Key Principles in Cross-Domain Hyper-Heuristic Performance

Václav Sobotka[1], Lucas Kletzander[2], Nysret Musliu[2], and Hana Rudová[1]

sobotka@mail.muni.cz, lucas.kletzander@tuwien.ac.at,
nysret.musliu@tuwien.ac.at, hanka@fi.muni.cz

[1] Faculty of Informatics, Masaryk University, Brno, Czech Republic
[2] Christian Doppler Laboratory for Artificial Intelligence and Optimization for Planning and Scheduling, DBAI, TU Wien, Austria

Abstract

Cross-domain selection hyper-heuristics aim to distill decades of research on problem-specific heuristic search algorithms into adaptable general-purpose search strategies. In this respect, existing selection hyper-heuristics primarily focus on an adaptive selection of low-level heuristics (LLHs) from a *predefined* set. In contrast, we concentrate on the composition of this set and its strategic transformations. We systematically analyze transformations based on three key principles: solution acceptance, LLH repetitions, and perturbation intensity, i.e., the proportion of a solution affected by a perturbative LLH. We demonstrate the raw effects of our transformations on a trivial unbiased random selection mechanism. With an appropriately constructed transformation, this trivial method outperforms all available state-of-the-art hyper-heuristics on three challenging real-world domains and finds 11 new best-known solutions. The same method is competitive with the winner of the CHeSC competition, commonly used as the standard cross-domain benchmark. Moreover, we accompany several recent hyper-heuristics with such strategic transformations. Using this approach, we outperform the current state-of-the-art methods on both the CHeSC benchmark and real-world domains while often simplifying their designs.

1 Introduction

Metaheuristic search strategies Gendreau and Potvin [2010] are the cornerstone of methodologies tackling a vast range of combinatorial optimization problems. While metaheuristics address issues common to all search algorithms, such as local optima evasion or balancing exploitation and exploration in the process, they still serve as templates for implementations of problem-specific algorithms. In contrast, selection hyper-heuristics Drake et al. [2020] aim to provide domain-agnostic search strategies. Given a set of low-level heuristics (LLHs), hyper-heuristics aim to steer the search process by adaptively selecting from the available LLHs, often using information from previous iterations of the search process. Research on hyper-heuristics has attracted attention, arguably due to the appealing idea of generalizing and consolidating decades of research on problem-specific approaches and metaheuristics into general domain-agnostic methods. As a result, a wide range of adaptive and learning LLH selection mechanisms has emerged.

Interestingly, the vast majority of works on hyper-heuristic methods assume a fixed set of LLHs is provided and use it as is. In this paper, we challenge this approach by transparently transforming existing LLH sets into new *virtual LLH sets*. Our virtual LLH sets effectively modify the original LLHs in three fundamental aspects: solution acceptance, repeated LLH applications, and perturbation intensity. By running existing hyper-heuristics on top of properly designed virtual LLH sets, we systematically obtain significant cross-domain performance benefits. With this approach, we demonstrate substantial improvements for the majority of the available recent hyper-heuristics, outperforming the current state-of-the-art hyper-heuristics on both the standard CHeSC cross-domain benchmark Burke et al. [2011] and three challenging real-world application domains. Critically, we exemplify the raw effects of the aforementioned three key principles on a trivial random unbiased selection mechanism. We show that the trivial selection mechanism, accompanied by a strategically transformed LLH set, outperforms all tested state-of-the-art hyper-heuristics on three real-world domains and provides performance comparable to the CHeSC-winning hyper-heuristic Mısır et al. [2012] on the standard CHeSC benchmark. To summarize, the key contributions of our paper are:

- We identify and analyze the three aforementioned key principles, use them to transparently transform existing LLH sets, and demonstrate their critical impacts on cross-domain search performance.
- We demonstrate that solely the three key principles accompanying a trivial selection mechanism are enough to obtain results comparable to or even outperforming recent hyper-heuristics. This finding strongly contrasts with the general focus on the LLH selection mechanisms.
- Using only the trivial selection mechanism and the three key principles, we obtain 11 new best-known solutions on two challenging real-world domains.
- By transforming existing LLH sets, we significantly improve the cross-domain performance for the majority of the available recent hyper-heuristics. We demonstrate the benefits of our methodology on both standard benchmarks and three real-world domains.

2 Related works

Hyper-heuristics aim to provide high-level search strategies. Previously, comprehensive reviews and classifications of existing hyper-heuristics were provided in Burke et al. [2013], Drake et al. [2020], and most recently by Dokeroglu et al. [2024]. In terms of the standard classification Burke et al. [2019], we concentrate on *perturbative selection hyper-heuristics*. Such hyper-heuristics operate *online* by progressively selecting the LLHs to be applied and adapting this selection based on the past search trajectory. A large body of research concentrating on this type of hyper-heuristics is centered around the Cross-Domain Heuristic Search Challenge 2011 (CHeSC) Burke et al. [2011]. The competition introduced the HyFlex framework Ochoa et al. [2012], providing a diverse benchmark implementing six search domains with a standardized interface and predefined LLH sets for each domain. The CHeSC competition-winning hyper-heuristic GIHH Mısır et al. [2012] combines a large number of adaptive mechanisms. It was followed by a self-adaptive variable-neighborhood search hyper-heuristic Hsiao et al. [2012], and a method using intensification-diversification cycles with reinforcement learning mechanisms Larose [2011]. Overall, there was a total of 20 teams competing in CHeSC, resulting in a collection of well-evaluated hyper-heuristic concepts and mechanisms. Notably, a recent meta-study Razali et al. [2025] systematically reviews all hyper-heuristics competing in CHeSC, comprehensively analyzing several key design decisions and their impact on the methods' performance.

Among the hyper-heuristics developed after CHeSC, we observe several trends. First is the aim for design simplicity. In Adriaensen and Nowé [2016] (LGIHH), the original GIHH algorithm was simplified by analyzing its mechanisms and eliminating the unnecessary ones. The simplified algorithm was shown to outperform the original GIHH algorithm. With a similar motivation, Adriaensen et al. [2014] (FSILS) introduced a conceptually simple algorithm combining an iterated local search (ILS) scheme with time normalization, acceptance, and restart mechanisms. FSILS is shown to outperform GIHH with a strikingly simpler design and a well-documented role of all applied mechanisms. Subsequently, the key design decisions of FSILS were later used in Adubi et al. [2021] (TSILS) with adaptive mechanisms combined with Thompson sampling LLH selection, and in Adubi et al. [2022] (EAILS), where evolutionary mechanisms steer the combination of perturbative LLHs. To the best of our knowledge, the results of TSILS on the CHesC benchmark form the current state-of-the-art. The last notable trend is the employment of learning, often based on reinforcement learning techniques Sutton and Barto [2018] (RL). In Sabar et al. [2014] (GEPHH), gene expression programming is used to automatically select LLHs and acceptance mechanisms. The Monte Carlo tree search scheme with multi-armed bandit principles was used in Sabar and Kendall [2015] (MCTS) and Ferreira et al. [2015] (FRAMAB) to steer the search trajectory. Later, both Choong et al. [2018] (QHH) and Mischek and Musliu [2022] (MC) used Q-learning-based selection of LLHs. Most recently, Kletzander and Musliu [2023] (LASTRL) combined ILS with adaptive RL strategies and rich state representation.

We separately emphasize Chuang [2020] (LUBY) using Luby sequence restarts Luby et al. [1993] in an automated synthesis of search strategies. To the best of our knowledge, this is the only existing work proposing an LLH set transformation. The described *domain amplification* doubles the LLH set size by adding an "amplified" duplicate for each LLH. The amplified LLHs execute the original LLH for 10 ms while rejecting non-improving solutions. Interestingly, this exact concept was later used in Mischek and Musliu [2022] with substantial performance benefits. Still, the only available description of this technique known to us [Chuang, 2020, p. 40-41] provides minimal justification of its design. The work motivates the repeated LLH applications with the aim to "promote collaboration among heuristics". Yet, if worse solutions are always discarded, such collaboration and the ability to escape local optima are vastly limited. Therefore, we cover the described gap by systematically examining LLH set transformations that introduce solution acceptance and repeated LLH applications (as in domain amplifications), and further add a third important principle, perturbation intensity. Table 1 concludes our review by summarizing all post-CHeSC methods focusing on these three principles and design features strongly influencing methods' exploration-exploitation balance. We later compare against all methods in Table 1.

Work	*Acceptance mechanism*	*LLH chaining*	*Perturbation intensity*	*LLH selection bias*	*Restarts*
LASTRL	All	LS-biased chains	Static	RL, ILS	Luby, Full
MC	All, Discard worse	Repeat until timeout	Static	RL	Luby
EAILS	μ-norm Metropolis	LS chains	Adaptive	EA-learned ILS	–
TSILS	μ-norm Metropolis	LS chains	Adaptive	Double shaking ILS	–
LUBY	All, Discard worse	Repeat until timeout	Static	Random	Luby
QHH	μ-norm Metropolis + others	LS chains	Not stated	RL, ILS	-
FRAMAB	Monte Carlo	–	Adaptive	Multi-armed bandit	-
LGIHH	AILLA	Relay hybridization	Adaptive	Adaptive LLH selection	Full
MCTS	Monte Carlo	–	Multiple static	Multi-armed bandit	–
FSILS	μ-norm Metropolis	LS chains	Static	ILS	Full
GEPHH	Evolving function	–	Not stated	Evolutionary	–

Table 1: Key design mechanisms affecting the exploration-exploitation balance in recent cross-domain hyper-heuristics.

3 LLH set transformation framework

Our work implements and publishes the proposed LLH set transformation methodology as an extension module of HyFlex Ochoa et al. [2012], a commonly used framework for the development and benchmarking of cross-domain selection hyper-heuristics. Thus, we frame the descriptions of our methodology using interfaces and terms tied to HyFlex. In HyFlex, a *problem domain* must specify a set of LLHs. Furthermore, the domain maintains a number of solutions in addressable *solution registers*. The stored solutions can be copied between the registers and inspected for solution costs. The hyper-heuristics then apply the available LLHs to these solutions and manipulate the solutions in the registers to steer the search trajectory. Further, the hyper-heuristics may instruct the domain object to alter its parameter controlling the perturbation intensity. From the hyper-heuristic point of view, the only available information about each LLH is its unique identification and membership in one of the following four categories. *Local search* (LS) LLHs are typically classical neighborhood-based search moves with the additional guarantee that their application to a solution returns a solution of equal or better quality. *Ruin&recreate* (RR) LLHs "ruin" the solution first, making it a partial solution. Then, the partial solution is "recreated", typically in a greedy manner. *Mutation* (MUT) LLHs only aim to introduce random changes to the solution. *Crossover* (XO) LLHs combine two existing solutions into a new one. Neither RR, MUT, nor XO LLHs provide solution quality guarantees. Often, the RR and MUT categories are jointly called *perturbative LLHs*. The LLH set and its division into these four categories form the input to our transformation procedures.

The key idea of our approach is to transform an original LLH set into a new virtual LLH set populated with *virtual LLHs* and let existing hyper-heuristics transparently operate on top of it. Our transformations target the three previously discussed principles, and we apply the transformations at the level of individual LLH categories. Within the virtual LLH set, each virtual LLH is based on one LLH from the original set (attribute LLH). The virtual LLH attributes ACCEPT, DURATION, and INTENSITY then describe the modifying effects in terms of the three principles as per the applied transformation. Lastly, we note that it is possible to apply several transformations on one LLH category, as we commonly do to obtain several LLH duplicates with different perturbation intensity. Figure 1 shows how the three modifiers take effect when applying a virtual LLH. On the highest level, the procedure APPLYVIRTUALLLH keeps an interface identical to the standard LLH application in HyFlex. The virtual LLH H is applied on a solution stored in the source solution register s^*, the resulting solution is saved in the target solution register s^{**}, and the cost of this new solution is returned. The repeated application of the wrapped LLH linked to the DURATION attribute is realized by means of the while loop on line 4. This aspect allows for executing the given original LLH for a certain amount of time repeatedly. Modification in terms of perturbation intensity related to the INTENSITY attribute is realized on lines 3 and 13. Line 3 instructs the domain to execute perturbative LLHs in the main loop with the intensity required by the virtual LLH (if applicable for the LLH type). At the end of the virtual LLH application, the original perturbation intensity is restored at line 13.

The most involved modifier is the acceptance strategy provided in the ACCEPT attribute. Its key role is to discard new solutions after the LLH application if their quality is deemed insufficient by the standards of the given strategy. This allows for keeping the search trajectory within promising parts of the search space. In order to accommodate this modifier, we introduce two additional solution registers, s_{cur} and s_{new}. The register s_{cur} holding the current solution to search from is initialized on

```
1: procedure ApplyVirtualLLH($D, H, s^*, s^{**}$)
   $D$          HyFlex problem domain
   $H$          virtual LLH
   $s^*$        source solution register ID
   $s^{**}$     target solution register ID
   $\boldsymbol{\mu}$          mean improvement statistic (global)
   $\boldsymbol{n_{imp}}$      number of improvements statistic (global)
2:   $D$.CopySolution($s^*$, $s_{\text{cur}}$)
3:   $D$.SetPerturbationIntensity($H$.Intensity)
4:   while $H$.Duration timeout not exceeded do
5:     $c_{\text{best}} \leftarrow D$.GlobalBestCost()
6:     $c_{\text{cur}} \leftarrow D$.Cost($s_{\text{cur}}$)
7:     $c_{\text{new}} \leftarrow D$.ApplyLLH($H$.LLH, $s_{\text{cur}}, s_{\text{new}}$)
8:     if $c_{\text{new}} < c_{\text{cur}}$ then
9:       $\boldsymbol{n_{\text{imp}}} \leftarrow \boldsymbol{n_{\text{imp}}} + 1$
10:      $\boldsymbol{\mu} \leftarrow \boldsymbol{\mu} + (c_{\text{cur}} - c_{\text{new}} - \boldsymbol{\mu})/\boldsymbol{n_{\text{imp}}}$
11:    if $H$.Accept($\boldsymbol{\mu}$, $c_{\text{best}}$, $c_{\text{cur}}$, $c_{\text{new}}$) then
12:      $D$.CopySolution($s_{\text{new}}$, $s_{\text{cur}}$)
13:  $D$.RestorePerturbationIntensity()
14:  $D$.CopySolution($s_{\text{cur}}$, $s^{**}$)
15:  return $D$.Cost($s^{**}$)
```

Figure 1: Application of virtual LLH.

line 2 by a copy of the solution in s^*. Then, line 7 inside the repetition loop applies the original LLH to this solution and saves the new solution to s_{new}. The acceptance strategy then decides whether the solution in s_{new} is of sufficient quality to be copied to s_{cur} on lines 11 and 12, or whether the solution in s_{new} shall be discarded. Ultimately, the last accepted solution from s_{cur} is copied to the target solution register s^{**} on line 14 as the result of the virtual LLH application, and its quality is returned on line 15. We note that, in general, the acceptance strategy decisions are based on the qualities of the globally best-so-far (line 5), current (line 6), and new (line 7) solutions. The last input to the acceptance decision is the global *mean improvement* statistic denoted as $\boldsymbol{\mu}$. This statistic is critical for the search acceptance strategies to be able to operate under the cross-domain context. This global statistic is initially set to 0 and then continuously updated by means of lines 8 to 10.

LLH transformation principles

Next, we detail and motivate the three key principles and propose novel cross-domain acceptance strategies utilizing the $\boldsymbol{\mu}$ statistic.

Solution acceptance

Acceptance methods such as threshold acceptance Dueck and Scheuer [1990] (TA), record-to-record travel Dueck [1993] (R2R), or Metropolis acceptance Metropolis et al. [1953] (MA) are common search-steering mechanisms in the problem-specific context. However, these methods face three critical issues in the context of cross-domain search. First, (1) the ranges and granularity of objective function values vastly differ across problem domains, and (2) across instances inside the individual domains. Moreover, (3) the objective function values within one search run may differ by several orders of magnitude. To the best of our knowledge, FSILS is the only work explicitly describing at least the points (1) and (2) as fundamental issues in cross-domain search. Importantly, FSILS proposes MA with a modification resolving not only the issues (1) and (2) described in their paper, but also the issue (3). The modification's key point is the normalization of the solution qualities using the mean improvement statistic μ. We adopt this μ-normalization technique with the difference that we update the statistic after every improvement, compared to the sparser updates at the end of LS chains in FSILS, resulting in more precise estimates. Critically, instead of just copying the FSILS design and μ-normalized MA as is common (see Table 1), we observe that μ-normalization solves a general cross-domain issue and can therefore be used with other common acceptance methods, not only with MA. Consequently, we propose novel cross-domain acceptance mechanisms by μ-normalizing standard problem-specific TA, R2R, and MA acceptances, all three with both constant (CONST) and exponential (EXP) threshold cooling schedules.

Equations 1 to 3 describe the acceptance criteria MA, TA, and R2R, respectively. τ is their threshold parameter. $UNIFORM(0,1)$ denotes a random choice from a uniform distribution on the interval 0 to 1. We note that in all strategies, strictly improving solutions are always accepted.

$$UNIFORM(0,1) < e^{\frac{c_{\text{cur}} - c_{\text{new}}}{\tau\mu}} \tag{1}$$

$$c_{\text{new}} \le c_{\text{cur}} + \tau\mu \tag{2}$$

$$c_{\text{new}} \le c_{\text{best}} + \tau\mu \tag{3}$$

The CONST variants set τ to a fixed value. The EXP variants decrease τ over time based on two initial parameters, τ_{start} and τ_{end}. We fit an exponential function f of base e such that $f(0) = \tau_{\text{start}}$ and $f(1) = \tau_{\text{end}}$. During the search, the effective τ is calculated as $f(x)$ where $x \in [0;1]$ is the proportion of already consumed search budget. We note that the CONST variant of MA matches the acceptance from FSILS.

LLH repetitions

We use the timeout-based repetitions introduced in LUBY for two reasons. First, with our proposed cross-domain acceptance mechanisms, the original (overly aggressive) discarding of worse solutions in domain amplifications can be replaced, allowing for the formerly advertised "collaboration among heuristics". Second, we observe that replacing the original LLHs such that *all* new LLHs have the *same* (high-enough) repetition timeout implicitly normalizes the computational resources allocated to individual LLHs (if no other bias is present). This contrasts with the original domain amplifications that double the LLH set by keeping the original LLHs. We note that projecting the speed of LLHs into their sampling probabilities has been shown as beneficial, e.g., in the *SpeedNew* mechanism from FSILS.

Perturbation intensity

Based on the recent meta-study of the hyper-heuristics competing in CHeSC Razali et al. [2025], 8 of the 20 methods use a static setting of the perturbative LLH intensity parameter. Similarly, the post-CHeSC methods summarized in Table 1 also often do not work with this parameter. At the same time, one of the conclusions of the aforementioned meta-study is that the static parameter setting is the only parameter control scheme inferior to other alternatives that are rather interchangeable in terms of performance. Here, our observations suggest that the critical point is primarily the ability to vary the perturbation intensity. Therefore, we propose duplication of perturbative LLHs based on a predefined set of variable intensity settings. This transformation effectively offers perturbations of varying intensities with the possibility to bias the sampling probabilities of more/less aggressive perturbations. Moreover, such a transformation can be used as an arguably simpler replacement for existing parameter adaptation mechanisms.

4 Experiments

Now, we experimentally demonstrate the benefits of strategic LLH set transformations. The first part of the experiments is based on the standard CHeSC benchmark. For 3 hyper-heuristics, we gradually construct complete LLH set transformations by sequentially transforming solution acceptance, LLH repetitions, and perturbation intensities. Further, we improve 4 additional hyper-heuristics by transforming only the intensity of perturbations, effectively replacing their internal intensity adaptation mechanisms. In the second part of the experiments, we take the developed transformations and show their generality on three real-world domains.

Methods

The 3 hyper-heuristic strategies that we gradually built upon are the following. First, we concentrate on the recent MC hyper-heuristic utilizing Q-learning LLH selection, Luby sequence restarts, and domain amplifications. Second, we inspect the closely related hyper-heuristic LUBY using only the Luby sequence restarts, domain amplifications, and uniform random selection of LLHs. Lastly, we introduce a baseline naive hyper-heuristic (NHH) that first uniformly randomly selects an LLH category (LS, RR, or MUT) and then uniformly randomly selects an LLH within this category, while always accepting new solutions. For method $\mathrm{X} \in \{\mathrm{MC}, \mathrm{LUBY}, \mathrm{NHH}\}$, we use the notation X^{+} and X^{0} to explicitly distinguish variants of X with domain amplification and without it. This choice of methods allows us to (1) assess raw effects of the transformations while avoiding interactions with common design biases using NHH, (2) improve MC as a non-trivial learning-based method, and (3) inspect LUBY as its logical subset, dropping MC's learning component. Furthermore, MC and LUBY are the only existing methods using the original domain amplifications, allowing for their direct comparison with our LLH set transformations. The 4 hyper-heuristics where we transform only the perturbation intensities are LGIHH, FSILS, TSILS, and EAILS. Lastly, we note that we also compare with FRAMAB, LASTRL, QHH, MCTS, and GEPHH on the CHeSC benchmark based on the results reported by the authors. In case of NHH, MC, LUBY, TSILS, EAILS, FSILS, LGIHH, and all their derived variants, we provide our own reevaluations under the same conditions.

CHeSC benchmark experiments

We use the standard cross-domain CHeSC benchmark and the connected HyFlex framework Ochoa et al. [2012] implementing 6 diverse search domains. Namely, the domains

	Variant	*Acceptance*	$\tau_{\min}$	$\tau_{\max}$	τ_{step}	τ_{end}
NHH	CONST	R2R	1.0	6.0	1.25	–
		MA	0.25	1.25	0.25	–
		TA	0.25	1.25	0.25	–
	EXP	R2R	2.5	7.5	1.25	1.0
		MA	0.5	1.5	0.25	0.25
		TA	0.5	1.5	0.25	0.25
MC+LUBY	CONST	R2R	1.0	6.0	1.25	–
		MA	0.25	2.25	0.5	–
		TA	1.0	2.0	0.25	–
	EXP	R2R	2.5	12.5	2.5	1.0
		MA	0.75	2.75	0.5	0.25
		TA	1.25	2.25	0.25	1.0

Table 2: Summary of parameter scales (5 values) for each combination of hyper-heuristic, acceptance, and variant.
CONST: τ from $\tau_{\min}$ to $\tau_{\max}$ with a step τ_{step}.
EXP: τ_{start} from $\tau_{\min}$ to $\tau_{\max}$ with a step τ_{step}. τ_{end} is fixed.

are Maximum Satisfiability, Bin Packing, Personnel Scheduling, Flowshop, Travelling Salesman Problem, and Vehicle Routing Problem. Our experiments are based on the 30 instances used in the original competition (5 for each domain), allowing for extensive comparison with existing hyper-heuristics. To quantify the gradual improvements during the sequential construction of LLH set transformations, we compare the methods following the rules and F1 scoring system used in CHeSC. Specifically, we calculate the F1 score for a method by letting it compete against the original results reported for the 20 hyper-heuristics competing in CHeSC. Higher F1 scores reflect better results. We perform 31 repeated evaluations of each of the 30 competition instances. The competition benchmark script allocated 276 seconds to one run in our environment running on *Debian 6.1.135 x86_64* using *AMD EPYC 7543* CPU (2.8 GHz), matching a 10-minute timeout on the original competition machine. The HyFlex code was compiled using *openjdk 11.0.27*. Further details about the domains, LLH sets, instances, and F1 scoring are described in Burke et al. [2011]. Detailed results of these experiments are in Appendix A.

Solution acceptance

We test transforming the LLHs in both RR and MUT categories with one of the 6 cross-domain acceptance strategies, i.e., MA, TA, and R2R, all in both CONST and EXP variants. We test 5 increasingly strict parameterizations for each strategy. The respective parameter ranges were identified with preliminary experiments. Table 2 summarizes the selected acceptance parameters used with NHH^0, LUBY^0, and MC^0. We note that the higher τ values in LUBY^0/MC^0 reflect their regular intensifying restarts to the best-so-far solution missing in NHH. Now, we evaluate the outlined setups and summarize our main observations.

Crucially, all of the 6 acceptance strategies, given a reasonable setting, provide benefits in terms of the method's cross-domain performance. For all NHH, LUBY, and MC, the best acceptance transformations outperform the respective X^0 and X$^+$ setups. Even more, this is the case for a large part of the tested configurations across all 6 acceptance strategies for NHH and LUBY. These results provide strong evidence that (1) a properly set acceptance mechanism is a design element of critical importance for the overall cross-domain performance, and (2) μ-normalization can be successfully used to derive novel cross-domain acceptance strategies as we propose. In relative comparison, we observe the most consistent performance in the R2R EXP strategy. It achieves the best results for MC and LUBY, and the second-best results, with 1 F1 point difference from TA EXP, for NHH. We provide two notable observations regarding R2R EXP. First, R2R strategies establish a global bound on what is (un)acceptable solution quality, contrasting with restrictions on a single acceptance step in MA and TA. Second, the EXP variant introduces an interesting advantage in the cross-domain context. While domains generally differ in their ideal setting of τ, exponential decay of τ ensures that searching under different settings of τ takes place. Interestingly, the experiments also revealed one important weakness of the TA strategy in the cross-domain context. Generally, we observe that the deteriorating steps should be kept smaller than μ, i.e., $\tau < 1.0$, at least for a larger part of the search process. At the same time, some domains (e.g., Maximum Satisfiability) have the inherent property of atomic improvement units (unsatisfied clauses). In case μ is close to 1.0, its combination with $\tau < 1.0$ often leads to premature inability to accept even small qualitative deterioration. Based on the aforementioned observations, we fix the acceptance transformations to R2R EXP with τ_{start} set to 5.0, 7.5, and 10.0 for NHH^{0}, LUBY^{0}, and MC^{0}, respectively. We refer to the resulting methods as NHH$^{\mathrm{A}}$, LUBY$^{\mathrm{A}}$, and MC$^{\mathrm{A}}$.

LLH repetitions

For NHH$^{\mathrm{A}}$, LUBY$^{\mathrm{A}}$, and MC$^{\mathrm{A}}$, we further introduce repeated LLH applications for the categories LS, RR, and MUT. We test 5 distinct repetition timeouts: 0.5, 1, 2.5, 5, and 10 milliseconds. Regarding the results, enforcing repetition timeouts on LLHs further boosted the performance of NHH$^{\mathrm{A}}$, LUBY$^{\mathrm{A}}$, and MC$^{\mathrm{A}}$ in all of the tested repetition timeouts. Generally, shorter timeouts resulted in better benefits. Longer timeouts converge back towards the results of NHH$^{\mathrm{A}}$, LUBY$^{\mathrm{A}}$, and MC$^{\mathrm{A}}$. We also performed an ablation analysis separating the repetitions' effects on the LS category from effects on the perturbative RR and MUT categories. We conclude that both LS and RR+MUT parts separately add to the transformation's performance. However, transforming all LS, RR, and MUT provides better performance than each of the components separately. We fix the best repetition configurations to 0.5 ms for LUBY$^{\mathrm{A}}$ and MC$^{\mathrm{A}}$, and 1 ms for NHH$^{\mathrm{A}}$. We refer to the resulting methods as NHH$^{\mathrm{AR}}$, LUBY$^{\mathrm{AR}}$, and MC$^{\mathrm{AR}}$.

Perturbation intensity

Apart from extending NHH$^{\mathrm{AR}}$, LUBY$^{\mathrm{AR}}$, and MC$^{\mathrm{AR}}$, we also modify (only) the perturbation intensities of LGIHH, TSILS, FSILS, and EAILS. While these methods handle acceptance and repetition aspects reasonably, they either do not handle intensities (FSILS) or use adaptive mechanisms that can be potentially overriden with substantially simpler alternatives (LGIHH, EAILS, TSILS). In the considered transformations, we target LLHs in the RR and MUT categories and duplicate their LLHs in several intensities. Setup I = $[0.1, 0.2...0.9, 1.0]$ uniformly covers the whole parameter scale. Setup *II* = $[0.05, 0.05, 0.05, 0.05, 0.1, 0.1, 0.2, 0.3, 0.5]$ covers the lower half of the parameter scale with an exponential ramp up, prioritizing low intensities. Setup *III* = $[0.1, 0.2, 0.3]$ allows for slight deviations from the HyFlex default 0.2. Setup *Base* refers to X$^{\mathrm{AR}}$ (NHH, LUBY, MC) or X^{0} (LGIHH, FSILS, TSILS, EAILS). The results are summarized in Table 3.

	NHH	LUBY	MC	LGIHH	FSILS	TSILS	EAILS
Base	115	154	164	210	193	220	187
I	94	82	130	**213**	**203**	203	142
II	**154**	143	**192**	**225**	192	210	179
III	**116**	145	**191**	**214**	**202**	**224**	**193**

Table 3: F1 scores for setups *I-III* and *Base* (X^{AR} or X^0) for the tested 7 hyperheuristics. **Bold**: outperforms *Base*.

	NHH	LUBY	MC	LGIHH	FSILS	TSILS	EAILS
X^0	1	69	105	210	193	220	187
X^+	14	82	137	–	–	–	–
X^A	94	136	149	–	–	–	–
X^{AR}	115	154	164	–	–	–	–
X^*	154	145	192	225	203	224	193

Table 4: Summary of F1 scores for 7 hyper-heuristics with (gradually) constructed LLH set transformations.

Overall, the intensities clearly show as an important performance leverage. All methods except for $LUBY^{AR}$ can be improved with at least one (but often multiple) setups *I-III*. Generally, we see the more conservative setups *II* and *III* as a better choice than *I*. The most consistent is the setup *III*. The setup *II* works well with methods not using the ILS scheme (see Table 1). Lastly, the more aggressive setup *I* rather deteriorates performance, an exception is methods using full search restarts (FSILS, LGIHH). Based on the results, we fix the setup to *II* for NHH^{AR}, MC^{AR}, and LGIHH, use the setup *III* for $LUBY^{AR}$, TSILS, and EAILS, and fix the setup *I* for FSILS. The resulting methods are referred to as X^*.

Results summary

Table 4 summarizes the F1 scores obtained for the incrementally constructed LLH set transformations in all of the 7 tested methods. The proposed transformations successfully boosted F1 scores for all of the 7 methods. For LGIHH, FSILS, TSILS, and EAILS, overriding (or adding the missing) intensity handling via transformed LLH sets systematically provides benefits. For LGIHH and TSILS, the results even outperform the original TSILS performance, setting the new state-of-the-art on the CHeSC benchmark. This overall suggests that manipulating the LLH sets offers a simpler and more transparent alternative to the parameter adaptation mechanisms present in the original methods. Regarding NHH, LUBY, and MC, we can see substantial benefits of the best transformations compared to both the X^0 and X^+ baseline variants. In the original CHeSC competition, NHH^* and $LUBY^*$ would rank 2nd with results close to the winning GIHH, and MC^* would score 1st. Strikingly, *NHH^* is a trivial unbiased LLH selection mechanism, only set to operate in a reasonably safe environment (acceptance) with normalized granularity at which new LLHs are sampled (repetitions), allowing the perturbation intensity to vary*. Crucially, this result implies that the key properties of the LLH set play a comparable, if not more important, role than the selection mechanism. In this regard, we contribute a general tool for controlling LLH aspects with critical impacts on the overall performance.

X vs.	NHH$^+$	LUBY$^+$	MC$^+$	LGIHH	FSILS	TSILS	EAILS	FRAMAB	LASTRL	QHH	MCTS	GEPHH
NHH*	**0.000**	0.226	0.379	0.931	0.824	0.878	0.908	**0.037**	0.525	0.185	0.948	0.941
LUBY*	**0.000**	**0.004**	0.059	0.899	0.792	0.982	0.605	**0.004**	0.617	0.103	0.906	0.725
MC*	**0.000**	**0.000**	**0.003**	0.686	0.625	0.878	0.311	**0.002**	0.245	0.067	0.533	0.410
LGIHH*	**0.000**	**0.000**	**0.002**	**0.037**	0.075	0.275	**0.003**	**0.000**	**0.004**	**0.000**	0.109	0.289
FSILS*	**0.000**	**0.006**	**0.017**	0.686	0.325	0.855	0.275	**0.000**	0.116	**0.001**	0.525	0.492
TSILS*	**0.000**	**0.000**	**0.002**	0.164	**0.015**	**0.020**	**0.000**	**0.000**	**0.002**	**0.000**	0.064	0.272
EAILS*	**0.000**	**0.000**	**0.031**	0.707	0.758	0.953	0.226	**0.000**	0.059	0.050	0.500	0.633

Table 5: Overview of p-values (< 0.05 in **bold**). Alternative hypothesis: ”row key is better than column key”.

NHH*	**446**	FSILS	463	FSILS	446	FSILS	451	FSILS	466	**TSILS***	**449**	FSILS	446
FSILS	445	EAILS	417	EAILS	412	**LGIHH***	**422**	EAILS	425	FSILS	441	**EAILS***	**431**
EAILS	395	TSILS	411	TSILS	405	EAILS	401	TSILS	416	EAILS	397	EAILS	393
TSILS	388	LGIHH	358	**MC***	**391**	TSILS	387	LGIHH	375	TSILS	386	TSILS	391
LGIHH	353	**LUBY***	**356**	LGIHH	347	LGIHH	349	**FSILS***	**278**	LGIHH	344	LGIHH	353
MC$^+$	244	MC$^+$	249	MC$^+$	254	MC$^+$	250	MC$^+$	263	MC$^+$	249	MC$^+$	247
LUBY$^+$	218	LUBY$^+$	219	LUBY$^+$	229	LUBY$^+$	227	LUBY$^+$	236	LUBY$^+$	222	LUBY$^+$	225
LUBY	153	LUBY	165	LUBY	157	LUBY	157	LUBY	173	LUBY	154	LUBY	153
MC	127	MC	131	MC	128	MC	124	MC	139	MC	127	MC	129

Table 6: F1 scores on the real-world domains. In each column, one X* method (in **bold**) competes against referential results.

Second, we provide statistical tests for our results following the approach taken in Choong et al. [2018]. For each instance, the methods' median results are min-max normalized. We use the min & max medians based on the original CHeSC results. Then, the normalized medians on the 30 competition instances are used to perform the Wilcoxon signed rank test for a pair of competing hyper-heuristics. Table 5 summarizes comparisons of the methods with the final LLH set transformations compared to the baselines and *all* recent hyper-heuristics. First, we conclude that the benefits of NHH*, LUBY*, and MC* compared to their X$^+$ counterparts are in all cases statistically significant (all X^0 are worse than X$^+$). Among the remaining methods, LGIHH* and TSILS* benefits are statistically significant. For FSILS and EAILS, the p-values are inconclusive. Interestingly, the improvements from MC$^+$ to MC* make it competitive with the majority of recent methods.

Real-world domains experiments

The second part of the experiments applies the developed LLH set transformations to three additional real-world domains, validating their generality. The first domain is a rich variant of the pickup-delivery problem with time windows (PDPTW) arising from a freight-transportation application Sassmann et al. [2023]. The objective is to minimize the travel distance and driver overtimes. The testing dataset consists of 18 instances, each with around 100 pickup-delivery transportation requests based on customer orders realized in the company Wereldo. We evaluate each instance 31 times with a 5-minute timeout per run, reflecting the typical use case for the solver. The other two domains have been described by Kletzander and Musliu [2024]. The second domain deals with the minimum shift design (MSD) problem. MSD aims to design shifts according to a given set of shift types such that a given demand for up to 50 employees working at each time slot with a granularity as low as 15 minutes is covered for a whole week. The objective is to cover the demand while minimizing the number of different pairs of shift starts and ends and the deviation from a target average shift length. We evaluate the same set of 33 realistic instances, each 5 times with a 60-minute timeout per run. The third domain is bus driver scheduling (BDS). In BDS, drivers are assigned to predetermined bus tours according to a complex set of constraints regarding limits of assignments and required breaks. A linear combination of several objectives is optimized. We evaluate 20 realistic instances with up to 1,000 bus legs. Each instance is evaluated 5 times with a 60-minute timeout per run. PDPTW runs are evaluated on *Debian 6.1.135 x86_64* using *AMD EPYC 7543* CPU (2.8 GHz). MSD and BDS runs are evaluated on *Ubuntu 22.04.2 LTS* with *Intel Xeon E5-2650 v4* processors (2.2 GHz). We provide further details about the individual domains and their implementation details (available LLHs, perturbation intensity handling) in Appendix B.

Regarding comparisons, we again use the F1 scoring system and Wilcoxon tests. We evaluate and use LUBY, LUBY$^+$, MC, MC$^+$, LGIHH, TSILS, EAILS, and FSILS as the referential results. We omit NHH and NHH$^+$ as both generally struggle to obtain feasible solutions. The referential results are used as the competitors in the F1 scoring and for both min-max normalization and comparisons with the Wilcoxon tests. Detailed results are in Appendix C.

Evaluation

When switching from the CHeSC to the real-world domains, we encountered only one important difference related to the repeated LLH applications. Compared to the

CHeSC domains, the real-world problems are more complex in terms of constraints and their evaluation. As a result, the LLHs have longer execution times, prolonging the repetition timeouts needed for the desirable effects. This shift is systematic and roughly one order of magnitude. Thus, we reflect the shift by a rough adjustment of the repetitions factor in NHH*, LUBY*, and MC* to 10 ms. With this only change, we evaluate our X* methods. Table 6 summarizes the F1 scores of X* competing against the referential results.

First, all methods X* using the proposed LLH set transformations clearly outperform their X^0 and X$^+$ counterparts with one exception. For FSILS, the formerly inconclusive effects of intensity modification *I* turned out to degrade the method's performance. The results thus tightly copy the key trends observed for the benchmark domains. Regarding the statistical significance, all observed improvements are significant, with the exception of EAILS* having the p-value of 0.051. Second, we perform an ablation analysis for NHH*, LUBY*, and MC* by sequentially adding the acceptance, repetition, and intensity transformations as for the CHeSC domains. We again confirm that each individual step adds to the overall performance of the transformations again replicating our observations from the CHeSC domains. Ultimately, we specifically emphasize the performance of NHH*. While the results suggest that the hyper-heuristics dominating the CHeCS benchmark (FSILS, EAILS, TSILS, LGIHH) generalize well to the new domains, NHH* outperforms all of these state-of-the-art methods. The only cases where NHH* does not outperform its competitor with a statistically significant difference are FSILS and TSILS with Wilcoxon test p-values of 0.232 and 0.070. When comparing NHH* with other X* methods, we report p-values 0.466 for TSILS*, 0.262 for EAILS*, 0.119 for LGIHH*, and p-values below the 0.05 threshold for FSILS*, LUBY*, and MC*. In a closing remark, NHH* also found several new best-known solutions, namely 3 for PDPTW and 8 for MSD, showing that simplicity and generality do not necessarily come at the expense of result quality.

5 Conclusion

Our paper identifies three critical principles affecting cross-domain search performance and exploits them to improve a wide range of existing hyper-heuristics. We demonstrate that the strategic transformations of LHH sets allow for outperforming the current state-of-the-art hyper-heuristics on both the standard CHeSC benchmark and three real-world domains. Strikingly, we achieve excellent results with a trivial random unbiased selection mechanism combined with a properly constructed set of LLHs. In this respect, we emphasize the conclusions of one of the pioneering works in hyper-heuristics Fisher and Thompson [1963]: "(1) an unbiased random combination of scheduling rules is better than any of them taken separately; (2) learning is possible". To (1), we add that a trivial unbiased random combination of LLHs may perform surprisingly well given the right set of LLHs. Regarding (2), we agree that learning *how to select LLHs* is possible. Yet, we demonstrate and emphasize that the control over *what LLHs we select* is comparably, if not more, important.

Acknowledgments

Computational resources were provided by the e-INFRA CZ project (ID:90254), supported by the Ministry of Education, Youth and Sports of the Czech Republic.
The research was supported by a grant from the OeAD-GmbH and AKTION as part of the scholarship programme Aktion Austria-Czech Republic.
We would like to thank our industrial partner, Wereldo, for providing us with the real-world problem instances.
The financial support by the Austrian Federal Ministry for Digital and Economic Affairs, the National Foundation for Research, Technology and Development and the Christian Doppler Research Association is gratefully acknowledged.

References

Michel Gendreau and Jean-Yves Potvin. *Handbook of metaheuristics*. Springer, 2010.

John H. Drake, Ahmed Kheiri, Ender Özcan, and Edmund K. Burke. Recent advances in selection hyper-heuristics. *European Journal of Operational Research*, 285(2): 405–428, 2020. doi: https://doi.org/10.1016/j.ejor.2019.07.073.

Edmund K Burke, Michel Gendreau, Matthew Hyde, Graham Kendall, Barry McCollum, Gabriela Ochoa, Andrew J Parkes, and Sanja Petrovic. The cross-domain heuristic search challenge–an international research competition. In *International Conference on Learning and Intelligent Optimization*, pages 631–634. Springer, 2011.

Mustafa Mısır, Katja Verbeeck, Patrick De Causmaecker, and Greet Vanden Berghe. An intelligent hyper-heuristic framework for CHeSC 2011. In *International Conference on Learning and Intelligent Optimization*, pages 461–466. Springer, 2012.

Edmund K. Burke, Michel Gendreau, Matthew Hyde, Graham Kendall, Gabriela Ochoa, Ender Özcan, and Rong Qu. Hyper-heuristics: A survey of the state of the art. *Journal of the Operational Research Society*, 64(12):1695–1724, 2013. doi: 10.1057/jors.2013.71.

Tansel Dokeroglu, Tayfun Kucukyilmaz, and El-Ghazali Talbi. Hyper-heuristics: A survey and taxonomy. *Computers & Industrial Engineering*, 187:109815, 2024.

Edmund K Burke, Matthew R Hyde, Graham Kendall, Gabriela Ochoa, Ender Özcan, and John R Woodward. A classification of hyper-heuristic approaches: revisited. *Handbook of Metaheuristics*, pages 453–477, 2019.

G. Ochoa, M. Hyde, T. Curtois, J.A. Vazquez-Rodriguez, J. Walker, M. Gendreau, G. Kendall, B. McCollum, A.J. Parkes, S. Petrovic, and E.K. Burke. HyFlex: A Benchmark Framework for Cross-domain Heuristic Search. *European Conference on Evolutionary Computation in Combinatorial Optimisation (EvoCOP 2012)*, 7245:136–147, 2012.

Ping-Che Hsiao, Tsung-Che Chiang, and Li-Chen Fu. A vns-based hyper-heuristic with adaptive computational budget of local search. In *2012 IEEE Congress on Evolutionary Computation*, pages 1–8. IEEE, 2012.

Mathieu Larose. A hyper-heuristic for the CHeSC 2011. *CHeSC2011 Competition*, 2011.

Mohamad Khairulamirin Md Razali, Masri Ayob, Abdul Hadi Abd Rahman, Razman Jarmin, Chian Yong Liu, Muhammad Maaya, Azarinah Izaham, and Graham Kendall. Unveiling effective heuristic strategies: A review of cross-domain heuristic search challenge algorithms. *CMES - Computer Modeling in Engineering and Sciences*, 142(2):1233–1288, 2025.

Steven Adriaensen and Ann Nowé. Case study: An analysis of accidental complexity in a state-of-the-art hyper-heuristic for hyflex. In *2016 IEEE Congress on Evolutionary Computation (CEC)*, pages 1485–1492. IEEE, 2016.

Steven Adriaensen, Tim Brys, and Ann Nowé. Fair-share ils: A simple state-of-the-art iterated local search hyperheuristic. In *Proceedings of the 2014 Annual Conference on Genetic and Evolutionary computation*, pages 1303–1310, 2014.

Stephen A Adubi, Olufunke O Oladipupo, and Oludayo O Olugbara. Configuring the perturbation operations of an iterated local search algorithm for cross-domain search: A probabilistic learning approach. In *2021 IEEE Congress on Evolutionary Computation (CEC)*, pages 1372–1379. IEEE, 2021.

Stephen A Adubi, Olufunke O Oladipupo, and Oludayo O Olugbara. Evolutionary algorithm-based iterated local search hyper-heuristic for combinatorial optimization problems. *Algorithms*, 15(11):405, 2022.

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

Nasser R Sabar, Masri Ayob, Graham Kendall, and Rong Qu. Automatic design of a hyper-heuristic framework with gene expression programming for combinatorial optimization problems. *IEEE Transactions on Evolutionary Computation*, 19(3): 309–325, 2014.

Nasser R. Sabar and Graham Kendall. Population based monte carlo tree search hyper-heuristic for combinatorial optimization problems. *Information Sciences*, 314:225–239, 2015.

Alexandre Silvestre Ferreira, Richard Aderbal Goncalves, and Aurora Trinidad Ramirez Pozo. A multi-armed bandit hyper-heuristic. In *2015 Brazilian conference on intelligent systems (BRACIS)*, pages 13–18. IEEE, 2015.

Shin Siang Choong, Li-Pei Wong, and Chee Peng Lim. Automatic design of hyper-heuristic based on reinforcement learning. *Information Sciences*, 436-437:89–107, 2018.

Florian Mischek and Nysret Musliu. Reinforcement learning for cross-domain hyper-heuristics. In *International Joint Conferences on Artificial Intelligence (IJCAI 2022)*, pages 4793–4799, 2022.

Lucas Kletzander and Nysret Musliu. Large-state reinforcement learning for hyper-heuristics. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 12444–12452, 2023.

Chung-Yao Chuang. *Combining multiple heuristics: Studies on neighborhood-base heuristics and sampling-based heuristics*. PhD thesis, Carnegie Mellon University, 2020.

Michael Luby, Alistair Sinclair, and David Zuckerman. Optimal speedup of las vegas algorithms. *Information Processing Letters*, 47(4):173–180, 1993.

Gunter Dueck and Tobias Scheuer. Threshold accepting: A general purpose optimization algorithm appearing superior to simulated annealing. *Journal of Computational Physics*, 90(1):161–175, 1990.

Gunter Dueck. New optimization heuristics: The great deluge algorithm and the record-to-record travel. *Journal of Computational Physics*, 104(1):86–92, 1993.

Nicholas Metropolis, Arianna W Rosenbluth, Marshall N Rosenbluth, Augusta H Teller, and Edward Teller. Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, 21(6):1087–1092, 1953.

Vojtěch Sassmann, Hana Rudová, Michal Gabonnay, and Václav Sobotka. Real-world vehicle routing using adaptive large neighborhood search. In Leslie Pérez Cáceres and Thomas Stützle, editors, *Evolutionary Computation in Combinatorial Optimization*, pages 34–49. Springer Nature Switzerland, 2023.

Lucas Kletzander and Nysret Musliu. Hyper-heuristics for personnel scheduling domains. *Artificial Intelligence*, 334:104172, 2024.

H Fisher and GL Thompson. Probabilistic learning combinations of local job-shop scheduling rules. In *Industrial Scheduling*, page 225–251. Prentice-Hall, 1963.