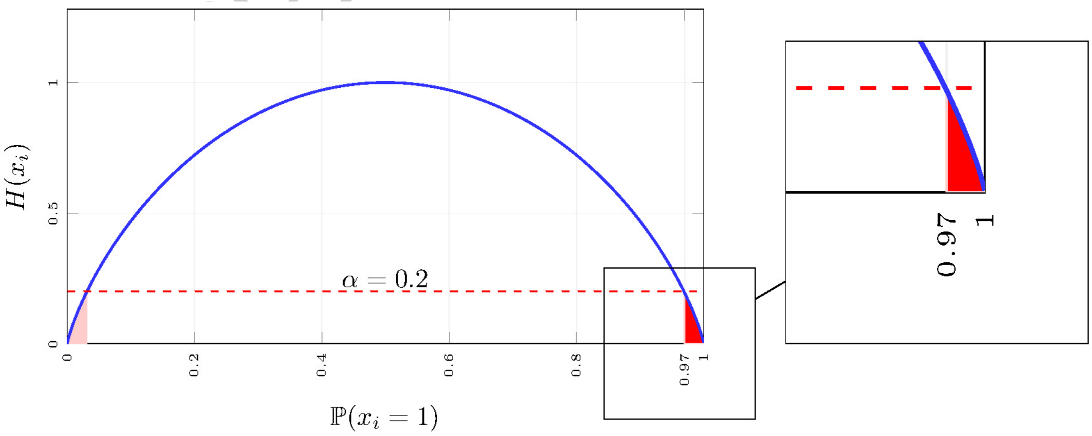

Figure 2. Illustrative example of the uncertainty threshold choice in a binary entropy function.

In Figure 3, we project this uncertainty threshold choice on the search process of a heuristic algorithm. This figure illustrates the same run given in Figure 1. As it is mentioned above, entropy values are obtained by tracking the values of two decision variables in the best found solution. In this particular search evolution, it is pointed that according this fixed value of $\alpha = 0.2$, the variables x_1 and x_2 attain the uncertainty threshold and can be fixed starting at iterations 99 and 292 respectively (i.e., $\mathbb{P}(x_1 = 1), \mathbb{P}(x_2 = 1) \in [0, 0.03] \cup [0.97, 1]$). In other words:

$$x_i = \begin{cases} 1, & \text{if } \mathbb{P}(x_i = 1) \geq 0.97, \\ 0, & \text{if } \mathbb{P}(x_i = 1) \leq 0.02. \end{cases}$$

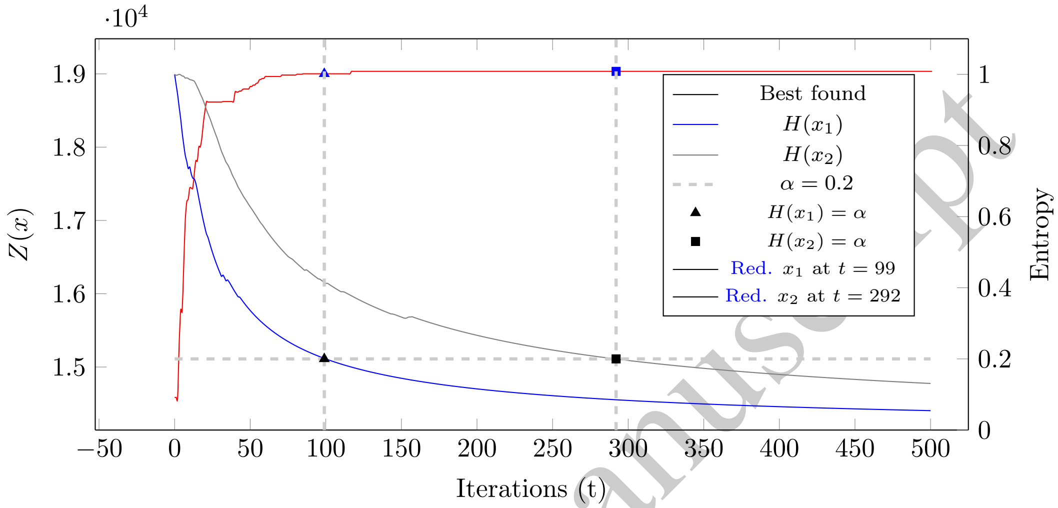

Figure 3. Illustrative example of the link between the uncertainty threshold and an iterative search process using random instance with 500 variables.

3.2. Hybrid framework: E/R as Pre-process

This approach can be used to design hybrid methods, where, one can incorporate this reduction operator as a heuristic pre-processing phase to reduce the search space and then address it using an exact method (see Figure 4). In particular, this approach exploits this operator only to reduce the search space. Consequently, it relies on its capacity to offer an acceptable compromise between its accuracy and its reduction rate. These performance indicators are measured in Section 4.2.

On the other hand, we present a general framework to incorporate this reduction operator into heuristic search, as to enhance the efficiency and the quality of solution provided by classical iterative heuristic search algorithms.

3.3. Interactive framework: E/R incorporation amid Heuristic Algorithms

In the rest of this section, we present the adopted approach for integrating the entropy-based reduction operator within the search process of a heuristic algorithm. As to efficiently exploit this operator, we suggest to restructure the search process into a repetitive three step process: (1) heuristic search, (2) uncertainty assessment, (3) search space reduction. Where, in the first step, one can use any iterative

[1] Here, the admissible space dimension is $n - |R_0 \cup R_1|$.

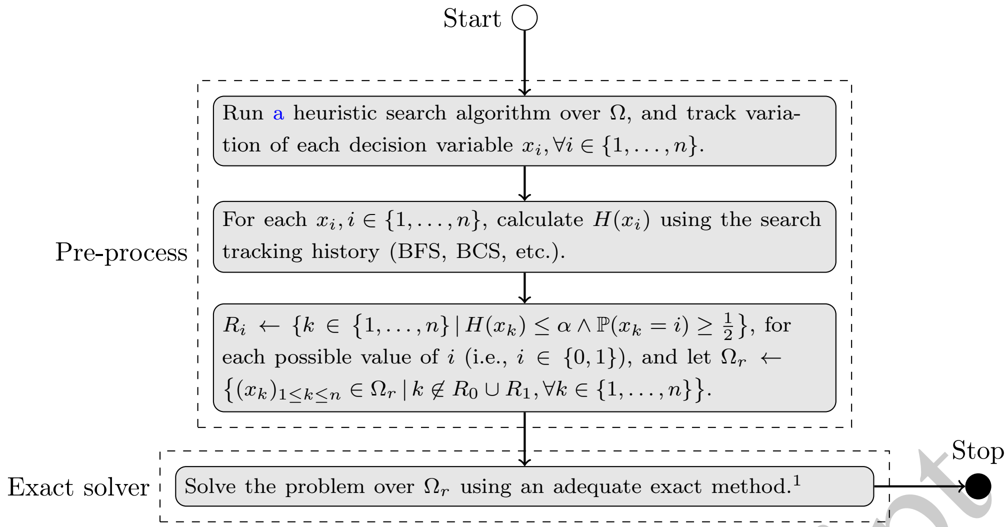

Figure 4. Schematic overview of incorporating the E/R reduction operator as a pre-process step of a hybrid approach.

heuristic search algorithm with a periodic tracking of elementary solution component values. The second step is triggered periodically (i.e., at each iteration at most), where, we assess the uncertainty towards elementary solution component values using the entropy of their corresponding information stream. In the final step, we reduce the search space by fixing the values of each solution component for which uncertainty has reached the predefined uncertainty threshold α.

Let $\Omega \subseteq \{0, 1\}^n$ be the initial search space, $R_i, i \in \{0, 1\}$ are the sets of reduced variables (fixed at i), and Ω_r the current reduced search space obtained along the search process. Figure 5 summarizes the incorporation of the entropy based reduction operator within a heuristic search algorithm.

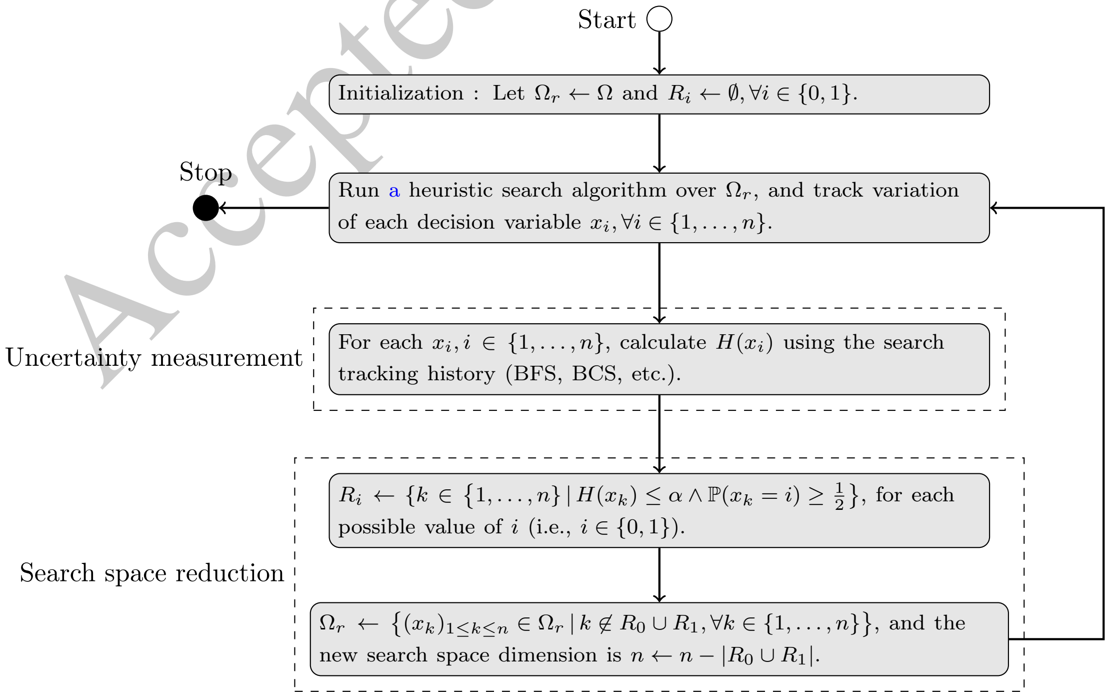

Figure 5. Schematic overview of incorporating the E/R reduction operator into heuristic algorithms.

Several design choices must be made within this framework, mainly the uncertainty measurement period and the specific choice of the information stream. In the case of a single-solution based approaches, one can use the best found solution or the current solution, obtained after each admissible transformation, as an information stream. Furthermore, in the case of a population-based approaches, one can either update the uncertainty using the entropy of the current population, the best found solution, or the current best solution (i.e., the best within the current population). Considering the best found solution as an information source, this approach often lead to a premature reduction, since the uncertainty evaluations drops rapidly. Which makes the algorithm highly sensitive to the uncertainty threshold variations. On the other hand, the use of the current solution only provides information about the current neighborhood. Thus, the uncertainty is measured only for the characteristics of solutions in the current neighborhood, which may not contain a near-optimal solution. Accordingly, one must achieve a compromising approach between the accuracy of the reduced space to do not alter the target solution (i.e., ideally to achieve optimality), and the effectiveness of this operator, by reducing the search space as much as possible according to the browsed solutions, and enhance the quality of the produced solution.

A further property shared by both frameworks is that variable fixing through the E/R operator is irreversible: once a variable is fixed, it is no longer reconsidered in subsequent iterations. This guarantees a monotonic reduction of the search space, but it also entails a risk of premature fixing. However, in a deceptive landscape, a variable may stabilize early at a suboptimal value, its empirical entropy may then drop, and the operator may fix it within a suboptimal region.

4. Experimental Study

In this section, we evaluate the efficiency, accuracy, and generality of the proposed E/R operator. The analysis proceeds in two phases, from constrained to unconstrained optimization landscapes. First, we study the behavior of the operator on the constrained Knapsack Problem. We begin by outlining the experimental methodology and parameter settings. In Section 4.2, we assess the capacity of the E/R operator as an offline pre-processing step to reduce the problem dimension while preserving the optimal objective value, using exact resolution. We then investigate the interactive incorporation of the E/R operator within two classical metaheuristics: Local Search (LS) and Genetic Algorithm (GA). We also benchmark the suggested E/R approach against the state-of-the-art variable-fixing approaches on the KP instances. Finally, Section 4.5 evaluates the E/R operator against state-of-the-art methods on Quadratic Unconstrained Binary Optimization benchmark suites, to assess its generality.

4.1. Experimental Methodology

The experimental study is divided into two domains. The first domain focuses on the constrained 0-1 Knapsack Problem (KP) to analyze the operator's fundamental dynamics, where we evaluate two primary operational approaches: (1) pre-process reduction, in which we assess the suggested operator as an offline pre-processing step that only aims to reduce the search space prior to exact resolution; and (2) interactive reduction, where we assess the suggested incorporation framework (see Section 3.3) using two classical heuristic approaches: GA and LS. We have chosen these two algorithms to study the incorporation into the principal heuristic design categories: an ameliorative single-solution heuristic (LS) and a population-based metaheuristic approach (GA). We then benchmark the E/R operator against state-of-the-art dynamic variable-fixing approaches. This benchmarking begins within the KP domain and then moves to the second domain, the QUBO problem. This progression from constrained (KP) to unconstrained (QUBO) environments tests both the structural efficiency and the generality of the proposed operator. Within the QUBO variable-fixing literature, Wang et al. [47] proposed TS-Backbone, which identifies and fixes variables that are consistent across a pool of high-quality local optima; and Karimi and Rosenberg [22] proposed the SPVAR mechanism, which fixes variables whose empirical selection frequency exceeds a persistence threshold over repeated sampling rounds.

For the KP experiments, the operator has been tested on random instances of the Knapsack Problem ranging from 50 to 1000 variables, with costs $c_j \in \mathcal{U}(1, 1000)$ and weights $w_j \in \mathcal{U}(1, 1000)$ for all $j \in \{1, \dots, n\}$, and a capacity $W \in \mathcal{U}\left(\frac{1}{3}\sum_{j=1}^{n} w_j, \frac{2}{3}\sum_{j=1}^{n} w_j\right)$.

On the other hand, the three benchmark suites used to evaluate QUBO heuristics originate from dedicated studies. Beasley [2] introduced the OR-Library QUBO test set together with simulated annealing and tabu search heuristics. We use the 40 instances with $n \leq 500$. Optimality for $n \leq 250$ has been established by exact methods (i.e., the semi-definite branch-and-bound of Rendl, Rinaldi and Wiegele [39]), while the iterated tabu search of Palubeckis [35, 36] provides the best-known reference values for $n = 500$. Glover, Kochenberger and Alidaee [14] proposed 45 instances of varying density, called the GKA suite, alongside tabu search and GRASP heuristics that established the primary BKV reference for this set. Billionnet and Elloumi [3] introduced 80 instances with $n \leq 250$, called the BE suit, and provided proven optima for $n \leq 150$. A comprehensive repository of certified optima and best-known bounds for all three suites is maintained in the Biq Mac Library [48].

The GA and LS implementation and setting details are summarized in Table 1. Note that, we have chosen to combine the mutation operator and the random correction procedure as the feasible transformation $\mathcal{T}$ required for the LS. For each instance size, a single random instance is generated, the 15 reported runs correspond to independent executions of the algorithm on that same instance. Furthermore, in the case of LS, the E/R is triggered after a predefined period, here we chose 50 iterations. The triggering period serves as a minimum sample size for the entropy estimator: a sufficient number of observations is required for the empirical frequency $\hat{P}(x_i = 1)$ to reliably approximate its true probability. A period that is too short yields high-variance entropy estimates, potentially leading to premature and erroneous variable fixing. The choice of 50 iterations is conservative for the tested instance sizes ($n \in [50, 1000]$), ensuring that each variable has been observed enough times for the entropy estimate to stabilize. That is, to provide adequate initialization for $H(x_i)$, and to prevent premature reduction (see, Algorithm 1).

On the other hand, the experiments were carried out on a personal computer equipped with an Intel® Core™ i7-1270P CPU, 2.20GHz with 32GBs of RAM. All of the implementations were achieved using JAVA SE platform and IBM ILOG CPLEX Optimization Studio 22.1.1.0.

4.2. E/R as a pre-process

Considering the first approach, where, we use the E/R operator as a pre-process, in which, we measure the uncertainty using all the visited solutions throughout the search process, and than deduce the final reduced space according to a predefined uncertainty threshold. Here, we address the main criteria for assessing the suggested operator's performance are: the reduction rate, the accuracy of the reduction. A pre-process

Output: The best found solution x
Generate an initial random feasible solution x;
repeat
  while $\neg$*(Reduction period met)* **do**
    Apply transformation $x' \leftarrow \mathcal{T}(x)$, with $\mathcal{T} : \Omega \rightarrow \Omega$;
    if $f(x) \leq f(x')$ **then**
      $x \leftarrow x'$;
    end
    Track changes in x;
  end
  Invoke E/R operator;
until *(Termination condition met)*;
return x;

Algorithm 1. General Framework for Local Search with E/R operator.

Table 1. Summary of experimental parameters for GA and LS.

Parameter	GA	LS
Population size	100	N/A
Selection method	Roulette wheel	N/A
Crossover type / rate	Uniform / 0.7	N/A
Mutation rate	0.01	N/A
Feasible transformation $\mathcal{T}$	N/A	Mutation + correction
Termination criterion	Max. iterations	Max. iterations
Maximum iterations	500	500
Number of runs per instance	15	15
E/R triggering period	Every generation†	50 iterations
E/R information stream	Best-found solution	Current solution
α (uncertainty threshold)	$\sim \mathcal{U}[0.1, 0.3]$	$\sim \mathcal{U}[0.1, 0.3]$

† For GA, entropy is computed cumulatively over all best-found solutions seen since the start of the run, not just within the current generation.

is usually used before solving the problem using an exact method. Hence, a high reduction rate decreases the run time for solving the original problem. In contrast, it is clear that the obtained problem after performing the reduction has to maintain the same optimal evaluation as the original problem. In other words, the reduction operator has to be accurate and, ideally, has no impact on the quality of the optimal solution.

4.2.1. Reduction rate and its accuracy

Table 2 summarizes the obtained results regarding the reduction rate. This latter can be quantified and assessed using: (1) the number of reduced variables, i.e., $|R_0|$ and $|R_1|$, and (2) the accuracy rate for R_0 and R_1. Furthermore, we report the reduction rate representing the reduced portion of the search space. Note that, for this experiment, the uncertainty threshold is chosen randomly within the interval $[0.2, 0.4]$.

Size	Mean redu. variables		Mean accuracy		Mean redu. rate (%)
	$\|R_0\|$	$\|R_1\|$	0	1	
50	18.4	12.0	0.99	1	60.8
100	12.8	64.0	0.79	1	76.8
250	92.6	76.6	0.97	0.88	67.7
500	174	77.2	0.93	0.91	50.2
750	143.8	284.4	0.96	0.944	57.09
1000	127.8	363.8	0.96	0.95	49.16

Table 2. Experimental results regarding the reduction rate and its accuracy of the E/R pre-process.

4.2.2. Impact on solution's quality

In order to assess the impact of the reduction operator on the quality of the solution, we calculate the mean deviation from the optimal solution. This value represents the mean absolute error. The deviation is calculated as follows:

$$\text{Deviation} = \frac{|Z^\star - Z(\hat{x})|}{Z^\star},$$

where, $Z^\star$ is the evaluation of an optimal solution, and $\hat{x}$ is the obtained solution. Table 3 reports the resulting mean deviation for each instance size.

Instance size	50	100	250	500	750	1000
Mean deviation	1.11E-16	4.13E-3	1.42E-3	3.04E-3	6.25E-3	8.46E-3

Table 3. Experimental results regarding the deviation from the target optimal evaluation (pre-process approach).

4.2.3. Discussion

As it is shown in Table 2, the suggested reduction operator offers a mean reduction rate within 49% to 76% with a rather stable accuracy within 91% to 100%. Furthermore, one can observe that the mean reduction rate and its accuracy do not depend on the size of the search space. On the other hand, the results presented in Table 3 show that solving the reduced problem still provides a good approximate solution for the original problem. The results also confirm that the reduction operator does not alter the target value.

4.3. Interactive approach

In this part, we present the experimental results obtained for the integration of the E/R operator within heuristic algorithms with real-time reduction throughout the search. The integration framework is presented in Section 3.3. Here, we mainly assess the impact of the suggested E/R operator on the quality of the produced solution using the deviation metric, and the efficiency enhancement of the host heuristic algorithm using the speedup metric. Figures 6 and 7 show examples of the mean current deviation and speedup of five runs of one random instance for each instance size. We assess these metrics for both LS and GA against their respective enhanced approach incorporating the E/R operator. The instant speedup is defined as the ratio between the current deviations of the original heuristic and the one incorporating the E/R operator. Note that, in Figure 7 with instances of sizes 100 and 200, the mean instant deviation of the suggested approach has reached zero (i.e., division by zero).

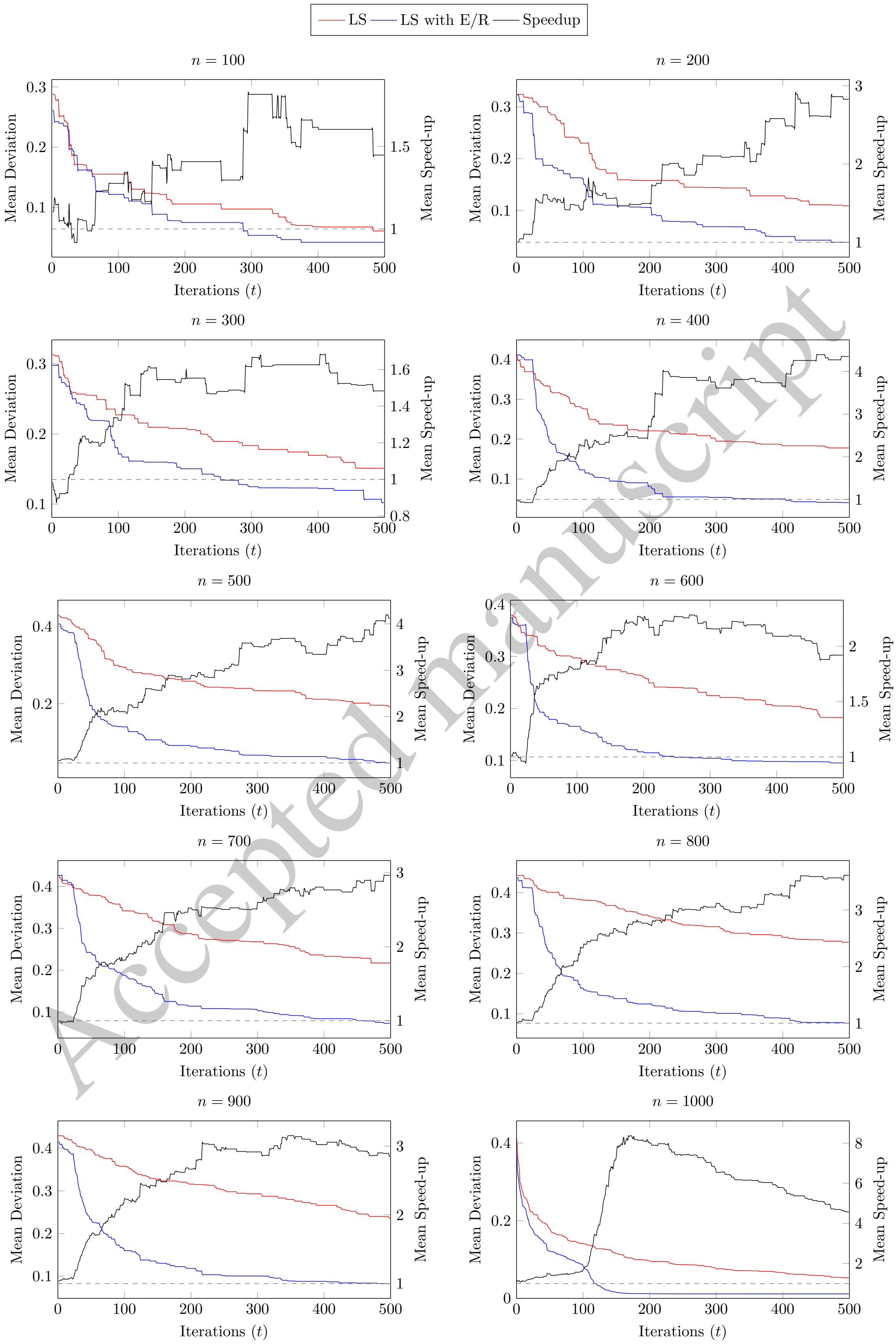

Figure 6. Illustrative examples of the mean instant deviation and speed-up ratio obtained for the LS against the LS with E/R operator.

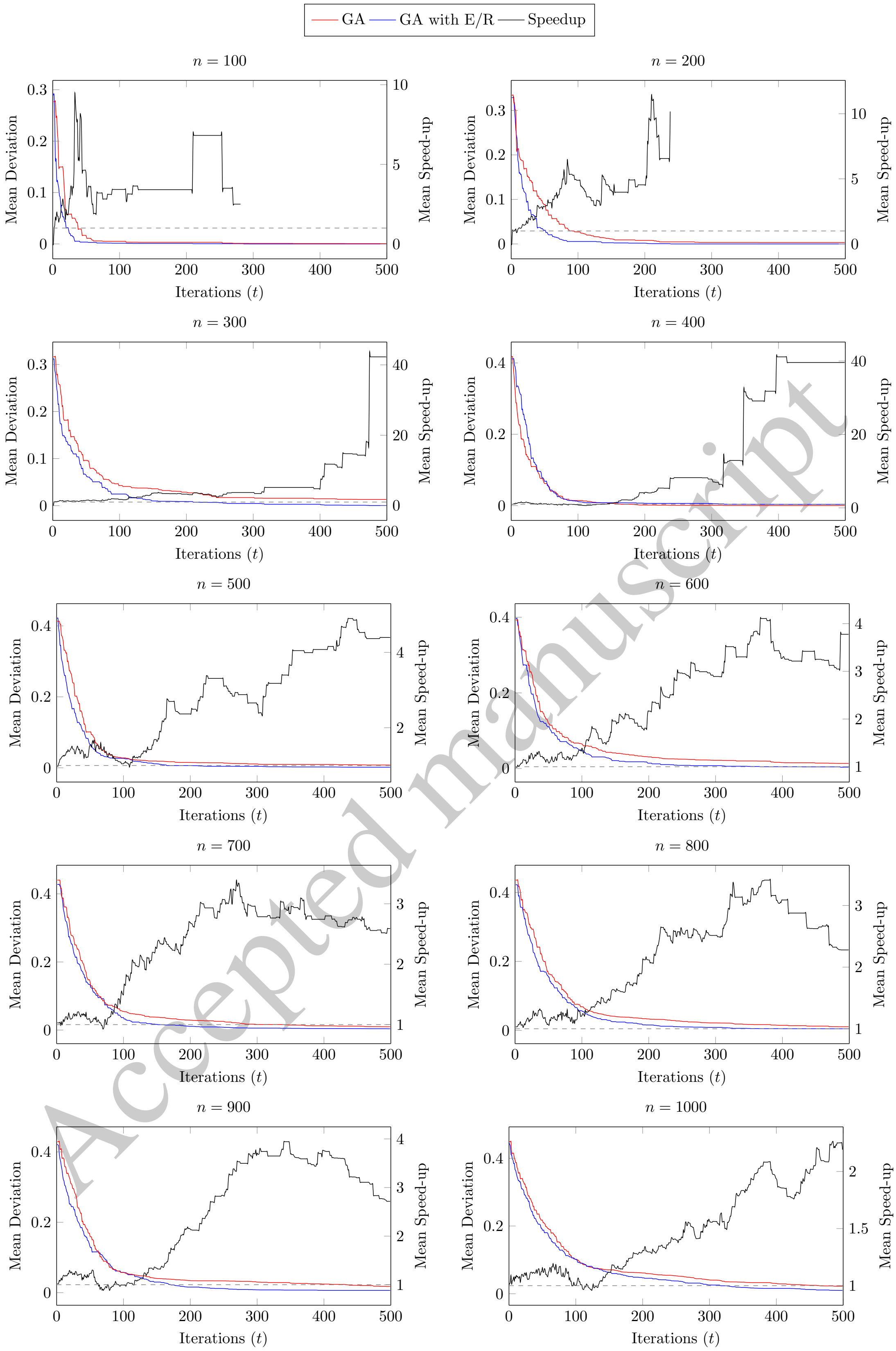

Figure 7. Illustrative examples of the mean instant deviation and speed-up ratio obtained for the GA against the GA with E/R operator.

One can observe in Figures 6 and 7 that in both incorporations (i.e., within LS and GA), the current deviation decreasing speed (i.e., convergence speed) is higher within both heuristic algorithms incorporating the E/R operator, when compared with the corresponding classical heuristic algorithm. This is confirmed with the increasing tendency of the speedup in both cases. Furthermore, one can also observe that current speedup value is predominantly greater than one, which indicates the advantage of the suggested approach. Moreover, one can also observe in Figure 6 an abrupt change in the convergence speed, especially for instances with sizes higher than 400. This can be explained by the fact that the E/R operator is triggered periodically, thus, the convergence speed increases after the first reduction. Also, the instant speed-up value increases when the E/R operator is triggered for the first time, which indicates that this operator improves the performance of the LS algorithm. However, there are no further notices for the rest of the iterations and reductions. Next, we present the detailed experimental results with respect to the effectiveness of the suggested approach and the computational efficiency enhancement using the overall speedup metric.

4.3.1. Effectiveness of the suggested operator

As to assess the effectiveness of heuristic algorithms incorporating E/R operator, we computed the reduction rate with α drawn uniformly from $[0.1, 0.3]$ (consistent with the comparative experiments) and the corresponding deviation obtained for the suggested incorporation approach and compared it with the deviation obtained for the original classical algorithm. As it is mentioned earlier, we tested the incorporation of the E/R operator within GA and LS. In this experiment, we performed fifteen runs per algorithm per instance size, and used an exact solver to produce optimal evaluations and calculate the deviations. Table 4 summarizes the obtained results.

Considering the obtained deviation values, the key performance metric, one can observe that algorithms with the E/R operator generally outperform their original counterparts (i.e., classical GA and LS) across various instance sizes. This positive impact indicator is further supported by the obtained reduction rates, which reflects the consistency of the achieved solutions. Here, algorithms with the E/R operator exhibit low standard deviation, indicating more consistent performance throughout the search. Furthermore, considering the obtained standard deviation values for classical algorithms, one can not notice that they are also consistent in terms of performance. This results suggest that the E/R operator does not alter the consistency of the host search algorithm.

In summary, the obtained results indicate the effectiveness of incorporating an E/R operator within both Genetic Algorithm (GA) and Local Search (LS). In other words, the obtained results suggest that this integration strategy can improve the quality of solutions achieved by the search process.

4.3.2. Computational efficiency

Here, we focus on the computational efficiency offered by the suggested approach compared with the corresponding classical approach. That is, by computing the computational overall speedup as the main metric for this comparison. Furthermore, the overall speedup is defined as the ratio of the deviation obtained using a classical heuristic approach over the deviation obtained using this latter with an E/R operator. The following speedup statistics are calculated using the deviation values obtained in the previous experiment. Table 5 resumes the obtained results regarding the computational speedup obtained using GA and LS as support algorithms.

The obtained results show that both GA and LS with the E/R operator have an average speedup greater

Table 4. Experimental results concerning the reduction rate and the associated deviation.

Size		Algorithm					
		GA with E/R		GA	LS with E/R		LS
		reduction	deviation	deviation	reduction	deviation	deviation
100	max	9.00E-01	9.96E-04	**8.20E-04**	7.90E-01	1.72E-02	**3.65E-01**
	min	6.20E-01	**0.00E+00**	**0.00E+00**	5.20E-01	6.33E-04	**2.92E-04**
	std	1.07E-01	3.98E-04	3.42E-04	9.70E-02	6.21E-03	1.44E-01
	average	7.74E-01	**1.99E-04**	2.69E-04	6.60E-01	**5.80E-03**	7.86E-02
	median	7.40E-01	**0.00E+00**	**0.00E+00**	7.00E-01	**2.27E-03**	4.96E-03
200	max	7.30E-01	1.88E-03	**1.81E-02**	7.55E-01	**5.60E-03**	1.48E-02
	min	5.10E-01	**0.00E+00**	3.83E-04	5.35E-01	**1.04E-03**	4.81E-03
	std	8.26E-02	7.45E-04	6.89E-03	8.58E-02	1.70E-03	3.59E-03
	average	6.01E-01	**6.85E-04**	7.93E-03	6.04E-01	**3.54E-03**	8.10E-03
	median	5.75E-01	**3.26E-04**	6.36E-03	5.45E-01	**3.64E-03**	6.18E-03
300	max	6.37E-01	2.66E-03	**1.81E-02**	7.90E-01	**1.87E-02**	2.54E-02
	min	3.97E-01	**0.00E+00**	3.68E-03	5.37E-01	**3.93E-03**	1.03E-02
	std	9.00E-02	8.90E-04	5.88E-03	1.07E-01	5.53E-03	5.58E-03
	average	5.21E-01	**1.08E-03**	8.90E-03	6.68E-01	**9.67E-03**	1.67E-02
	median	5.53E-01	**9.55E-04**	4.79E-03	7.17E-01	**8.38E-03**	1.57E-02
400	max	7.80E-01	**1.55E-02**	2.26E-02	8.43E-01	**4.60E-02**	5.00E-02
	min	4.03E-01	**1.05E-03**	4.04E-03	5.98E-01	**7.10E-04**	9.34E-03
	std	1.46E-01	5.47E-03	6.37E-03	9.51E-02	1.67E-02	1.83E-02
	average	6.07E-01	**4.73E-03**	1.04E-02	7.05E-01	**1.68E-02**	2.69E-02
	median	5.48E-01	**2.34E-03**	9.28E-03	7.10E-01	**7.04E-03**	1.69E-02
500	max	7.48E-01	**7.29E-03**	2.21E-02	8.20E-01	**4.04E-02**	6.22E-02
	min	2.94E-01	**1.28E-03**	8.86E-03	5.88E-01	**1.60E-03**	2.97E-03
	std	1.61E-01	2.09E-03	4.30E-03	8.36E-02	1.30E-02	2.23E-02
	average	5.23E-01	**3.34E-03**	1.53E-02	7.04E-01	**1.90E-02**	3.14E-02
	median	4.86E-01	**2.94E-03**	1.45E-02	6.88E-01	2.12E-02	**2.08E-02**
600	max	8.07E-01	**5.16E-02**	1.57E-01	7.58E-01	**2.49E-02**	3.52E-02
	min	2.98E-01	**8.77E-04**	6.21E-03	6.45E-01	**2.32E-03**	2.99E-03
	std	2.02E-01	1.84E-02	5.60E-02	3.79E-02	8.37E-03	1.31E-02
	average	5.12E-01	**1.63E-02**	5.24E-02	7.06E-01	**1.05E-02**	1.91E-02
	median	4.57E-01	8.50E-03	1.89E-02	7.05E-01	**6.64E-03**	2.15E-02
700	max	7.53E-01	**4.57E-02**	1.34E-01	8.46E-01	**2.50E-02**	5.86E-02
	min	3.29E-01	**5.44E-03**	2.84E-02	5.31E-01	**1.14E-02**	2.52E-02
	std	1.51E-01	1.70E-02	4.05E-02	1.12E-01	5.45E-03	1.18E-02
	average	5.68E-01	**1.96E-02**	5.90E-02	7.23E-01	**1.95E-02**	3.95E-02
	median	5.43E-01	**6.67E-03**	3.19E-02	7.23E-01	**2.27E-02**	3.92E-02
800	max	6.50E-01	**3.42E-02**	8.04E-02	8.83E-01	**3.72E-02**	5.69E-02
	min	4.79E-01	**6.03E-03**	3.05E-02	7.66E-01	**4.50E-03**	1.31E-02
	std	6.46E-02	1.08E-02	2.07E-02	4.51E-02	1.19E-02	1.53E-02
	average	5.61E-01	**1.64E-02**	4.91E-02	8.23E-01	**1.62E-02**	3.02E-02
	median	5.46E-01	**9.30E-03**	3.58E-02	8.43E-01	**1.26E-02**	2.72E-02
900	max	7.16E-01	**1.81E-02**	4.54E-02	9.26E-01	**4.71E-02**	7.27E-02
	min	2.43E-01	**8.78E-03**	3.13E-02	7.02E-01	**1.01E-02**	2.30E-02
	std	1.51E-01	4.13E-03	5.31E-03	8.80E-02	1.33E-02	1.69E-02
	average	4.59E-01	**1.24E-02**	3.71E-02	8.10E-01	**2.22E-02**	4.24E-02
	median	4.53E-01	**9.31E-03**	3.50E-02	8.23E-01	**1.82E-02**	3.75E-02
1000	max	6.55E-01	**4.40E-02**	6.68E-02	9.02E-01	**5.35E-02**	8.60E-02
	min	2.89E-01	**5.50E-03**	3.28E-02	6.22E-01	**6.78E-03**	2.03E-02
	std	1.37E-01	1.34E-02	1.27E-02	1.05E-01	1.84E-02	2.49E-02
	average	5.31E-01	**2.00E-02**	4.19E-02	7.47E-01	**2.25E-02**	4.06E-02
	median	6.12E-01	**1.41E-02**	3.64E-02	7.04E-01	**1.02E-02**	2.42E-02

Values in bold indicate the best obtained result in the correspondent row, compared with the same algorithm.

size	GA					LS				
	max	min	average	median	std	max	min	average	median	std
100	7.81	0.98	3.95	3.19	1.67	6.50	0.83	2.93	3.13	1.71
200	10.76	1.18	3.44	3.29	2.25	7.46	0.89	3.57	3.46	1.73
300	7.40	0.85	3.36	3.49	1.17	5.02	1.89	2.57	2.48	0.62
400	8.86	0.92	3.33	3.37	1.57	8.04	1.85	3.41	2.66	1.70
500	8.52	0.54	4.03	4.79	1.78	4.74	0.96	2.81	2.60	1.19
600	21.62	0.95	3.40	2.73	1.61	5.00	0.78	2.78	2.94	1.07
700	6.00	0.90	2.88	2.94	1.15	5.19	0.80	2.76	2.61	1.08
800	5.17	1.29	3.50	3.42	1.41	5.18	1.02	3.44	3.35	0.74
900	4.56	0.98	2.76	3.11	0.99	6.10	1.20	3.59	3.35	1.06
1000	2.97	0.99	1.85	1.69	0.65	5.40	0.90	3.28	3.12	0.93

Table 5. Experimental results concerning the overall speedup rate.

than 1 across most instance sizes, suggesting a potential reduction in deviation and faster algorithms. The speedup appears more consistent for LS, possibly indicating a better enhancement with the E/R operator. Additionally, the speedup doesn't necessarily increase with larger instance sizes, implying the efficiency gains might not always scale proportionally with complexity. Overall, the E/R operator can improve computational efficiency alongside the previously observed gains in solution quality.

4.3.3. Sensitivity analysis of the uncertainty threshold α

As it is mentioned earlier, the reduction rate and the accuracy indicator depend on the predefined uncertainty threshold α. Furthermore, the choice of this latter may depend also on the choice of the information source for entropy evaluation. Hence, in the following experiment we study the impact of the uncertainty threshold on the efficiency of the suggested approach. First, we present in Figures 8 and 9 some illustrative examples of the mean deviation obtained using five runs of the suggested approach with different uncertainty threshold values (5 runs per size per algorithm).

The illustrations given in Figures 8 and 9 suggest that the uncertainty threshold value does not alter the speed of convergence at the early iterations of both algorithms (i.e., LS and GA). However, it constantly shows, in the late quarter of iterations, that the higher the value one chooses, the sooner it stagnates.

Next, we tested both algorithms (i.e., GA and LS) incorporating the E/R operator using random uncertainty thresholds chosen within three intervals: $[0.1, 0.3]$, $[0.4, 0.6]$, and $[0.7, 0.9]$. That is, using randomly generated instances with four sizes 100, 400, 700, and 1000. The aim of this experiment is to determine the adequate values and to assess the impact of its variations on the quality of solutions produced using LS and GA. That is, by calculating the deviations from the corresponding optimal solutions. The obtained results are summarized in Table 6.

One can observe, in Table 6, that the best deviation values are obtained for low values of the uncertainty threshold. Furthermore, the deviation increases when increasing the parameter value. Thus, the obtained results indicate that the uncertainty threshold should be chosen within the interval of $[0.1, 0.3]$. This can be explained by the fact that higher values may cause premature reduction, since the exploration procedures (i.e., genetic operators for GA and the feasible transformation procedure for LS) are algorithmically restrained to feasible space and their intrinsic limitations. Hence, higher values may not be reliable as a convergence indicator. We present an illustrative example in Figure 10.

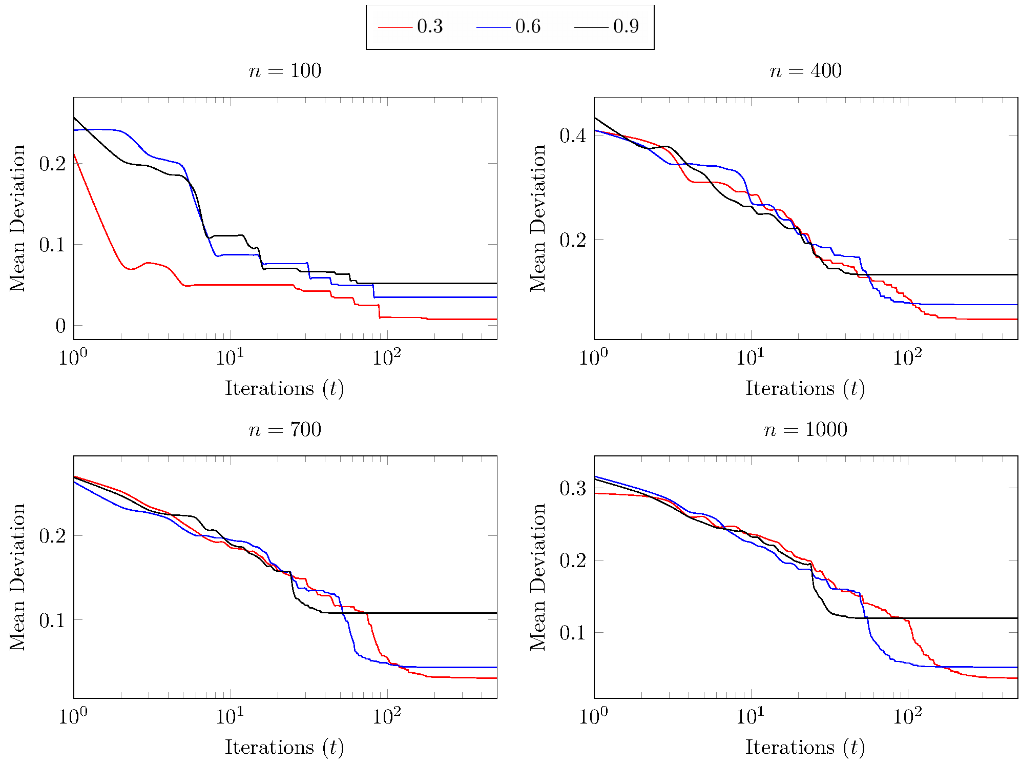

Figure 8. Illustrative example of the impact of the uncertainty threshold on the quality of solutions obtained using LS algorithm with E/R operator.

α	size	GA with E/R				LS with E/R			
		max	min	average	median	max	min	average	median
[0.1, 0.3]	100	2.19E-03	0.00E+00	**3.65E-04**	**0.00E+00**	1.90E-02	0.00E+00	**5.26E-03**	**3.44E-03**
	400	9.58E-03	4.67E-03	**5.86E-03**	**4.37E-03**	4.46E-02	1.64E-02	**2.47E-02**	**2.87E-02**
	700	1.49E-02	4.67E-03	**9.52E-03**	**8.90E-03**	4.03E-02	5.67E-03	**2.47E-02**	**2.87E-02**
	1000	2.66E-02	6.51E-03	**1.50E-02**	**1.41E-02**	4.70E-02	1.02E-02	**2.94E-02**	**2.87E-02**
[0.4, 0.6]	100	2.02E-02	0.00E+00	5.48E-03	3.61E-03	2.98E-02	0.00E+00	1.25E-02	1.49E-02
	400	6.09E-02	2.70E-02	4.26E-02	3.87E-02	8.61E-02	2.90E-02	5.68E-02	5.42E-02
	700	3.69E-02	1.50E-02	2.20E-02	3.04E-02	7.04E-02	2.23E-02	5.39E-02	5.92E-02
	1000	1.03E-01	3.82E-02	6.20E-02	5.62E-02	6.75E-02	3.24E-02	4.99E-02	5.01E-02
[0.7, 0.9]	100	4.35E-02	1.47E-02	3.04E-02	2.84E-02	4.63E-02	1.18E-02	3.13E-02	3.10E-02
	400	1.51E-01	8.91E-02	1.08E-01	1.06E-01	1.36E-01	9.57E-02	1.17E-01	1.25E-01
	700	1.46E-01	6.48E-02	9.56E-02	8.06E-02	1.04E-01	7.63E-02	9.22E-02	9.69E-02
	1000	1.28E-01	9.29E-02	1.10E-01	1.10E-01	1.12E-01	7.66E-02	1.01E-01	1.05E-01

Values in bold indicate the best obtained result in the correspondent size, compared with the same algorithm.

Table 6. Experimental results concerning the variations of the uncertainty threshold and its impact on solutions quality.

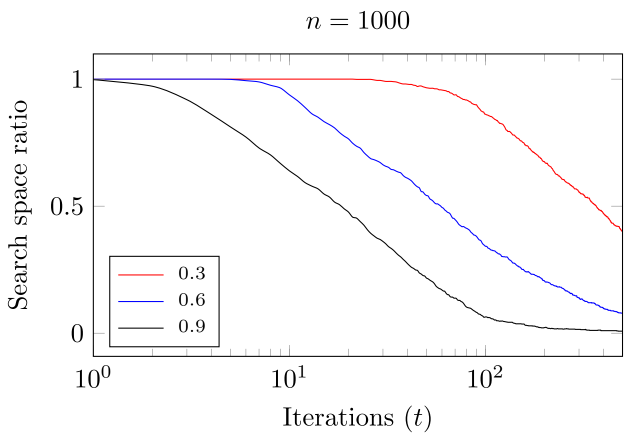

Figure 10. Illustrative example of the impact of the uncertainty threshold on the GA algorithm with E/R operator over 500 iterations, using a random instance with 1000 variables.

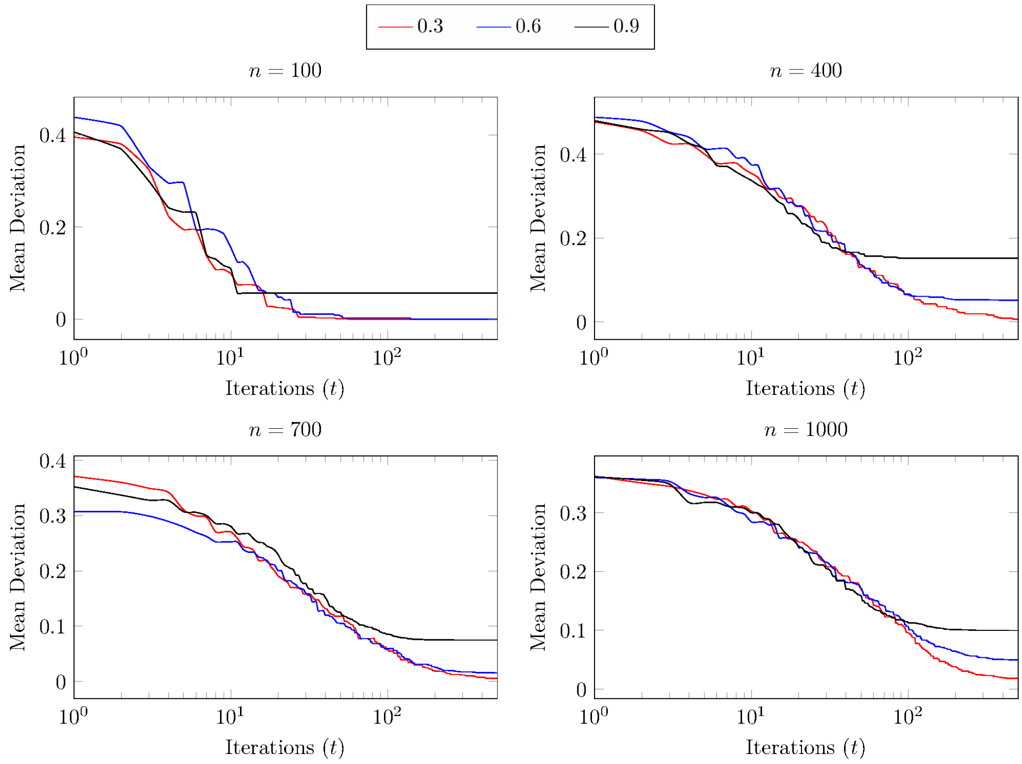

Figure 9. Illustrative example of the impact of the uncertainty threshold on the quality of solutions obtained using GA algorithm with E/R operator.

It is indicated that the search space size decreases faster when increasing the uncertainty threshold. In other words, the E/R is triggered for lower iterative occurrences of solution's components, which may falsely indicate convergence. Hence, the last drawn conclusion stands.

These observations motivate our choice of drawing $\alpha \sim \mathcal{U}[0.1, 0.3]$ in the interactive and comparative experiments: the interval is small enough to remain in the conservative, high-accuracy regime, while randomising within it avoids per-instance tuning and suggests that the operator's performance is stable.

4.4. Comparison with State-of-the-Art approaches on KP

To position the E/R operator against existing variable-fixing approaches, we compared it against the iterative SPVAR approach applied to Simulated Annealing (SA) and the Backbone strategy applied to Tabu Search (TS). These two methods represent the principal generic paradigms for dynamic variable fixing, namely frequency-based persistence and solution-intersection-based backbone identification, and are evaluated alongside the E/R operator incorporated within GA, TS, and SA.

All algorithms parameters are summarized in Table 7, and all algorithms are run with an equal number of evaluations. We conducted 5 independent runs on instances ranging from 100 to 1000 variables, using optimal solutions obtained via CPLEX solver for deviation calculations.

Table 8 presents the comparative results reporting the maximum, minimum, standard deviation, average, and median of the reduction rates and deviations from the optimum.

The obtained results point to the advantage of the E/R operator, particularly when integrated with the Genetic Algorithm and Tabu Search. GA with E/R achieves the lowest deviations for $n \leq 500$ (e.g., an average deviation of 0.16% at $n = 100$, and 0.98% at $n = 400$), alongside near-complete problem

Table 7. Summary of parameters for all comparisons with state-of-the-art algorithms.

Parameter	Value / Description
Simulated Annealing (SA), shared by SA-SPVAR and SA with E/R	
Cooling schedule	Geometric: $T_k = T_0 \cdot \gamma^k$
Initial temperature T_0	Calibrated to accept 80% of worsening moves initially
Cooling rate γ	0.995
Moves per temperature level	n
SA-SPVAR	
Sampling rounds	3
SA runs per round	20; final phase: 5 runs on reduced space
Persistence threshold	0.95
Tabu Search (TS), shared by TS-Backbone and TS with E/R	
Tabu tenure	15
Cycle length	$n/2$
TS-Backbone	
Backbone frequency threshold	0.90 (fixed to 1) / 0.10 (fixed to 0)
Stagnation response	Unfix 30% of backbone variables at random
GA with E/R	
Generations	500
Population size	100
Crossover type / rate	Uniform / $P_c = 0.7$
Mutation rate	$1/n$
Selection	Tournament (size 3)
E/R operator (all E/R variants)	
α (uncertainty threshold, all hosts)	Drawn $\sim \mathcal{U}[0.1, 0.3]$ per run

reduction. For larger instances ($n \geq 600$), TS with E/R is generally the best-performing method, attaining the best average deviation at $n = 600$, 700, 900, and 1000 (e.g., 4.85% at $n = 1000$ against 7.27% for TS-Backbone), with GA with E/R remaining a close second. The iterative SPVAR consistently produces the highest deviations despite moderate reduction, and SA with E/R achieves only limited reduction. Figure 11 illustrates these mean deviations across all instance sizes.

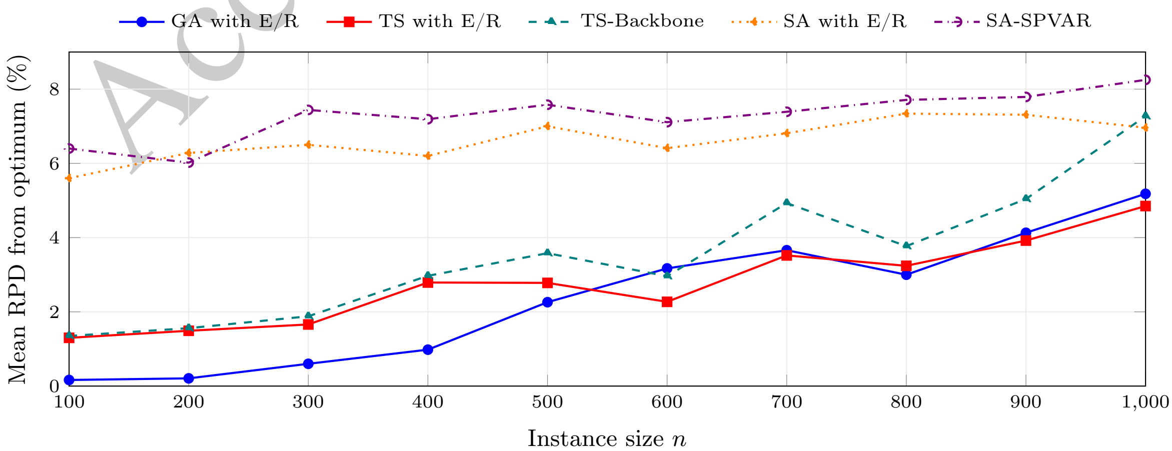

Figure 11. The obtained mean deviations from the optimum values according to instance size on the Knapsack Problem.

Table 8. The obtained experimental results regrading comparisons with state-of-the-art approaches on KP random instances.

n		SA-SPVAR		TS-Backbone		GA with E/R		SA with E/R		TS with E/R	
		Red. Rate	Div. Rate	Red. Rate	Div. Rate	Red. Rate	Div. Rate	Red. Rate	Div. Rate	Red. Rate	Div. Rate
100	max	2.90E-01	8.18E-02	6.40E-01	1.61E-02	1.00E+00	**6.20E-03**	3.90E-01	6.67E-02	5.60E-01	1.73E-02
	min	2.40E-01	5.20E-02	4.80E-01	9.42E-03	9.90E-01	**2.48E-04**	2.50E-01	4.63E-02	4.50E-01	8.18E-03
	std	1.82E-02	1.13E-02	6.78E-02	2.71E-03	4.47E-03	**2.56E-03**	5.79E-02	9.00E-03	3.97E-02	3.60E-03
	average	2.66E-01	6.40E-02	5.20E-01	1.35E-02	9.98E-01	**1.64E-03**	3.50E-01	5.60E-02	4.96E-01	1.30E-02
	median	2.70E-01	6.32E-02	5.00E-01	1.39E-02	1.00E+00	**7.43E-04**	3.80E-01	5.23E-02	4.90E-01	1.39E-02
200	max	3.05E-01	7.05E-02	9.00E-01	2.01E-02	1.00E+00	**4.73E-03**	3.25E-01	6.96E-02	6.70E-01	1.84E-02
	min	2.80E-01	5.33E-02	6.00E-01	1.27E-02	9.45E-01	**3.55E-04**	2.80E-01	5.32E-02	4.55E-01	1.04E-02
	std	1.02E-02	6.74E-03	1.20E-01	2.96E-03	2.26E-02	**1.78E-03**	1.75E-02	6.86E-03	8.07E-02	3.19E-03
	average	2.89E-01	6.02E-02	6.89E-01	1.56E-02	9.75E-01	**2.08E-03**	3.03E-01	6.28E-02	5.62E-01	1.49E-02
	median	2.90E-01	5.81E-02	6.45E-01	1.61E-02	9.70E-01	**1.54E-03**	3.00E-01	6.14E-02	5.70E-01	1.51E-02
300	max	2.73E-01	8.07E-02	8.00E-01	2.29E-02	9.50E-01	**1.01E-02**	2.97E-01	7.65E-02	9.23E-01	1.84E-02
	min	2.10E-01	5.82E-02	6.17E-01	1.45E-02	8.97E-01	**1.19E-03**	2.50E-01	5.99E-02	6.77E-01	1.45E-02
	std	2.39E-02	9.43E-03	7.72E-02	3.31E-03	2.33E-02	3.22E-03	1.77E-02	6.63E-03	9.85E-02	**1.67E-03**
	average	2.40E-01	7.44E-02	6.63E-01	1.88E-02	9.33E-01	**6.00E-03**	2.78E-01	6.50E-02	8.37E-01	1.66E-02
	median	2.40E-01	7.84E-02	6.37E-01	1.92E-02	9.47E-01	**5.86E-03**	2.83E-01	6.22E-02	8.60E-01	1.62E-02
400	max	2.68E-01	8.02E-02	9.40E-01	4.63E-02	9.45E-01	**1.45E-02**	2.97E-01	7.13E-02	9.38E-01	3.85E-02
	min	2.30E-01	6.57E-02	6.03E-01	2.33E-02	9.20E-01	**5.36E-03**	2.95E-01	5.33E-02	5.30E-01	2.14E-02
	std	1.56E-02	6.72E-03	1.34E-01	9.56E-03	1.06E-02	**4.33E-03**	1.37E-03	6.80E-03	1.66E-01	6.68E-03
	average	2.58E-01	7.19E-02	7.29E-01	2.97E-02	9.38E-01	**9.81E-03**	2.96E-01	6.20E-02	6.71E-01	2.79E-02
	median	2.65E-01	6.97E-02	6.65E-01	2.50E-02	9.43E-01	**7.77E-03**	2.95E-01	6.14E-02	6.72E-01	2.75E-02
500	max	2.42E-01	8.58E-02	9.06E-01	5.03E-02	9.48E-01	3.66E-02	2.88E-01	7.51E-02	9.68E-01	**3.15E-02**
	min	2.24E-01	7.15E-02	6.16E-01	2.80E-02	9.06E-01	**1.30E-02**	2.74E-01	6.45E-02	6.66E-01	2.00E-02
	std	6.48E-03	5.87E-03	1.09E-01	8.86E-03	1.63E-02	1.08E-02	5.73E-03	4.69E-03	1.17E-01	**4.62E-03**
	average	2.32E-01	7.58E-02	7.36E-01	3.58E-02	9.23E-01	**2.26E-02**	2.80E-01	7.00E-02	8.05E-01	2.78E-02
	median	2.32E-01	7.44E-02	7.24E-01	3.39E-02	9.16E-01	**1.78E-02**	2.80E-01	7.02E-02	8.04E-01	2.94E-02
600	max	2.87E-01	7.68E-02	9.63E-01	4.67E-02	9.40E-01	4.63E-02	3.17E-01	6.55E-02	9.67E-01	**2.56E-02**
	min	2.53E-01	6.45E-02	8.63E-01	2.12E-02	8.97E-01	2.43E-02	2.95E-01	6.06E-02	5.95E-01	**1.91E-02**
	std	1.49E-02	4.59E-03	4.05E-02	1.01E-02	2.09E-02	9.05E-03	8.93E-03	**2.04E-03**	1.73E-01	3.03E-03
	average	2.69E-01	7.11E-02	9.23E-01	2.96E-02	9.21E-01	3.17E-02	3.08E-01	6.41E-02	7.68E-01	**2.27E-02**
	median	2.65E-01	7.12E-02	9.17E-01	2.57E-02	9.33E-01	2.91E-02	3.10E-01	6.50E-02	6.70E-01	**2.36E-02**
700	max	2.76E-01	8.05E-02	9.64E-01	5.38E-02	9.59E-01	5.04E-02	2.99E-01	7.72E-02	9.59E-01	**3.75E-02**
	min	2.49E-01	6.41E-02	6.60E-01	4.47E-02	9.20E-01	**2.24E-02**	2.57E-01	6.13E-02	5.66E-01	3.23E-02
	std	1.15E-02	6.23E-03	1.11E-01	3.77E-03	1.63E-02	1.14E-02	1.68E-02	6.63E-03	1.81E-01	**1.97E-03**
	average	2.58E-01	7.39E-02	8.34E-01	4.93E-02	9.30E-01	3.66E-02	2.84E-01	6.81E-02	7.63E-01	**3.52E-02**
	median	2.51E-01	7.64E-02	8.47E-01	4.84E-02	9.23E-01	4.05E-02	2.90E-01	6.75E-02	6.83E-01	**3.53E-02**
800	max	2.61E-01	8.04E-02	9.75E-01	**4.04E-02**	9.35E-01	5.02E-02	3.09E-01	7.85E-02	9.66E-01	4.18E-02
	min	2.45E-01	6.97E-02	6.80E-01	3.31E-02	9.12E-01	**2.00E-02**	2.92E-01	6.99E-02	6.67E-01	2.65E-02
	std	6.52E-03	4.26E-03	1.43E-01	**3.18E-03**	1.01E-02	1.23E-02	6.77E-03	3.87E-03	1.60E-01	6.62E-03
	average	2.55E-01	7.71E-02	8.29E-01	3.77E-02	9.27E-01	**3.00E-02**	2.98E-01	7.34E-02	7.87E-01	3.24E-02
	median	2.55E-01	7.80E-02	8.38E-01	3.91E-02	9.31E-01	**2.71E-02**	2.95E-01	7.23E-02	6.74E-01	3.04E-02
900	max	2.59E-01	8.82E-02	9.60E-01	5.59E-02	9.32E-01	5.67E-02	2.98E-01	7.86E-02	9.77E-01	**4.12E-02**
	min	2.33E-01	6.86E-02	6.66E-01	4.14E-02	9.09E-01	**3.02E-02**	2.83E-01	6.76E-02	6.53E-01	3.64E-02
	std	1.01E-02	7.69E-03	1.24E-01	6.42E-03	9.20E-03	1.20E-02	7.35E-03	3.96E-03	1.50E-01	**1.94E-03**
	average	2.42E-01	7.79E-02	8.84E-01	5.04E-02	9.24E-01	4.13E-02	2.91E-01	7.31E-02	8.19E-01	**3.92E-02**
	median	2.40E-01	7.91E-02	9.43E-01	5.33E-02	9.27E-01	**3.77E-02**	2.96E-01	7.31E-02	8.61E-01	3.95E-02
1000	max	2.55E-01	8.64E-02	9.70E-01	8.31E-02	9.64E-01	6.93E-02	2.99E-01	7.20E-02	9.57E-01	**5.65E-02**
	min	2.37E-01	7.80E-02	6.72E-01	6.31E-02	9.38E-01	**3.77E-02**	2.77E-01	6.76E-02	6.31E-01	4.26E-02
	std	7.13E-03	3.24E-03	1.27E-01	8.00E-03	1.02E-02	1.19E-02	8.12E-03	**1.84E-03**	1.62E-01	5.77E-03
	average	2.43E-01	8.25E-02	8.75E-01	7.27E-02	9.54E-01	5.18E-02	2.86E-01	6.96E-02	8.25E-01	**4.85E-02**
	median	2.41E-01	8.25E-02	9.48E-01	7.22E-02	9.57E-01	5.25E-02	2.84E-01	7.00E-02	9.31E-01	**4.73E-02**

Values in bold indicate the best obtained deviation in the corresponding row across all methods.

4.4.1. Statistical validation

To test whether the observed performance differences are statistically significant, we conduct pairwise hypothesis testing on the deviation values from the 50 paired observations (i.e., 10 instance sizes with 5 runs each). Given the non-normal distribution of metaheuristic deviations and the paired nature of the data (i.e., each algorithm was run on the same instances), we employ the Wilcoxon signed-rank test [49], which is the used non-parametric test for comparing stochastic optimization algorithms over multiple problems [10], and has been effectively adopted in evaluating advanced heuristics for the Knapsack Problem [55]. The null hypothesis H_0 is that the median difference in deviations between two algorithms is zero. We adopt a significance level of 0.05 throughout. Table 9 presents the symmetric matrix of p-values obtained from the pairwise Wilcoxon signed-rank tests.

Table 9. Pairwise Wilcoxon signed-rank p-values on the obtained deviations from the optimum on the KP random instances.

	SA-SPVAR	TS-Backbone	GA with E/R	SA with E/R	TS with E/R
SA-SPVAR	-	**3.55E-15**	**1.78E-15**	**2.83E-06**	**1.78E-15**
TS-Backbone	**3.55E-15**	-	**4.06E-09**	**1.24E-13**	**1.45E-06**
GA with E/R	**1.78E-15**	**4.06E-09**	-	**3.55E-15**	**1.22E-02**
SA with E/R	**2.83E-06**	**1.24E-13**	**3.55E-15**	-	**1.78E-15**
TS with E/R	**1.78E-15**	**1.45E-06**	**1.22E-02**	**1.78E-15**	-

Values in bold indicate a statistically significant difference at the 0.05 level.

All pairwise comparisons yield p-values below the adopted significance level of 0.05, so the performance differences among the strategies are statistically significant. In particular, the proposed GA with E/R significantly outperforms both SA-SPVAR and TS-Backbone. The two best-performing methods, GA with E/R and TS with E/R, are themselves separated at the 0.05 level, i.e., $p = 1.22 \times 10^{-2}$, with a slight advantage for GA. This separation is consistent with their complementary performance profiles: GA with E/R tends to dominate for small-to-medium instances (i.e., $n \leq 500$), where population diversity and global exploration are more beneficial, while TS with E/R performs best for larger instances (i.e., $n \geq 600$). Thus, the E/R operator offers a consistent improvement over state-of-the-art variable-fixing approaches on KP random instances.

4.5. Comparison with State-of-the-Art Approaches on QUBO

Having established the effectiveness of the E/R operator on KP, we now consider the unconstrained case. Three standard QUBO benchmark suites are used: Beasley with 40 instances of size $n \in \{50, 100, 250, 500\}$, GKA with 45 instances, grouped into sets `a-f` with size n ranging from 20 to 500, and BE with 80 instances with size $n \in \{100, 120, 150, 200, 250\}$. Each instance is solved across 5 independent runs. As in the Knapsack study, the E/R uncertainty threshold α is drawn uniformly at random from $[0.1, 0.3]$ for every reported run. We adopt for these runs the same parameter settings presented in Table 7. For each instance and algorithm, we record the gap to the Best Known Value (BKV) $\Delta = \mathrm{BKV} - f_{\text{found}}$, and subsequently the deviation rate Δ/BKV. That is, to makes deviations comparable across instances whose objective values may differ by several orders of magnitude. For every instance and algorithm we summarize this ratio over the five runs and each aggregate summary table reports the corresponding descriptive statistical details of the per-instance mean ratios within each size or structural group. The pairwise Wilcoxon signed-rank tests are applied to the per-instance mean deviations Δ. For each benchmark, we first present the aggregate summary and a discussion of the key findings, and then the pairwise Wilcoxon signed-rank p-value matrix.

4.5.1. Beasley Benchmark Suite

The Beasley instance matrices are sparse (i.e., density 0.1), with integer coefficients drawn uniformly from $[-100, 100]$ and ten instances per size. Table 10 reports the per-size reduction rates and deviation rates according to the instances size, and Table 11 the pairwise Wilcoxon tests.

Table 10. Aggregate results on the Beasley suite grouped by problem size.

n	Stat.	SA-SPVAR		TS-Backbone		GA with E/R		SA with E/R		TS with E/R	
		Red. Rate	Div. Rate	Red. Rate	Div. Rate	Red. Rate	Div. Rate	Red. Rate	Div. Rate	Red. Rate	Div. Rate
50	min	9.12E-01	**0.00E+00**	6.52E-01	**0.00E+00**	5.84E-01	**0.00E+00**	8.44E-01	**0.00E+00**	5.76E-01	**0.00E+00**
	max	1.00E+00	**0.00E+00**	7.00E-01	1.48E-03	9.40E-01	**0.00E+00**	1.00E+00	1.48E-03	7.00E-01	1.48E-03
	std	2.92E-02	**0.00E+00**	1.77E-02	4.68E-04	9.82E-02	**0.00E+00**	5.24E-02	4.68E-04	4.06E-02	4.68E-04
	average	9.72E-01	**0.00E+00**	6.86E-01	1.48E-04	7.36E-01	**0.00E+00**	9.53E-01	1.48E-04	6.69E-01	1.48E-04
	median	9.80E-01	**0.00E+00**	6.92E-01	**0.00E+00**	7.36E-01	**0.00E+00**	9.80E-01	**0.00E+00**	6.84E-01	**0.00E+00**
100	min	8.78E-01	**0.00E+00**	6.04E-01	**0.00E+00**	6.32E-01	**0.00E+00**	7.90E-01	**0.00E+00**	5.84E-01	**0.00E+00**
	max	1.00E+00	1.66E-03	7.00E-01	1.66E-03	9.80E-01	**0.00E+00**	1.00E+00	1.57E-04	7.00E-01	5.33E-04
	std	4.90E-02	5.85E-04	3.25E-02	5.24E-04	9.22E-02	**0.00E+00**	8.24E-02	5.30E-05	4.91E-02	1.69E-04
	average	9.71E-01	2.87E-04	6.81E-01	1.66E-04	8.12E-01	**0.00E+00**	9.41E-01	2.36E-05	6.62E-01	5.33E-05
	median	1.00E+00	**0.00E+00**	7.00E-01	**0.00E+00**	7.96E-01	**0.00E+00**	1.00E+00	**0.00E+00**	6.93E-01	**0.00E+00**
250	min	9.43E-01	**0.00E+00**	6.67E-01	**0.00E+00**	8.65E-01	**0.00E+00**	9.07E-01	**0.00E+00**	5.83E-01	**0.00E+00**
	max	1.00E+00	7.48E-04	7.00E-01	1.05E-04	9.50E-01	**0.00E+00**	1.00E+00	2.50E-05	6.85E-01	1.31E-04
	std	1.94E-02	3.03E-04	1.05E-02	3.36E-05	2.80E-02	**0.00E+00**	3.74E-02	7.91E-06	3.21E-02	4.94E-05
	average	9.90E-01	1.99E-04	6.88E-01	1.64E-05	9.20E-01	**0.00E+00**	9.82E-01	2.50E-06	6.50E-01	2.61E-05
	median	1.00E+00	1.25E-05	6.90E-01	**0.00E+00**	9.21E-01	**0.00E+00**	1.00E+00	**0.00E+00**	6.56E-01	**0.00E+00**
500	min	9.63E-01	**0.00E+00**	6.18E-01	**0.00E+00**	9.15E-01	**0.00E+00**	9.03E-01	**0.00E+00**	5.71E-01	**0.00E+00**
	max	1.00E+00	6.82E-04	7.30E-01	1.36E-03	9.72E-01	**2.61E-04**	1.00E+00	4.35E-04	6.67E-01	8.18E-04
	std	1.26E-02	2.08E-04	3.52E-02	4.34E-04	1.75E-02	**1.02E-04**	4.03E-02	1.59E-04	3.08E-02	2.92E-04
	average	9.91E-01	1.71E-04	6.66E-01	5.60E-04	9.54E-01	**5.81E-05**	9.54E-01	9.85E-05	6.15E-01	4.02E-04
	median	9.99E-01	1.03E-04	6.68E-01	4.67E-04	9.60E-01	**4.61E-06**	9.58E-01	1.33E-05	6.13E-01	4.17E-04

Values in bold indicate the best (lowest) deviation rate in the corresponding row across all methods.

At small and medium scales ($n \leq 250$), all algorithms perform reasonably well, yet GA with E/R reaches a zero deviation across all 30 instances in these groups, with no run deviating from the BKV. SA-SPVAR shows early drawbacks at $n = 100$, and a comparable at $n = 250$. At $n = 500$, TS-Backbone degrades with regards to mean deviation rate, while GA with E/R maintains a better mean deviation rate, which indicates that entropy-guided variable fixing yields more resilient convergence than backbone-based strategies at this scale. SA with E/R follows as the second-best method at $n = 500$, while TS with E/R and SA-SPVAR demonstrates lower performance.

Table 11. Pairwise Wilcoxon signed-rank p-values on the Beasley suite.

	SA-SPVAR	TS-Backbone	GA with E/R	SA with E/R	TS with E/R
SA-SPVAR	-	9.20E-02	**2.34E-04**	**3.92E-03**	4.33E-01
TS-Backbone	9.20E-02	-	**8.05E-04**	**6.22E-03**	1.88E-01
GA with E/R	**2.34E-04**	**8.05E-04**	-	1.77E-01	**8.03E-04**
SA with E/R	**3.92E-03**	**6.22E-03**	1.77E-01	-	**7.23E-03**
TS with E/R	4.33E-01	1.88E-01	**8.03E-04**	**7.23E-03**	-

Values in bold indicate a statistically significant difference at the 0.05 level.

Table 11 reports the pairwise Wilcoxon tests. GA with E/R is statistically superior to both SA-SPVAR and TS-Backbone, confirming that the aggregate advantage observed in Table 10 is not driven by a few outlier instances. The difference between GA with E/R and SA with E/R is not significant, indicating that these two E/R variants perform comparably overall. SA-SPVAR against TS with E/R is also not significant. This can be explained by the fact that TS with E/R's strong performance on smaller instances is offset by its degradation at $n = 500$. Fixing the search engine isolates the effect of the reduction rule: within Simulated Annealing, E/R lowers the deviation rate significantly over SPVAR (i.e., SA-SPVAR against SA with E/R, p = 3.92E-03), while within Tabu Search the entropy and backbone rules stay comparable (i.e., TS-Backbone against TS with E/R, p = 1.88E-01).

4.5.2. GKA Benchmark Suite

The GKA sets `a–f` each fix a different size and density, ranging from $d = 0.0625$ to fully dense with coefficients in $[-100, 100]$. We note that two sets are atypical: set `b` is fully dense ($d = 1$) with a sign-constrained coefficient structure (i.e., a non-positive diagonal and a non-negative off-diagonal) that yields optima of small magnitude, and set `f` is the only $n = 500$ group and the only one without proven optima. We retain this heterogeneity deliberately, since it exercises the operator on structurally distinct landscapes within a single suite. Table 12 reports the per-set results (i.e., sets `a–f`), and Table 13 the pairwise Wilcoxon tests.

Table 12. Aggregate results on the GKA suite grouped by set `a–f`.

Set	Stat.	SA-SPVAR		TS-Backbone		GA with E/R		SA with E/R		TS with E/R	
		Red. Rate	Div. Rate	Red. Rate	Div. Rate	Red. Rate	Div. Rate	Red. Rate	Div. Rate	Red. Rate	Div. Rate
a	min	9.60E-01	**0.00E+00**	6.80E-01	**0.00E+00**	7.54E-01	**0.00E+00**	9.60E-01	**0.00E+00**	6.26E-01	**0.00E+00**
	max	1.00E+00	6.28E-04	7.00E-01	**0.00E+00**	9.13E-01	**0.00E+00**	1.00E+00	**0.00E+00**	7.00E-01	**0.00E+00**
	std	1.79E-02	2.22E-04	7.07E-03	**0.00E+00**	5.52E-02	**0.00E+00**	1.73E-02	**0.00E+00**	2.72E-02	**0.00E+00**
	average	9.90E-01	7.85E-05	6.98E-01	**0.00E+00**	8.45E-01	**0.00E+00**	9.91E-01	**0.00E+00**	6.86E-01	**0.00E+00**
	median	1.00E+00	**0.00E+00**	7.00E-01	**0.00E+00**	8.41E-01	**0.00E+00**	1.00E+00	**0.00E+00**	7.00E-01	**0.00E+00**
b	min	9.10E-01	**0.00E+00**	6.80E-01	**0.00E+00**	3.30E-01	**0.00E+00**	9.60E-01	**0.00E+00**	6.40E-01	**0.00E+00**
	max	1.00E+00	4.79E-02	7.00E-01	1.10E-02	8.90E-01	2.21E-02	1.00E+00	5.89E-02	6.90E-01	**0.00E+00**
	std	3.30E-02	1.70E-02	6.94E-03	3.49E-03	1.74E-01	8.28E-03	1.44E-02	2.02E-02	1.84E-02	**0.00E+00**
	average	9.83E-01	1.77E-02	6.94E-01	1.10E-03	7.48E-01	4.23E-03	9.93E-01	1.11E-02	6.63E-01	**0.00E+00**
	median	1.00E+00	2.14E-02	6.97E-01	**0.00E+00**	8.31E-01	**0.00E+00**	1.00E+00	**0.00E+00**	6.63E-01	**0.00E+00**
c	min	1.00E+00	**0.00E+00**	7.00E-01	**0.00E+00**	7.98E-01	**0.00E+00**	1.00E+00	**0.00E+00**	6.96E-01	**0.00E+00**
	max	1.00E+00	**0.00E+00**	7.00E-01	**0.00E+00**	1.00E+00	**0.00E+00**	1.00E+00	**0.00E+00**	7.00E-01	**0.00E+00**
	std	0.00E+00	**0.00E+00**	1.20E-16	**0.00E+00**	6.95E-02	**0.00E+00**	0.00E+00	**0.00E+00**	1.51E-03	**0.00E+00**
	average	1.00E+00	**0.00E+00**	7.00E-01	**0.00E+00**	9.03E-01	**0.00E+00**	1.00E+00	**0.00E+00**	6.99E-01	**0.00E+00**
	median	1.00E+00	**0.00E+00**	7.00E-01	**0.00E+00**	8.88E-01	**0.00E+00**	1.00E+00	**0.00E+00**	7.00E-01	**0.00E+00**
d	min	8.52E-01	**0.00E+00**	6.66E-01	**0.00E+00**	8.62E-01	**0.00E+00**	7.98E-01	**0.00E+00**	6.48E-01	**0.00E+00**
	max	1.00E+00	1.50E-03	7.00E-01	**0.00E+00**	9.58E-01	**0.00E+00**	1.00E+00	1.22E-03	7.00E-01	**0.00E+00**
	std	4.83E-02	5.70E-04	1.08E-02	**0.00E+00**	3.16E-02	**0.00E+00**	7.99E-02	4.14E-04	1.84E-02	**0.00E+00**
	average	9.79E-01	3.65E-04	6.97E-01	**0.00E+00**	9.08E-01	**0.00E+00**	9.44E-01	1.85E-04	6.89E-01	**0.00E+00**
	median	1.00E+00	**0.00E+00**	7.00E-01	**0.00E+00**	9.03E-01	**0.00E+00**	9.84E-01	**0.00E+00**	7.00E-01	**0.00E+00**
e	min	9.43E-01	**0.00E+00**	6.75E-01	**0.00E+00**	8.87E-01	**0.00E+00**	8.65E-01	**0.00E+00**	5.99E-01	**0.00E+00**
	max	1.00E+00	5.12E-04	7.00E-01	5.69E-05	9.57E-01	**0.00E+00**	1.00E+00	**0.00E+00**	6.97E-01	**0.00E+00**
	std	2.48E-02	1.86E-04	1.08E-02	2.54E-05	3.39E-02	**0.00E+00**	6.68E-02	**0.00E+00**	3.58E-02	**0.00E+00**
	average	9.82E-01	2.89E-04	6.89E-01	1.14E-05	9.23E-01	**0.00E+00**	9.47E-01	**0.00E+00**	6.45E-01	**0.00E+00**
	median	9.96E-01	2.92E-04	6.90E-01	**0.00E+00**	9.29E-01	**0.00E+00**	9.86E-01	**0.00E+00**	6.43E-01	**0.00E+00**
f	min	9.20E-01	5.98E-05	5.62E-01	1.51E-04	9.62E-01	**0.00E+00**	8.13E-01	5.04E-05	5.38E-01	3.33E-04
	max	1.00E+00	2.08E-03	6.62E-01	1.59E-03	9.89E-01	**3.11E-04**	1.00E+00	8.54E-04	6.59E-01	1.32E-03
	std	3.46E-02	8.39E-04	4.03E-02	5.73E-04	1.14E-02	**1.37E-04**	7.55E-02	3.47E-04	4.36E-02	4.09E-04
	average	9.81E-01	5.88E-04	6.32E-01	8.08E-04	9.74E-01	**6.63E-05**	9.46E-01	2.36E-04	5.96E-01	6.61E-04
	median	9.99E-01	2.64E-04	6.51E-01	6.47E-04	9.75E-01	**0.00E+00**	9.75E-01	7.79E-05	5.99E-01	4.93E-04

Values in bold indicate the best (lowest) deviation rate in the corresponding row across all methods.

The structurally simpler GKA sets separate cleanly from the dense set `f`, one can observe that: set `c` is solved to the BKV by every method, and set `a` is solved by all approaches but SA-SPVAR. On sets `d` and `e` the E/R variants and TS-Backbone reach the BKV with small deviation. Set `b` is the hardest of the smaller sets: only TS with E/R attains a zero mean deviation rate there, while we obtained relatively high deviation rates with the remaining methods, including GA with E/R. On the other hand, regarding the set `f`, GA with E/R achieves the lowest mean deviation rate, an order of magnitude below TS-Backbone and TS with E/R. SA with E/R occupies a middle position, while still roughly 3.6 times the GA with E/R ratio.

The pairwise Wilcoxon tests in Table 13 follow the structure of this suite. GA with E/R is significantly better than SA-SPVAR and SA with E/R. However, the comparison between GA with E/R and TS-

Table 13. Pairwise Wilcoxon signed-rank p-values on the GKA suite ($N = 45$).

	SA-SPVAR	TS-Backbone	GA with E/R	SA with E/R	TS with E/R
SA-SPVAR	-	1.52E-01	**7.73E-05**	**7.96E-04**	5.37E-02
TS-Backbone	1.52E-01	-	6.40E-02	2.26E-01	6.01E-01
GA with E/R	**7.73E-05**	6.40E-02	-	**6.38E-03**	8.32E-02
SA with E/R	**7.96E-04**	2.26E-01	**6.38E-03**	-	2.50E-01
TS with E/R	5.37E-02	6.01E-01	8.32E-02	2.50E-01	-

Values in bold indicate a statistically significant difference at the 0.05 level.

Backbone falls short of the 5% threshold (i.e., $p = 0.0640$), and GA with E/R against TS with E/R is also not significant. This is a direct consequence of the zero-deviation floor: since most instances in sets `a`–`e` are perfectly solved by multiple algorithms, the statistical test draws its difficulty, primarily from the five set-`f` instances. The non-significant result between TS-Backbone and TS with E/R confirms that these two methods are comparable outside the set-`f` regime. The same-engine comparison follows the same pattern: within Simulated Annealing E/R again improves significantly over SPVAR (i.e., $p = 7.96$E-04), while within Tabu Search the two rules are not separated (i.e., $p = 6.01$E-01), E/R keeping only a small numerical edge on the dense set `f`.

4.5.3. BE Benchmark Suite

The BE instances pair a diagonal in $[-100, 100]$ with a smaller off-diagonal in $[-50, 50]$, and at the intermediate sizes they are provided in low- and high-density variants (i.e., $d = 0.3$ and $d = 0.8$). Being the largest of the three suites, BE gives the pairwise Wilcoxon tests the most statistical significance, while its paired densities isolate the effect of problem density at a fixed size. Table 14 reports the per-size results, and Table 15 the pairwise Wilcoxon tests.

On the BE suite, three methods form a closely matched high-performing group: GA with E/R attains the lowest overall mean deviation rate, narrowly ahead of TS with E/R and TS-Backbone. Furthermore, GA with E/R reaches a zero mean deviation rate at $n = 100$ and stays at or below 3.40E-05 in every size group. TS-Backbone performs better on this suite than on Beasley or GKA set `f`, remaining at or below 6.37E-05 at all sizes. While the overall worst performance was recorded for SA with E/R and SA-SPVAR, the latter producing the largest mean deviation rates at $n = 120$–200.

The pairwise tests on the BE suite (see Table 15) confirm this tiered structure. GA with E/R, TS with E/R, and TS-Backbone are mutually indistinguishable: none of the three pairwise comparisons among them is significant. Each of these three is significantly better than both SA with E/R and SA-SPVAR. Thus, although GA with E/R records the lowest mean, on these comparatively regular instances its advantage over the other two approaches is not statistically significant. Controlling for the search engine reproduces this: SA with E/R is significantly better than SA-SPVAR (i.e., $p = 1.98$E-04), while TS with E/R and TS-Backbone stay indistinguishable (i.e., $p = 6.13$E-01).

4.5.4. Cross-Benchmark Synthesis

Across all 165 QUBO instances, GA with E/R is the most reliable method overall, though the picture is suite-dependent. It attains the lowest aggregate mean deviation rate on the Beasley suite and on the BE suite, and it achieves the lowest deviation on the dense GKA set `f`, while solving GKA sets `a`, `c`, `d`, and

Table 14. Aggregate results on the BE suite grouped by problem size.

n	Stat.	SA-SPVAR		TS-Backbone		GA with E/R		SA with E/R		TS with E/R	
		Red. Rate	Div. Rate	Red. Rate	Div. Rate	Red. Rate	Div. Rate	Red. Rate	Div. Rate	Red. Rate	Div. Rate
100	min	9.10E-01	**0.00E+00**	7.00E-01	**0.00E+00**	8.34E-01	**0.00E+00**	7.48E-01	**0.00E+00**	6.20E-01	**0.00E+00**
	max	1.00E+00	**0.00E+00**	7.00E-01	**0.00E+00**	9.64E-01	**0.00E+00**	1.00E+00	**0.00E+00**	7.00E-01	**0.00E+00**
	std	2.85E-02	**0.00E+00**	1.17E-16	**0.00E+00**	4.20E-02	**0.00E+00**	8.57E-02	**0.00E+00**	2.51E-02	**0.00E+00**
	average	9.91E-01	**0.00E+00**	7.00E-01	**0.00E+00**	9.12E-01	**0.00E+00**	9.52E-01	**0.00E+00**	6.91E-01	**0.00E+00**
	median	1.00E+00	**0.00E+00**	7.00E-01	**0.00E+00**	9.24E-01	**0.00E+00**	1.00E+00	**0.00E+00**	7.00E-01	**0.00E+00**
120	min	8.17E-01	**0.00E+00**	6.45E-01	**0.00E+00**	8.42E-01	**0.00E+00**	6.73E-01	**0.00E+00**	4.42E-01	**0.00E+00**
	max	1.00E+00	5.64E-03	7.00E-01	**0.00E+00**	9.83E-01	3.19E-04	1.00E+00	1.34E-03	7.00E-01	**0.00E+00**
	std	4.89E-02	1.26E-03	1.23E-02	**0.00E+00**	3.66E-02	7.13E-05	8.43E-02	3.98E-04	5.85E-02	**0.00E+00**
	average	9.80E-01	4.74E-04	6.97E-01	**0.00E+00**	9.23E-01	1.59E-05	9.48E-01	1.78E-04	6.75E-01	**0.00E+00**
	median	1.00E+00	**0.00E+00**	7.00E-01	**0.00E+00**	9.26E-01	**0.00E+00**	1.00E+00	**0.00E+00**	6.95E-01	**0.00E+00**
150	min	8.25E-01	**0.00E+00**	5.81E-01	**0.00E+00**	8.56E-01	**0.00E+00**	7.32E-01	**0.00E+00**	5.19E-01	**0.00E+00**
	max	1.00E+00	2.98E-03	7.00E-01	**1.07E-04**	9.89E-01	2.88E-04	1.00E+00	2.95E-03	7.00E-01	5.77E-04
	std	4.25E-02	7.47E-04	3.45E-02	**2.40E-05**	3.16E-02	6.45E-05	8.14E-02	7.68E-04	5.58E-02	1.29E-04
	average	9.76E-01	4.86E-04	6.87E-01	**5.36E-06**	9.39E-01	1.44E-05	9.30E-01	3.62E-04	6.60E-01	3.33E-05
	median	1.00E+00	4.45E-05	7.00E-01	**0.00E+00**	9.43E-01	**0.00E+00**	9.59E-01	**0.00E+00**	6.81E-01	**0.00E+00**
200	min	7.84E-01	**0.00E+00**	5.78E-01	**0.00E+00**	8.41E-01	**0.00E+00**	7.48E-01	**0.00E+00**	4.91E-01	**0.00E+00**
	max	1.00E+00	2.99E-03	7.00E-01	4.51E-04	9.96E-01	3.89E-04	1.00E+00	1.16E-03	7.00E-01	**2.25E-04**
	std	5.58E-02	7.44E-04	3.85E-02	1.52E-04	4.07E-02	1.06E-04	7.37E-02	3.30E-04	5.14E-02	**7.28E-05**
	average	9.63E-01	4.62E-04	6.68E-01	6.37E-05	9.42E-01	3.40E-05	9.29E-01	1.42E-04	6.38E-01	**3.37E-05**
	median	9.88E-01	1.08E-04	6.86E-01	**0.00E+00**	9.58E-01	**0.00E+00**	9.46E-01	**0.00E+00**	6.54E-01	**0.00E+00**
250	min	9.60E-01	**0.00E+00**	6.40E-01	**0.00E+00**	8.56E-01	**0.00E+00**	8.90E-01	**0.00E+00**	6.06E-01	**0.00E+00**
	max	1.00E+00	**2.68E-04**	7.00E-01	4.18E-04	9.82E-01	3.02E-04	1.00E+00	5.06E-04	7.00E-01	3.55E-04
	std	1.33E-02	1.03E-04	2.06E-02	1.32E-04	3.54E-02	**9.54E-05**	4.13E-02	1.64E-04	3.35E-02	1.12E-04
	average	9.91E-01	6.06E-05	6.84E-01	6.10E-05	9.20E-01	**3.02E-05**	9.73E-01	9.26E-05	6.63E-01	3.63E-05
	median	9.98E-01	**0.00E+00**	6.92E-01	**0.00E+00**	9.20E-01	**0.00E+00**	9.97E-01	**0.00E+00**	6.76E-01	**0.00E+00**

Values in bold indicate the best (lowest) deviation rate in the corresponding row across all methods.

Table 15. Pairwise Wilcoxon signed-rank p-values on the BE suite ($N = 80$).

	SA-SPVAR	TS-Backbone	GA with E/R	SA with E/R	TS with E/R
SA-SPVAR	-	**1.28E-06**	**7.98E-07**	**1.98E-04**	**1.13E-06**
TS-Backbone	**1.28E-06**	-	4.32E-01	**7.36E-04**	6.13E-01
GA with E/R	**7.98E-07**	4.32E-01	-	**5.52E-05**	3.47E-01
SA with E/R	**1.98E-04**	**7.36E-04**	**5.52E-05**	-	**8.35E-05**
TS with E/R	**1.13E-06**	6.13E-01	3.47E-01	**8.35E-05**	-

Values in bold indicate a statistically significant difference at the 0.05 level.

e to a zero mean. Its sole GKA weakness is set b, where only TS with E/R reaches a zero mean. In the pairwise tests, GA with E/R is significantly better than SA-SPVAR on all three suites and than SA with E/R on GKA and BE. Against TS-Backbone, it is significantly better on Beasley but statistically tied on GKA and BE suites. TS-Backbone shows the most uneven profile: it is competitive on the comparatively regular BE instances, yet degrades markedly on the Beasley $n = 500$ group and the dense GKA set f. Controlling for the search engine separates the reduction rule from the host algorithm: E/R within Simulated Annealing is significantly better than SPVAR on every suite (i.e., $p = 3.92\text{E-03}$, 7.96E-04, and 1.98E-04 on Beasley, GKA, and BE), while E/R and Backbone within Tabu Search stay statistically tied on every suite (i.e., $p = 1.88\text{E-01}$, 6.01E-01, and 6.13E-01). Thus the entropy rule matches the backbone rule and improves on frequency-based persistence, independently of the host metaheuristic.

To complement the per-suite analyses with a single test over the entire suites, we pooled the per-instance mean deviation rates of all 165 QUBO instances (i.e., 40 Beasley, 45 GKA, 80 BE) and applied the Friedman test [12] with the five algorithms as treatments, following the recommendations for comparing multiple approaches over multiple problems [9, 10]. The test rejects the null hypothesis of equal performance with $\chi_F^2 = 104.49$, $p = 1.1 \times 10^{-21}$, and the Iman–Davenport correction [19] gives $F = 30.85$ with $(4, 656)$ degrees of freedom ($p \approx 1.5 \times 10^{-23}$). The resulting average ranks order the methods as follows: GA with E/R (2.56), TS with E/R (2.88), TS-Backbone (2.96), SA with E/R (3.01), and SA-SPVAR (3.60). Figure 12 shows the corresponding critical-difference (CD) diagram [9] (Nemenyi post-hoc [33], CD $= 0.47$ at the 0.05 level): SA-SPVAR is separated from every other method, whereas the four remaining methods all fall within one critical distance and are therefore not distinguished by this conservative test. The more powerful pairwise Holm-corrected [18] Wilcoxon signed-rank procedure resolves this top group further: GA with E/R is significantly better than each of the other four methods with all Holm-adjusted $p < 5 \times 10^{-3}$, SA-SPVAR is significantly lower than all, and the middle ranking approaches (i.e., TS with E/R, SA with E/R, TS-Backbone) is mutually indistinguishable. This confirms the suite-by-suite picture: GA with E/R is the most reliable method across the 165 QUBO instances, while SA-SPVAR is consistently the lowest performing approach. We emphasize that this ranking is rank-based: GA with E/R leads because it is rarely over-performed on any individual instance, whereas its single large residual on GKA set b increases its overall mean deviation rate above that of TS with E/R, which attains the lowest overall mean, and TS-Backbone. The two views are complementary, the rank-based tests rewarding consistency and the mean penalizing any single large gap.

Figure 12. Critical-difference diagram of the average Friedman ranks over all 165 QUBO instances. Methods connected by a bold bar are not significantly different under the Nemenyi post-hoc test (CD $= 0.47$, $\alpha = 0.05$).

5. Conclusion

In this paper we presented a reduction operator based on entropy for optimization problems with binary variables, we call it Entropy-based Reduction (E/R). The suggested operator can be incorporated within the design of heuristic algorithms, it measures the empirical entropy of each decision variable along the search and fixes those whose uncertainty drops below a predefined threshold, thus reducing the dimension of the problem online. Here, we presented two E/R employment approaches: (1) pre-process reduction and (2) interactive reduction. Furthermore, we extended the experimental study with a comparison against two state-of-the-art variable-fixing approaches, namely the frequency-based SPVAR and the solution-intersection Backbone, considering both the Knapsack Problem and standard Quadratic Unconstrained Binary Optimization benchmark instances. The obtained results indicate the advantages of the suggested reduction operator over the native heuristic approach and established state-of-the-art approaches. However, some open directions must be investigated as perspectives of this work, including: an adaptive tuning of the uncertainty threshold α, here drawn uniformly at random from $[0.1, 0.3]$ rather than tuned per instance, the extension of the E/R operator to explicitly constrained binary problems such as Max-k-Cut and penalized QUBO formulations.

Acknowledgement

The authors are grateful to the anonymous reviewers for their valuable time and constructive comments. This work has been supported by the Directorate-General for Scientific Research and Technological Development (DGRSDT).

References

[1] BALAS, E., AND ZEMEL, E. An algorithm for large zero-one knapsack problems. *Operations Research 28*, 5 (1980), 1130–1154.

[2] BEASLEY, J. E. Heuristic algorithms for the unconstrained binary quadratic programming problem. Tech. rep., The Management School, Imperial College London, 1998.

[3] BILLIONNET, A., AND ELLOUMI, S. Using a mixed integer quadratic programming solver for the unconstrained quadratic 0-1 problem. *Mathematical Programming 109*, 1 (2007), 55–68.

[4] BOTEV, Z. I., KROESE, D. P., RUBINSTEIN, R. Y., ET AL. The cross-entropy method for optimization. In *Handbook of statistics*, vol. 31. Elsevier, 2013, pp. 35–59.

[5] BOUROUBI, S., AND BOUMEDINE, N. A new hybrid genetic algorithm for protein structure prediction on the 2d triangular lattice. *Turkish Journal of Electrical Engineering and Computer Sciences 29*, 2 (2021), 499–513.

[6] CASERTA, M., AND NODAR, M. C. A cross entropy based algorithm for reliability problems. *Journal of Heuristics 15* (2009), 479–501.

[7] CASERTA, M., RICO, E. Q., AND URIBE, A. M. A cross entropy algorithm for the knapsack problem with setups. *Computers & Operations Research 35*, 1 (2008), 241–252.

[8] CINQUE, L., DE AGOSTINO, S., AND LOMBARDI, L. Binary image compression via monochromatic pattern substitution: sequential and parallel implementations. *Mathematics in Computer Science 7* (2013), 155–166.

[9] DEMŠAR, J. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research 7* (2006), 1–30.

[10] DERRAC, J., GARCÍA, S., MOLINA, D., AND HERRERA, F. A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm and Evolutionary Computation 1*, 1 (2011), 3–18.

[11] DUKKANCI, O., CAMPBELL, J. F., AND KARA, B. Y. Facility location decisions for drone delivery with riding: A literature review. *Computers & Operations Research* (2024), 106672.

[12] FRIEDMAN, M. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the American Statistical Association 32*, 200 (1937), 675–701.

[13] FUERTES, G., VARGAS, M., ALFARO, M., ET AL. Chaotic genetic algorithm and the effects of entropy in performance optimization. *Chaos: An Interdisciplinary Journal of Nonlinear Science 29*, 1 (2019).

[14] GLOVER, F., KOCHENBERGER, G., AND ALIDAEE, B. Tabu search and grasp algorithms for the unconstrained binary quadratic programming problem. *Journal of Heuristics 4*, 3 (1998), 281–304.

[15] GRAY, R. M. *Entropy and information theory*. Springer Science & Business Media, 2011.

[16] GUAN, B., ZHAO, Y., AND LI, Y. An ant colony optimization based on information entropy for constraint satisfaction problems. *Entropy 21*, 8 (2019), 766.

[17] HAYNES, T. W., HEDETNIEMI, S., AND SLATER, P. *Fundamentals of domination in graphs*. CRC press, 2013.

[18] HOLM, S. A simple sequentially rejective multiple test procedure. *Scandinavian Journal of Statistics 6*, 2 (1979), 65–70.

[19] IMAN, R. L., AND DAVENPORT, J. M. Approximations of the critical region of the Friedman statistic. *Communications in Statistics – Theory and Methods 9*, 6 (1980), 571–595.

[20] JIACHENG, L., AND LEI, L. A hybrid genetic algorithm based on information entropy and game theory. *IEEE Access 8* (2020), 36602–36611.

[21] KANTOUR, N., AND BOUROUBI, S. Cryptanalysis of merkle-hellman cipher using parallel genetic algorithm. *Mobile Networks and Applications 25*, 1 (2020), 211–222.

[22] KARIMI, H., AND ROSENBERG, G. Boosting quantum annealer performance via sample persistence. *Quantum Information Processing 16*, 7 (2017), 166.

[23] KARIMI, H., ROSENBERG, G., AND KATZGRABER, H. G. Effective optimization using sample persistence: A case study on quantum annealers and various monte carlo optimization methods. *Physical Review E 96*, 4 (2017), 043312.

[24] KARP, R. M. Reducibility among combinatorial problems. In *Complexity of computer computations* (1972), Springer, pp. 85–103.

[25] KELLERER, H., PFERSCHY, U., AND PISINGER, D. *Knapsack Problems*. Springer, 2004.

[26] KOCHENBERGER, G., HAO, J.-K., GLOVER, F., LEWIS, M., LÜ, Z., WANG, H., AND WANG, Y. The unconstrained binary quadratic programming problem: a survey. *Journal of combinatorial optimization 28*, 1 (2014), 58–81.

[27] KUMAR, S., MALLIK, A., AND PANDA, B. Influence maximization in social networks using transfer learning via graph-based lstm. *Expert Systems with Applications 212* (2023), 118770.

[28] LAGUNA, M., DUARTE, A., AND MARTÍ, R. Hybridizing the cross-entropy method: An application to the max-cut problem. *Computers & Operations Research 36*, 2 (2009), 487–498.

[29] Lai, C.-M., Yeh, W.-C., and Huang, Y.-C. Entropic simplified swarm optimization for the task assignment problem. *Applied Soft Computing 58* (2017), 115–127.

[30] Lorie, J. H., and Savage, L. J. Three problems in rationing capital. *The Journal of Business 28*, 4 (1955), 229–239.

[31] Ma, D., Wang, S., and Zhang, Z. Hybrid algorithm of minimum relative entropy-particle swarm optimization with adjustment parameters for gas source term identification in atmosphere. *Atmospheric Environment 94* (2014), 637–646.

[32] Martello, S., and Toth, P. *Knapsack Problems: Algorithms and Computer Implementations*. John Wiley & Sons, 1990.

[33] Nemenyi, P. B. *Distribution-free multiple comparisons*. PhD thesis, Princeton University, 1963.

[34] Olivares, R., Soto, R., Crawford, B., et al. Entropy–based diversification approach for bio–computing methods. *Entropy 24*, 9 (2022), 1293.

[35] Palubeckis, G. Multistart tabu search strategies for the unconstrained binary quadratic optimization problem. *Annals of Operations Research 131*, 1 (2004), 259–282.

[36] Palubeckis, G. Iterated tabu search for the unconstrained binary quadratic optimization problem. *Informatica 17*, 2 (2006), 279–296.

[37] Papadimitriou, C. H., and Steiglitz, K. *Combinatorial Optimization: Algorithms and Complexity*. Prentice-Hall, 1982.

[38] Pisinger, D. An expanding-core algorithm for the exact 0–1 knapsack problem. *European Journal of Operational Research 87*, 1 (1995), 175–187.

[39] Rendl, F., Rinaldi, G., and Wiegele, A. Solving max-cut to optimality by intersecting semidefinite and polyhedral relaxations. *Mathematical Programming 121*, 2 (2010), 307–335.

[40] Saroj, K., Kumar, D., et al. A genetic algorithm with entropy based probabilistic initialization and memory for automated rule mining. In *Advances in Computer Science and Information Technology: First International Conference, CCSIT 2011, Bangalore, India, January 2-4, 2011. Proceedings, Part I 1* (2011), Springer Berlin Heidelberg, pp. 604–613.

[41] Shylo, O. V., and Shams, H. Boosting binary optimization via binary classification: A case study of job shop scheduling. *arXiv preprint arXiv:1808.10813* (2018).

[42] Spillman, R. Cryptanalysis of knapsack ciphers using genetic algorithms. *Cryptologia 17*, 4 (1993), 367–377.

[43] Stegherr, H., Heider, M., and Hähner, J. Classifying metaheuristics: Towards a unified multi-level classification system. *Natural Computing 21*, 2 (2022), 155–171.

[44] Subbotin, S., and Oleynik, A. Entropy based evolutionary search for feature selection. In *2007 9th International Conference-The Experience of Designing and Applications of CAD Systems in Microelectronics* (2007), IEEE, pp. 442–443.

[45] Talbi, E.-G. *Metaheuristics: from design to implementation*. John Wiley & Sons, 2009.

[46] Tijjani, S., Ab Wahab, M. N., and Noor, M. H. M. An enhanced particle swarm optimization with position update for optimal feature selection. *Expert Systems with Applications 247* (2024), 123–337.

[47] Wang, Y., Lü, Z., Glover, F., and Hao, J.-K. Backbone guided tabu search for solving the ubqp problem. *Journal of Heuristics 19*, 4 (2013), 679–695.

[48] Wiegele, A. Biq mac library—a collection of max-cut and quadratic boolean optimization instances. Alpen-Adria-Universität Klagenfurt, 2007. Available at http://biqmac.uni-klu.ac.at/, accessed 2026.

[49] Wilcoxon, F. Individual comparisons by ranking methods. *Biometrics bulletin 1*, 6 (1945), 80–83.

[50] Wolpert, D. H., and Macready, W. G. No free lunch theorems for optimization. *IEEE transactions on evolutionary computation 1*, 1 (1997), 67–82.

[51] Wolpert, D. H., Macready, W. G., et al. No free lunch theorems for search. Tech. rep., Technical Report SFI-TR-95-02-010, Santa Fe Institute, 1995.

[52] Zhang, H., Xie, J., Ge, J., et al. An entropy-based pso for dar task scheduling problem. *Applied Soft Computing 73* (2018), 862–873.

[53] Zhang, W. Configuration landscape analysis and backbone guided tabu search. *Artificial Intelligence 158*, 1 (2004), 1–26.

[54] Zhang, X., and Zhang, X. Cross entropy method meets local search for continuous optimization problems. *International Journal on Artificial Intelligence Tools 26*, 06 (2017), 1750020.

[55] Zou, J., Zheng, J., and Shen, R. A cooperative swarm intelligence algorithm for multi-objective discrete optimization with application to the knapsack problem. *Applied Soft Computing 11*, 8 (2011), 5263–5282.