

# Divide and Learn: Multi-Objective Combinatorial Optimization at Scale

Esha Singh

Dongxia Wu

Chien-Yi Yang

*Department of Computer Science &  
Engineering UC, San Diego*

*Department of Statistics  
Stanford University*

*Department of Computer Science &  
Engineering UC, San Diego*

Tajana Rosing

Rose Yu

Yi-An Ma

*Department of Computer Science & Department of Computer Science & Halıcıoğlu Data Science Institute  
Engineering UC, San Diego      Engineering UC, San Diego      UC, San Diego*

## Abstract

Multi-objective combinatorial optimization seeks Pareto-optimal solutions over exponentially large discrete spaces, yet existing methods sacrifice generality, scalability, or theoretical guarantees. We reformulate it as an online learning problem over a decomposed decision space, solving position-wise bandit subproblems via adaptive expert-guided sequential construction. This formulation admits regret bounds of  $O(d\sqrt{T \log T})$  depending on subproblem dimensionality  $d$  rather than combinatorial space size. On standard benchmarks, our method achieves 80–98% of specialized solvers performance while achieving two to three orders of magnitude improvement in sample and computational efficiency over Bayesian optimization methods. On real-world hardware-software co-design for AI accelerators with expensive simulations, we outperform competing methods under fixed evaluation budgets. The advantage grows with problem scale and objective count, establishing bandit optimization over decomposed decision spaces as a principled alternative to surrogate modeling or offline training for multi-objective optimization.

## 1 Introduction

Multi-objective combinatorial optimization (MOCO) requires balancing multiple competing objectives over combinatorial spaces. Oftentimes, each evaluation can cost hours of simulation, hence exhaustive search becomes impossible and gradient descent inapplicable. Yet this is precisely the setting practitioners face in hardware-software co-design, protein engineering, and drug discovery [Yang et al. \(2025\)](#); [Krishnan et al. \(2023\)](#); [Romero et al. \(2013\)](#); [Brown et al. \(2019\)](#): approximating Pareto-optimal (i.e., solutions where no objective can improve without degrading another) trade-offs over discrete or mixed-integer variables, with no closed-form gradients and no cheap evaluations.

Existing approaches to MOCO include surrogate-based methods such as Multi-objective Bayesian optimization which has advanced significantly for continuous domains [Cockuyt et al. \(2014\)](#); [Zhao et al. \(2019\)](#); [Knowles \(2006\)](#) but struggle in combinatorial settings. Discrete spaces introduce distinct challenges: search spaces grow exponentially ( $|\mathcal{X}| = O(k^n)$  for  $n$  decision variables with  $k$  values each) [Baptista and Poloczek \(2018\)](#); [Oh et al. \(2019\)](#), smooth structure is absent for surrogate modeling [Garrido-Merchán and Hernández-Lobato \(2020\)](#), and correlation patterns between configurations are difficult to capture with standard kernels [Saves et al. \(2023\)](#). Classical evolutionary methods [Deb et al. \(2002\)](#); [Zhang and Li \(2007\)](#) and, more recently, neural methods

that learn search policies via deep reinforcement learning [Bello et al. \(2016\)](#); [Kool et al. \(2019\)](#) have emerged; Multi-objective extensions [Lin et al. \(2022\)](#) achieve strong generalization across problem instances but demand millions of training samples and substantial computational overhead. Any practical algorithm must therefore balance three competing demands: (i) computational scalability as the search space grows, (ii) sample efficiency under expensive evaluation budgets, and (iii) tractable per-iteration complexity. Achieving this balance remains hard: surrogate methods minimize evaluations but incur cubic model overhead; evolutionary heuristics offer generality but lack convergence guarantees; problem-specific heuristics [Helsgaun \(2000\)](#); [Tinós et al. \(2018\)](#) exploit structure effectively but do not generalize across domains.

We propose an alternative framing: MOCO as a sequential decision problem under uncertainty, where feedback from each evaluation informs subsequent search. This formulation requires no offline training, learns directly from expensive evaluations, and admits formal regret guarantees. The key insight is that tractable structure can be imposed on exponential search spaces, enabling principled exploration where exhaustive search is infeasible.

**Contributions** We make the following contributions

- We propose DIVIDE & LEARN (D&L), a novel *position-wise bandits with multiple experts* (D&L) framework, for online **multi-objective combinatorial optimization**, that transforms intractable exponential search into efficient sequential decision-making by exploiting problem structure and adapting to observed rewards—without offline training or surrogate models.
- We establish regret bounds of  $O(d\sqrt{T \log T})$  that depend only on subproblem size  $d \ll n$  (with  $n$  decision variables and  $T$  iterations), rather than on the size of the combinatorial action space  $|\mathcal{X}| \in \{2^n, n!\}$ —an exponential-to-polynomial reduction over standard combinatorial bandit approaches.
- We validate on standard MOCO benchmarks with search spaces up to  $10^{60}$  configurations and a real-world hardware-software co-design problem with extreme evaluation costs. D&L achieves 80–98% of specialized solvers performance, matches offline-trained neural solvers without their training overhead, uses 90% less compute than Bayesian optimization, yielding superior Pareto front diversity—with advantages growing at larger scales and objective counts.

## 2 Related Work

**Surrogates, Evolutionary, and Neural Methods.** Multi-objective Bayesian optimization uses Gaussian Process (GP) surrogates and scalarizations such as ParEGO [Knowles \(2006\)](#) or EHVI [Daulton et al. \(2021\)](#), but scales cubically in evaluations and requires intractable acquisition optimization over combinatorial domains. Evolutionary approaches such as MOEA/D [Zhang and Li \(2007\)](#) decompose objectives via weight vectors and rely on heuristic operators without finite-time guarantees in black-box combinatorial settings. Neural combinatorial optimization methods amortize inference through offline training [Kool et al. \(2019\)](#); [Lin et al. \(2022\)](#); [Chen et al. \(2023\)](#) but require large training corpora and may degrade under distribution shift.

**Decomposition-based MOCO.** MOEA/D [Zhang and Li \(2007\)](#) decomposes in **objective space**: subproblems correspond to weight vectors, solved via evolutionary operators with implicit coordination through mating restriction. D&L decomposes in **decision space**: subproblems correspond to variable groups, solved via bandit-based selection with explicit Lagrangian coordination. The extensive MOEA/D literature [Qi et al.](#)

(2013); Jiang et al. (2011); Cheng et al. (2016); Trivedi et al. (2017) focuses on adaptive weights and scalarization without global regret guarantees. Existing theoretical analyses Huang et al. (2021); Li et al. (2015) are *post-hoc* under restrictive assumptions; D&L yields  $O(d\sqrt{T \log T})$  regret as a direct consequence of its formulation.

**Neural Combinatorial Optimization.** Attention-based models Bello et al. (2016); Kool et al. (2019) achieve strong single-objective performance; multi-objective extensions include preference-conditioned models (PMOCO) Lin et al. (2022) and diversity-enhanced heuristics (NHDE) Chen et al. (2023). These require substantial offline training and may need retraining when problem structure shifts Angioni et al. (2025). D&L requires no pretraining and adapts online, suited to expensive evaluations and limited budgets.

**Combinatorial Bandits.** Combinatorial bandit methods Chen et al. (2013); Kveton et al. (2015) achieve sublinear regret under *semi-bandit feedback*, observing per-component rewards rather than aggregate solution quality. Linear bandits Dani et al. (2008) handle full-bandit feedback but impose linear reward structure, and multi-objective extensions (COMO-MAB) Öner et al. (2018) inherit both restrictions, enabling stronger estimators and Pareto regret guarantees at the cost of generality. Adversarial variants Cesa-Bianchi and Lugosi (2012); Audibert et al. (2014) address non-stochastic settings but still require semi-bandit observations. These assumptions fail in black-box MOCO, where evaluating a solution yields only aggregate objective values and rewards may depend nonlinearly on global interactions. D&L targets this harder feedback model, trading estimator strength for generality while achieving regret scaling with subproblem dimension  $d$  rather than  $n$ .

**Online learning for black-box MOCO** faces exponential action spaces, non-differentiable objectives, and costly evaluations, precluding enumeration, gradient-based optimization, and exhaustive sampling; any sample-efficient algorithm must therefore select queries adaptively based on past observations—the explore-exploit structure formalized by online learning, yet existing combinatorial and MO bandit formulations rely on semi-bandit feedback, linear structure, or oracle selection, enabling tighter estimators but limiting applicability in black-box combinatorial settings. To the best of our knowledge, there are no regret-theoretic results or application of black-box MOCO under full-bandit (scalar) feedback. D&L bridges this gap by decomposing combinatorial decisions into coordinated local components learned online from aggregate feedback, achieving finite-time regret guarantees with lightweight coordination (Sections 4, 5).

### 3 Preliminaries

**Problem Statement.** We study multi-objective optimization over combinatorial domains:

$$\min_{x \in \mathcal{X}} \mathbf{f}(x) = [f_1(x), \dots, f_m(x)]^\top \quad (1)$$

where each objective  $f_i : \mathcal{X} \rightarrow \mathbb{R}$  is a black-box, expensive-to-evaluate, non-convex, and possibly non-smooth function. The domain  $\mathcal{X}$  is a combinatorial or mixed-discrete space comprising permutations, binary vectors, integer or categorical assignments, or combinations thereof (see Appendix 10.1.1). Consequently, the search space scales as  $|\mathcal{X}| = O(n!)$  or  $O(2^n)$  depending on problem structure.

**Pareto Optimality.** A solution  $x \in \mathcal{X}$  is *Pareto optimal* if no  $x' \in \mathcal{X}$  satisfies  $f_i(x') \leq f_i(x)$  for all  $i \in \{1, \dots, m\}$  with at least one strict inequality. The *Pareto set*  $\mathcal{P}^* \subset \mathcal{X}$  comprises all Pareto optimal solutions:  $\mathcal{P}^* = \{x \in \mathcal{X} : \nexists x' \in \mathcal{X}, \mathbf{f}(x') \preceq \mathbf{f}(x) \wedge \mathbf{f}(x') \neq \mathbf{f}(x)\}$  where  $\preceq$  denotes component-wise inequality (Pareto dominance); image  $\mathbf{f}(\mathcal{P}^*)$  in objective space is the *Pareto front*.

**Gaussian Processes.** Bayesian optimization Močkus (1975) models objectives via Gaussian Processes priors, yielding posterior mean  $\mu_{i,t}(x)$  and variance  $\sigma_{i,t}^2(x)$  that guide acquisition functions balancing exploration-exploitation. Common multi-objective acquisitions include EHVI, ParEGO Knowles (2006), and UCB Auer et al. (2002a).

**Acquisition Functions.** In sequential optimization, acquisition functions balance exploration and exploitation. Upper Confidence Bound (UCB) Auer et al. (2002a) selects actions  $a$  maximizing  $\hat{V}(a) + c\sqrt{\log t/N(a)}$ —estimated value plus an uncertainty bonus that shrinks with visits. D&L adapts this principle to combinatorial domains via position-wise bandits (Section 4.2.2).

## 4 The D&L Algorithm

### 4.1 Online Learning Formulation

We reformulate multiobjective combinatorial optimization as an online learning problem over  $T$  iterations Cesa-Bianchi and Lugosi (2006). At each iteration  $t$ , the learner selects a solution  $x_t \in \mathcal{X}$  and observes a scalar reward  $r_t(x_t) = \phi(\mathbf{f}(x_t)) + \epsilon_t$ , where  $\mathbf{f}(x) = (f_1(x), \dots, f_m(x))$  is the multi-objective vector,  $\phi(\cdot)$  is a scalarization<sup>1</sup> (e.g., weighted sum), and  $\epsilon_t$  is stochastic noise Ehrgott (2005). Furthermore, the reward function  $r_t(\cdot)$  is **non-convex** and observed with bounded, zero-mean stochastic noise, i.e.,  $\mathbb{E}[\epsilon_t | x_1, r_1, \dots, x_{t-1}, r_{t-1}] = 0$  and  $|\epsilon_t| \leq \sigma$  almost surely. While D&L is empirically robust to non-stationarity, our analysis focuses on this stochastic setting without convexity or linearity assumptions. Crucially, unlike standard multi-armed or semi-bandit settings, the learner observes only a single scalar reward for the entire selected solution—*full-bandit feedback* Combes et al. (2015)—not per-position or per-arm observations. This is among the most challenging feedback models for combinatorial optimization.

A solution  $x = (x_1, \dots, x_n)$  assigns action  $x_i \in \mathcal{A}_i$  to each position  $i \in [n]$ ; we treat each position as a multi-armed bandit with arm set  $\mathcal{A}_i$ . The exponential cardinality  $|\mathcal{X}| = O(2^n)$  or  $O(n!)$  combined with bandit feedback renders direct application of online learning algorithms intractable. Our objective is to minimize cumulative regret against the best fixed solution in hindsight for the induced scalar reward:

$$R(T) = \sum_{t=1}^T r_t(x^*) - \sum_{t=1}^T r_t(x_t), \quad x^* = \arg \max_{x \in \mathcal{X}} \mathbb{E}[r_t(x)]$$

To approximate the Pareto front, we execute D&L with multiple scalarizations; under finite evaluation budgets, each run yields a regret-bounded approximation. To enable tractable regret analysis, we impose the following weak regularity conditions on the problem.

**Definition 4.1 (Assumptions).** We require (detailed formal statements in Appendix 12.2):

A.1 (**Decomposability**)  $\mathcal{X}$  decomposes into  $K$  overlapping subproblems  $\{\mathcal{X}^k\}_{k=1}^K$  with overlap set  $\mathcal{O}$ .

A.2 (**Discrete Lipschitz / Bounded Differences**)  $|f_j(x) - f_j(x')| \leq L \cdot \delta(x, x')$  for problem-specific metric  $\delta$ . This bounds global sensitivity under arbitrary solution perturbations.

A.3 (**Bounded Coupling**) Modifying subproblem  $k$  induces bounded non-local effects:  $|f_j(x') - f_j(x)| \leq L \cdot \delta(x, x') + C \cdot |S_k \cap \mathcal{O}|$ , where  $S_k \subseteq [n]$  is the index set of subproblem  $k$ . This only requires that cross-subproblem interactions scale linearly with overlap—not additive decomposability of  $f$ .

---

<sup>1</sup>Scalarization defines the reward signal  $r_t = \phi(\mathbf{f}(x_t))$ , not the optimization procedure—D&L is scalarization-agnostic. See 9.1

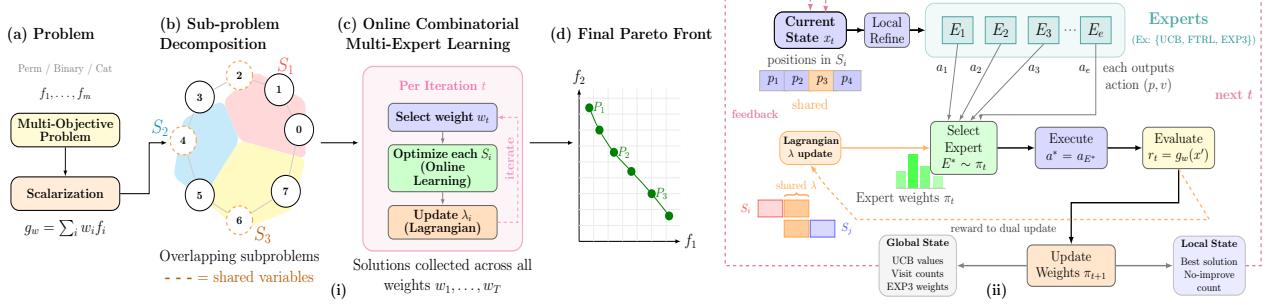


Figure 1: **Overview of D&L.** (i) Any multi-objective problem is scalarized & decomposed into overlapping subproblems with shared variables (dashed nodes in (b)). Bandit-based action selection optimizes each subproblem independently; Lagrangian duality enforces cross-subproblem consistency (ii) **Online Combinatorial Multi-expert learning:** For each position in a subproblem, an expert (this work: UCB, FTRL, EXP3 or TS, EXP3, see Section 4.2.2) is sampled via the mixture distribution  $\pi_t$  to propose an action. All experts update shared position-action statistics, amortizing exploration across exponential solution spaces. Lagrangian multipliers coordinate overlaps.

A.4 (**Bounded Range**) Objectives satisfy  $f_j(x) \in [0, B]$  for all  $x \in \mathcal{X}$  and  $j \in [m]$ .

Given these conditions, we now describe how D&L achieves sublinear regret through problem decomposition and an ensemble of no-regret algorithms Freund and Schapire (1997); Arora et al. (2012). At each position, one algorithm—termed an *expert*—is stochastically selected to propose an action based on learned action-quality estimates, enabling heterogeneous exploration across decision coordinates. All experts update from every observed reward—decoupling strategy selection from reward assumptions. The selection distribution adapts to local estimation uncertainty, favoring exploration where confidence is low and exploitation as estimates converge.

Since maintaining statistics over  $|\mathcal{X}|$  is intractable, experts share parameters at the position-action level, requiring  $O(n \cdot |\mathcal{A}|)$  memory. The shared parameters are the minimal statistics each expert requires (Section 4.2.2). When a reward is observed, all position-action pairs in the selected solution update simultaneously. Combined with subproblem decomposition (Section 4.2.1), this enables information transfer: observations in one subproblem improve decisions wherever the same variables appear.

## 4.2 Algorithm Overview

D&L addresses the intractability of combinatorial bandits through three coordinated mechanisms, see Algorithm 1, Figure 1. First, we *decompose* the solution space into  $K$  overlapping subproblems, each involving  $O(n/K)$  variables, reducing effective action space from  $O(2^n)$  to  $O(K \cdot 2^{n/K})$ . *Lagrangian relaxation* (DUALUPDATE) reconciles overlaps through dual multipliers, ensuring cross-subproblem consistency without destroying separability. Second, *multi-expert learning* (SELECTACTION) proposes actions at each position by sampling from experts sharing global statistics; LOCALREFINE captures within-subproblem interactions via zeroth-order perturbations. Third, all experts update statistics (UPDATEREPORTS) from observed reward  $r^{(t)}$ , decoupling selection from learning. We detail each component in the following subsections.

### 4.2.1 Decomposition Strategy

The exponential action space makes maintaining per-solution statistics intractable—the fundamental bottleneck for bandit algorithms in combinatorial domains without finite arm assumptions. Our key insight is that while

---

**Algorithm 1** D&L: Position-wise Bandits with Multiple Experts

---

**Require:** Size  $n$ , objectives  $\mathbf{f} : \mathcal{X} \rightarrow \mathbb{R}^m$ , subproblems  $\{S_1, \dots, S_K\}$ , weights  $\mathcal{W}$ , iters  $T$ ,  $\hat{V}$ : values,  $N$ :counts,  $W$ : EXP3 weights,  $\tilde{L}$ : FTRL losses

**Output:** Pareto archive  $\mathcal{P}$

```

1: for  $w \in \mathcal{W}$  do ▷ Each scalarization
2:    $\hat{V}, N, W, \tilde{L} \leftarrow \mathbf{0}_{n \times |\mathcal{A}|}; \lambda \leftarrow \mathbf{0}_n; r^* \leftarrow -\infty$ 
3:    $x_i^{(0)} \leftarrow \text{SELECTACTION}(i) \forall i \in [n]$  ▷ ALG. 2: Expert-guided init
4:   for  $t = 1, \dots, T$  do
5:     for  $k = 1, \dots, K$  do  $x^{(t)} \leftarrow \text{LOCALREFINE}(x^{(t)}, S_k)$  ▷ ALG. 3
6:     end for
7:      $r^{(t)} \leftarrow \sum_i w_i f_i(x^{(t)})$  ▷ Scalarized reward
8:      $\text{UPDATEEXPERTS}(x^{(t)}, r^{(t)}, \hat{V}, N, W, \tilde{L})$  ▷ ALG. 5
9:      $\lambda^{(t)} \leftarrow \text{DUALUPDATE}(\lambda^{(t-1)}, x^{(t)}, \mathcal{S})$  ▷ ALG. 4
10:    if  $r^{(t)} > r^*$  then  $x^* \leftarrow x^{(t)}; r^* \leftarrow r^{(t)}$  ▷ Track best
11:    end if
12:   end for
13:    $\mathcal{P} \leftarrow \mathcal{P} \cup \{(x^*, \mathbf{f}(x^*))\}$  ▷ Add to Pareto archive
14: end for

```

---

the solution space is exponential, the *decision structure* is factored: a solution comprises  $n$  position-action assignments optimizable in coordinated groups. This is **decision-space** decomposition—partitioning variables, not objectives. Our decomposition maintains constant subproblem size ( $d$ ) as  $n$  grows while preserving regret bounds—combining scalability with theoretical grounding.

We partition the  $n$  variables into  $k$  overlapping subproblems  $\{S_1, \dots, S_K\}$ , each involving  $d = O(n/K)$  variables. This achieves two reductions: (1) per-subproblem action space shrinks from  $2^n$  to  $2^d$ , and (2) effective dimensionality reduces from  $n$  to  $d$ , improving sample efficiency. Overlap regions  $\mathcal{O}_{pq} = \mathcal{S}_p \cap \mathcal{S}_q$  enable information flow across partitions. However, overlapping variables must satisfy consistency constraints  $x_i^{(p)} = x_i^{(q)}$  for all  $i \in \mathcal{O}_{pq}$ . Without coordination, independent subproblem optimization yields conflicting assignments.

**Definition 4.2 (Metric-Based Decomposition).** Given problem-specific distance  $D : [n] \times [n] \rightarrow \mathbb{R}_+$ , we construct subproblems via:

1. *(Sliding Window)*  $S_k = \{(k-1)(s-o_t) + 1, \dots, \min(n, (k-1)(s-o_t) + s)\}$  with diminishing overlap  $o_t = \lfloor o_0 \cdot t^{-\alpha} \rfloor$ .
2. *(k-Nearest Neighbors)*  $S_c = \arg \min_{|S|=s} \sum_{j \in S} D_{c,j}$ .<sup>2</sup>

When overlap is time-varying,  $S_k^{(t)}$  denotes the subproblem at iteration  $t$ . See figure 20 and Appendix 12.16, 12.7

**Remark 4.3 (Domain Agnosticity).** The decomposition operates on indices  $[n]$ , not solution values—the same construction applies to permutations (using Kendall tau distance Kendall (1938)), binary vectors (Hamming distance), and other combinatorial domains.

<sup>2</sup>  $D$  encodes problem locality: geographic distance (TSP), value-weight similarity (Knapsack). Grouping nearby indices reduces cross-subproblem dependencies.

**Proposition 4.4 (Bounded Coupling).** *The metric-based decomposition (Definition 4.2) satisfies Assumption A.3. Let  $\rho := \max_i |\{k : i \in S_k\}|$  denote the maximum overlap multiplicity. For any modification on subproblem  $S_k$ :*

$$|f_j(x') - f_j(x)| \leq L \cdot \Delta_s + C \cdot (\rho - 1) \cdot s$$

where  $\Delta_s := \max_{x,x' \in \mathcal{X}^s} \delta(x, x')$  denotes the diameter of a subproblem of size  $s$  under the problem-specific metric, and  $\rho$  is the maximum overlap multiplicity.

Proposition 4.4 ensures subproblem-level optimization induces bounded objective variation—the foundation for regret decomposition.

**The Conflict Problem** The overlap regions  $\mathcal{O}_{pq}$  in Definition 4.2 are intentional: shared variables enable learning to transfer across partitions. However, early in learning, different subproblems may favor incompatible actions for shared positions, producing inconsistent solutions whose rewards cannot be reliably attributed and whose errors can cascade across the partition structure. We control these effects using two complementary mechanisms. *Diminishing overlap* ( $o_t = \lfloor o_0 t^{-\alpha} \rfloor$ ,  $\alpha = 0.5$ ) progressively shrinks the shared region, guaranteeing  $\sum_t o_t = O(\sqrt{T})$  (Appendix 12.26); this limits where conflicts can occur but does not prevent them. *Lagrangian relaxation* is adaptive: it penalizes disagreement precisely at positions where conflicts arise and vanishes as estimates converge. Neither mechanism alone suffices—Lagrangian penalties over large overlaps can incur linear cost, while diminishing overlap without active resolution leaves conflicts unresolved (see 11.6).

**Coordinating Overlapping Subproblems via Lagrangian Relaxation** To resolve conflicts without destroying separability we relax the hard consistency constraints  $x_i^{(p)} = x_i^{(q)}$  into soft penalties. We maintain dual variables  $\lambda_i \geq 0$  for each overlapping position  $i \in \mathcal{O}$  & penalize a *soft violation measure*  $\xi_i^{(t)}$  that quantifies coordination risk with shared value estimates  $\hat{V}$ , visit counts  $N$ :

$$\begin{aligned} \mathcal{L}(x, \lambda) &= r_t(x) - \sum_{i \in \mathcal{O}} \lambda_i \cdot \xi_i^{(t)} \\ \xi_i^{(t)} &= (k_i - 1) \cdot \text{Var}(\hat{V}_{i,\cdot}^{(t)}) \cdot \left( 1 - \frac{N_{i,x_i^{(t)}}}{\sum_a N_{i,a}} \right) \end{aligned}$$

where  $k_i$  counts subproblems containing position  $i$ , the variance captures disagreement potential, and the visit ratio reflects assignment confidence. Crucially,  $\xi_i^{(t)}$  upper-bounds the expected disagreement probability across overlapping subproblems (Lemma 12.34), permitting Lagrangian relaxation when hard violations are unobservable under full-bandit feedback. For fixed  $\lambda$ , this decomposes across subproblems, preserving separability. We update  $\lambda_i$  via accelerated mirror descent Beck and Teboulle (2003) on the resulting convex dual objective, with step size  $\alpha_t = O(1/\sqrt{t})$ ; as learning progresses,  $\text{Var}(\hat{V}_{i,\cdot}) \rightarrow 0$  implies  $\xi_i^{(t)} \rightarrow 0$ , naturally diminishing penalties (Appendix 12.6.3).

**Theorem 4.5** (Coupling Error Bound). *Under Assumptions A.1–A.4 and Lagrangian coordination with accelerated mirror descent, expected cumulative coupling error  $C_t$  satisfies:*

$$\mathbb{E} \left[ \sum_{t=1}^T C_t \right] = O(K^2 Q R_{\max} \sqrt{T})$$

where  $K$  is the number of subproblems,  $Q = \max_{p,q} |\mathcal{O}_{pq}|$  is maximum overlap size, and  $R_{\max}$  is maximum per-position regret. See Appendix 12.11 for the full proof.

This  $O(\sqrt{T})$  rate ensures coordination costs diminish sublinearly—the “**price of coordination.**” The bound reveals a tradeoff: larger  $K$  reduces per-subproblem regret but increases coordination overhead ( $\tilde{O}(K)$  under sparse local overlap). Balancing yields an asymptotically optimal decomposition  $K^* = O(\sqrt{n/(QR_{\max})})$ . With coordination established, we now turn to how each subproblem is solved.

#### 4.2.2 Multi-Expert Learning for Subproblems

We employ a *multi-expert* strategy instantiating three no-regret algorithms with complementary strengths. The first expert (UCB) provides optimistic action selection, balancing exploration and exploitation under stochastic rewards. The second expert (EXP3) is adversarially robust and addresses the core challenge of bandit feedback—estimating rewards for unchosen actions via importance weighting by inverse selection probabilities [Auer et al. \(2002b\)](#), which can suffer high variance when probabilities are small [Bubeck and Cesa-Bianchi \(2012\)](#). Under full-bandit feedback, we use a clipped importance-weighted estimator for each position-action pair  $(i, a)$  with marginal selection probability  $\pi_{t,i}(a) = \sum_e p_e^{(t)} \pi_{t,i}^{(e)}(a)$  induced by the expert mixture; clipping controls variance with bounded cumulative bias  $O(dB \log d/\eta)$  that diminishes as probability concentrates on optimal actions, yielding  $O(d\sqrt{T \log T})$  regret (Appendix 12.14). In our setting, EXP3 operates at the position-action level, correcting for the probability of selecting each action under the expert mixture. The third expert (FTRL) [Hazan \(2016a\)](#) optimizes a distribution over actions on the probability simplex  $\Delta_{|\mathcal{A}|}$ , where expected loss becomes linear, enabling convex optimization with entropic regularization for variance reduction and accelerated convergence (see Fig 1-ii). Thus, at each position, one expert is sampled via mixture weights  $\rho$  to select actions, but *all experts update parameters* after observing rewards—ensuring robustness across reward structures. Unlike standard expert advice, selection adapts to local estimation uncertainty rather than tracking expert performance (Appendix 11.3). Parameters  $\hat{V}, N, W, \tilde{L}$  indexed by position-action pairs  $(i, a)$  are shared globally across subproblems, enabling information transfer through overlapping positions.

**Expert Selection and Coordination** Algorithm 2 details the selection procedure. The dual variables  $\lambda_i$  from Lagrangian coordination (12.6.3) track coordination uncertainty at overlapping positions without altering action preferences within a position. As estimates converge, conflicts naturally diminish (Theorem 4.5), coupling bandit learning with constraint satisfaction: experts<sup>3</sup> maximize reward while respecting cross-subproblem consistency.

**Local Refinement and Guarantees.** Expert-driven selection operates at the position level, constructing solutions from per-position statistics. However, this coarse granularity cannot capture fine-grained interactions within subproblems. Between expert-driven phases (every  $c$  iterations), *gradient-free local search* refines solutions via zeroth-order perturbations [Flaxman et al. \(2005\)](#) within subproblems, exploiting local structure that per-position statistics miss. Critically, all expert parameters update at every iteration, including during local search, ensuring no samples are wasted. See Lemma 12.4 and Algorithm 3 for details.

## 5 Regret Bounds

We establish regret guarantees for D&L through a novel decomposition that separates subproblem learning from coordination costs.

---

<sup>3</sup>Note, the multi-expert set {UCB, EXP3, FTRL} is flexible—any no-regret algorithms with complementary strengths may be substituted.

---

**Algorithm 2** SELECTACTION: Multi-Expert Action Selection

---

**Require:** Position  $i$ , actions  $\mathcal{A}_i$ , Global:  $(\hat{V}, N, W, \tilde{L})$ ; HP:  $(\rho_0, c, \gamma)$

**Output:** Selected action  $a \in \mathcal{A}_i$

- 1:  $u_i \leftarrow (1 + \log(1 + \frac{1}{|\mathcal{A}_i|} \sum_{a'} N_{i,a'}))^{-1}$   $\triangleright \in (0, 1]$ ; decays slowly with visits
  - 2:  $\rho \leftarrow [(1 - \rho_0) \cdot u_i/2, (1 - \rho_0)(1 - u_i/2), \rho_0]; e \sim \rho, e \in [3]$   $\triangleright$  Uncertainty-adaptive
  - 3: **if**  $e = 1$  **then**  $\triangleright$  Expert 1: UCB-based selection
  - 4:  $a \leftarrow \arg \max_{a' \in \mathcal{A}_i} [\hat{V}_{i,a'} + c\sqrt{\log t / N_{i,a'}}]$
  - 5: **else if**  $e = 2$  **then**  $\triangleright$  Expert 2: EXP3-based selection
  - 6:  $p_{a'} \leftarrow W_{i,a'} / \|W_i\|_1; a \sim \text{Categorical}(p)$
  - 7: **else**  $\triangleright$  Expert 3: FTRL-based selection
  - 8:  $a \leftarrow \arg \max_{a' \in \mathcal{A}_i} [-\tilde{L}_{i,a'} + \gamma\sqrt{N_{i,a'}}]$
  - 9: **end if**
- 

**Definition 5.1 (Decomposed Regret).** Under Assumption A.3 (Bounded Coupling), surrogate subproblem objectives  $\{g_k\}_{k=1}^K$  can be constructed (induced by the decomposition) such that the instantaneous regret  $\ell_t := r_t(x^*) - r_t(x_t)$  admits the upper bound:

$$\ell_t \leq \underbrace{\sum_{k=1}^K \left[ g_k((x^*)^{(k)}) - g_k(x_t^{(k)}) \right]}_{\text{Subproblem Regret } S_t} + \underbrace{C_t}_{\text{Coupling Error}} \quad (2)$$

where  $(x^*)^{(k)}$  and  $x_t^{(k)}$  denote the restriction of  $x^*$  and  $x_t$  to subproblem  $k$ , and the coupling error  $C_t$  captures non-local interactions across overlapping positions  $\mathcal{O}$ .

The surrogates  $g_k$  are constructed by fixing variables outside subproblem  $k$  to  $x^*$ ; we do *not* assume  $f$  decomposes additively. Assumption A.3 guarantees that modifying subproblem  $k$  affects  $f$  by at most  $L \cdot \delta(x, x') + C \cdot |S_k \cap \mathcal{O}|$ , which suffices for (2) and separates subproblem regret from coupling error (Appendix 12.7). Complete definition in 12.5.

**Theorem 5.2 (Regret Decomposition).** Let Assumptions A.1–A.4 hold. Under the decomposed regret structure of Definition 5.1, the average regret of D&L satisfies:

$$R_{\text{avg}}(T) \leq R_{\text{subproblem}}^{\text{avg}}(T) + R_{\text{overlap}}^{\text{avg}}(T) + R_{\text{local}}^{\text{avg}}(T) + C \quad (3)$$

where  $R_{\text{subproblem}}^{\text{avg}}(T) = \frac{1}{K} \sum_{k=1}^K \sum_e p_e R_{\text{alg}}^{(k,e)}(T)$  aggregates expert regret weighted by selection probabilities  $p_e$  over  $T_k$  rounds per subproblem ( $R_{\text{alg}}^{(k,e)}(T)$  is regret of expert  $e$  on subproblem  $k$ ),  $R_{\text{overlap}}^{\text{avg}}(T)$  captures averaged Lagrangian coordination cost (Theorem 4.5), and  $R_{\text{local}}^{\text{avg}}(T)$  accounts for gradient-free local refinement. The  $1/K$  normalization defines average regret ( $R_{\text{avg}}(T) := \frac{1}{K} R_{\text{total}}(T)$ ) and isolates per-subproblem learning.

*Proof Sketch* By Definition 5.1, the instantaneous regret decomposes as  $\ell_t \leq S_t + C_t$  where  $S_t$  is subproblem regret,  $C_t$  coupling error. Summing over  $T$  rounds and averaging over  $K$  subproblems, we bound each term separately. For the subproblem regret, Assumption A.3 permits constructing surrogate objectives  $g_k$  without requiring additive decomposition of  $f$  (see Appendix 12.7). Each subproblem  $k$  is updated over  $T_k$  rounds, where  $\sum_k T_k = T$ . (1) The per-subproblem regret inherits from each expert's no-regret guarantee; averaging and applying Cauchy-Schwarz ( $\sum_k \sqrt{T_k} \leq \sqrt{KT}$ ) under the  $1/K$  average. (2) The coupling error is bounded by Theorem 4.5, and (3) local refinement contributes  $O(LD\sqrt{dT/K})$  via standard zeroth-order analysis (Appendix 12.5). Full proof in Appendix 12.4.

**Corollary 5.3 (Explicit Regret Bound).** Let D&L use experts  $\{UCB, EXP3, FTRL\}$  with mixture weights  $(p_1, p_2, p_3)$ . With  $K$  subproblems of size  $d = n/K$ , overlap size  $Q$ , Lipschitz constant  $L$ , and domain diameter  $\Delta$ , the expected average regret satisfies:

$$\begin{aligned} \mathbb{E}[R_{\text{avg}}(T)] &\leq \underbrace{O(d\sqrt{T \log T})}_{\text{Subproblem learning}} + \underbrace{O(KQR_{\max}\sqrt{T})}_{\text{Overlap coordination}} \\ &\quad + \underbrace{O(L\Delta\sqrt{dT/K})}_{\text{Local refinement}} \end{aligned} \tag{4}$$

At optimal decomposition  $K^* = O(\sqrt{n/(QR_{\max})})$ , the bound simplifies to  $O(\sqrt{nQR_{\max}T \log T})$ .

The subproblem learning term combines contributions from each expert: UCB contributes  $O(d\sqrt{T \log T}/K)$ , EXP3 contributes  $O(dB\sqrt{T \log d}/C_{\text{clip}})$  (including the clipped-estimator bias; using  $\log d \leq \log T$  yields the rate  $O(d\sqrt{T \log T})$ ), and FTRL contributes  $O(d\sqrt{T \log d}/K)$ . In the worst case, the dominant term  $O(d\sqrt{T \log T})$  arises from EXP3, which benefits less from a  $1/K$  averaging gain but provides robustness to adversarial and misspecified rewards (see Appendix 12.14 for individual bounds).

**Interpretation.** The bound reveals three key insights: (1) *subproblem-independent scaling*—the dominant term depends on subproblem size  $d = n/K$ , not full problem size  $n$ , yielding exponential-to-polynomial reduction over naive combinatorial bandits with  $O(\sqrt{T|\mathcal{X}|})$  regret where  $|\mathcal{X}|$  may be  $n!$  or  $2^n$ ; (2) the “*price of coordination*”  $O(KQ\sqrt{T})$  is sublinear in  $T$ , so average coordination cost per round vanishes as  $T \rightarrow \infty$ ; (3) optimal decomposition  $K^* = O(\sqrt{n/(QR_{\max})})$  balances subproblem complexity against coordination overhead.

**Comparison to prior work.** Stochastic linear bandits Dani et al. (2008) achieve  $O(n^{3/2}\sqrt{T})$  under full-bandit feedback; classical combinatorial methods Chen et al. (2013); Kveton et al. (2015) achieve  $O(\sqrt{nkT \log T})$  under semi-bandit feedback ( $n$  base arms with  $k$  selected per round). D&L achieves  $O(d\sqrt{T \log T})$  under full-bandit feedback without linearity—where  $d = n/K$  is subproblem size. Naive combinatorial bandits incur  $O(\sqrt{T|\mathcal{X}|})$  where  $|\mathcal{X}|$  may be  $2^n$  or  $n!$ ; D&L’s structured decomposition reduces this to polynomial dependence on  $n$ . The multi-expert and coordination mechanisms ensure robustness without inflating the rate (Extended related work in Appendix 8).

## 6 Experiments

### 6.1 Datasets

For task of multi-objective optimization (MOO) Xue et al. (2024), our evaluation spans two complementary benchmarks differing in structure, scale, and evaluation cost.

**Multi-Objective Combinatorial Optimization (MOCO).** We consider multi-objective variants of classical NP-hard problems: Traveling Salesman Problem (TSP; up to 100 cities,  $n! \approx 10^{157}$ ), Knapsack (up to 200 items,  $2^n \approx 10^{60}$ ) Zitzler and Thiele (1999), and Capacitated Vehicle Routing Problem (CVRP; up to 100 customers) Jozefowicz et al. (2008), covering permutation, binary, and constrained combinatorial domains. These benchmarks enable direct comparison with state-of-the-art across diverse Pareto geometries and exponentially scaling search spaces.

**Hardware-Software Co-Design.** This real-world four-objective problem requires expensive simulation of neural network accelerators Dutta et al. (2022) over a 20-dimensional mixed-integer design space ( $\sim (2.2 \times 10^5)$  configurations). Each evaluation incurs substantial computational cost ( $O(\text{hours})$ ) and intrinsic stochasticity from hardware-level variance, presenting a challenging regime for sample-efficient optimization under noisy, expensive black-box feedback with unknown Pareto geometry.

Together, these benchmarks validate performance across search spaces ranging from  $10^5$  to  $10^{157}$  configurations, encompassing binary, permutation, and mixed-integer domains with heterogeneous evaluation costs.

## 6.2 Experimental Details

All BO baselines use GP surrogates with Matérn-5/2 kernels and ARD (Rasmussen and Williams, 2006); acquisition functions use Sobol QMC sampling Sobol (1967) with 64–128 MC samples. Objectives are normalized to  $[0, 1]$  using bounds computed from initial samples, with minimization transformed to maximization orientation for acquisition functions. All methods share identical initialization (50–100 random feasible solutions depending on problem size). Since MOCO evaluations are inexpensive, we tune each baseline individually and report *best-achievable* hypervolume. For D&L, we report results with two choices for Expert 1: UCB and Thompson Sampling (denoted D&L and D&L-TS respectively); the latter replaces FTRL with posterior sampling; weighted-sum scalarization is used for MOCO benchmarks, Tchebycheff for HW-SF (Appendix 9.1). Full hyperparameters in Appendix 10.5. For evaluation, we report the *hypervolume ratio*  $\text{HV}'_r(F) = \text{HV}_r(F) / \prod i=1^M |r_i - z_i|$  (Pareto front  $F$ , reference point  $r$ , ideal point  $z$ ) ensuring comparability across problem scales and instance sizes. We adopt standardized reference and ideal points, see Table 3. Each method is evaluated over 200 randomly generated problem instances, and we report mean hypervolume ratio with standard error.

## 6.3 Baselines

We compare D&L against baselines spanning five algorithmic paradigms. NSGA-II Deb et al. (2002) is a heuristic-based evolutionary method employing non-dominated sorting with crowding distance, testing whether D&L can match performance achieved through problem-specific operators. For Bayesian optimization, MOBO-qParEGO Knowles (2006) decomposes via random Chebyshev scalarizations, testing whether explicit decomposition outperforms bandit-based weight selection; MOBO-qNEHVI Daulton et al. (2021) maximizes expected hypervolume improvement, testing whether modeling objective correlations improves sample efficiency; and BOPR Daulton et al. (2022) handles discrete structures through probabilistic reparameterization—the most directly comparable baseline, we extend it to multi-objective settings to compare gradient-based acquisition vs. D&L. PPLS/D-C Shi et al. (2022) decomposes objective space into sub-regions with archive cooperation, testing parallel local search against learning-based methods. WS-LKH (40 wt.) and WS-DP Miettinen (1999)(101 wt.) apply weighted-sum scalarization with LKH Jaskiewicz (2002) and dynamic programming respectively, providing near-optimal references. PMOCO Lin et al. (2022) employs preference-conditioned hypernetworks trained via reinforcement learning; while requiring pretraining, we include it as a representative neural method. For fair comparison, we tune each baseline individually and report best achievable hypervolume. Problem-specific operators are employed where applicable Lacomme et al. (2004); Prins (2004); details in Appendix 10.3.

Table 1: Results across benchmarks (200 instances each), reporting Hypervolume (HV, ↑), number of non-dominated solutions (NDS, ↑), computational cost in Tera-FLOPs ( $\mathcal{F}$ , ↓).  $^\dagger$ Cost includes training; \*evaluated under matched compute budget, NA: no WS solver for CVRP. The vertical line separates specialized problem-specific solvers (left) from general-purpose methods (right).

WS-*				PPLS/D-C			PMOCO $^\dagger$			PMOCO*			NSGA-II			qNEHVI			qParEGO			PR			D&L			D&L-TS			
	HV	NDS	$\mathcal{F}$	HV	NDS	$\mathcal{F}$	HV	NDS	$\mathcal{F}$	HV	NDS	$\mathcal{F}$	HV	NDS	$\mathcal{F}$	HV	NDS	$\mathcal{F}$	HV	NDS	$\mathcal{F}$	HV	NDS	$\mathcal{F}$	HV	NDS	$\mathcal{F}$	HV	NDS	$\mathcal{F}$	
Bi-TSP	20	.625	21	.001	.626	71	.03	.509	69	5e3	.36	8	.68	.573	225	.15	.352	18	.24	.373	4	4.6	.536	10	4.4	.585	45	.01	.60	102	.01
	50	.629	26	.06	.628	213	.07	.550	76	24e3	.33	6	.734	.347	20	.15	.126	16	.29	.137	7	.274	.472	23	51	.530	62	.05	.54	147	.07
	100	.69	50	1.0	.63	533	.15	.67	86	105e3	.309	6	.685	.294	51	.31	.083	17	.37	.104	6	2462	.07	11	357	.40	48	.13	.47	114	.16
Bi-KP	50	.356	17	.003	.357	25	.01	.355	55	10e3	.34	40	.362	.219	68	.31	.301	28	.41	.29	8	4.4	.264	6	2.0	.35	45	.13	.347	71	.07
	100	.440	23	.02	.447	49	.02	.443	42	22e3	.37	34	.816	.221	24	.31	.25	18	.131	.314	7	.19	.304	10	3.7	.338	62	.28	.385	56	.13
	200	.36	30	.08	.362	63	.07	.354	52	47e3	.186	2	.583	.28	13	.32	.10	9	.590	.21	8	300	.266	13	9.9	.26	21	.39	.30	73	.22
Tri-TSP	20	.467	46	.008	.388	35	.01	.460	105	5e3	.43	40	51	.411	200	1.5	.23	68	3.9	.32	6	1.7	.296	45	5.1	.43	108	.11	.42	218	.02
	50	.429	189	.34	.262	93	.04	.420	105	24e3	.39	25	.236	.173	198	1.5	.052	69	4.9	.062	10	.87	.04	30	83	.262	172	.26	.282	494	.09
	100	.488	200	5.2	.241	158	.16	.44	104	104e3	.30	40	.654	.135	200	6.1	.024	107	6.4	.03	31	1408	.02	39	308	.21	223	.26	.234	310	.24
Bi-CVRP	20	NA	NA	NA	.48	7	.10	.36	8	6e3	.36	8	.398	.43	81	.15	.352	13	2.5	.21	4	1.8	.22	5	3.0	.45	22	.01	.46	36	.04
	50	NA	NA	NA	.45	21	.12	.346	6	28e3	.33	6	.571	.31	9	1.0	.14	7	14	.06	3	14	.052	8	1.3	.35	37	.12	.37	50	.10
	100	NA	NA	NA	.43	25	.14	.35	7	126e3	.30	6	.2e3	.31	14	3.5	.11	10	17	.11	9	128	.08	4	1	.253	17	.31	.30	42	.19

## 6.4 Experimental Results

We evaluate D&L across combinatorial benchmarks (Tables 1 and hardware-software co-design (Table 2), comparing against problem-specific solvers, neural methods, and general-purpose MOCO algorithms.

### 6.4.1 Multi-Objective Combinatorial Optimization

**Expected performance hierarchy.** Problem-specific solvers (WS-LKH/DP, PPLS/D-C) exploit domain structure through tailored heuristics and establish quality upper bounds. Neural methods (PMOCO) amortize computation through offline training on problem distributions, approaching specialized performance at inference. NSGA-II, while general-purpose, relies on problem-specific crossover and mutation operators for combinatorial search. In contrast, D&L operates as a pure black-box method—requiring only objective evaluations without domain operators or pretraining. Our goal is to approach the quality of specialized methods under this more restrictive setting.

**Solution quality.** D&L achieves the highest hypervolume among black-box methods across most benchmarks, reaching 80–98% of specialized solvers performance while outperforming Bayesian baselines by 20–90%. For fair comparison with neural methods, PMOCO\* matches training instances to our evaluation budget—under this setting, our approach exceeds offline training by 50–70% across problems. D&L’s advantage grows with scale: MOBO methods degrade 70–85% from small to large instances.

**Scaling to higher objectives.** In tri-objective settings, Bayesian methods collapse under the expanded Pareto front dimensionality. D&L’s bandit-based decomposition scales gracefully, discovering 2–3× more

non-dominated solutions than the strongest baselines with competitive HV.

**Computational efficiency.** To isolate algorithmic efficiency from hardware effects, we report analytical FLOPs in Tables 1; wall-clock times appear in the Appendix. FLOP ratios closely track runtime ratios across components (see 10.6, Fig 2). D&L reduces computation by 90-99% compared to Bayesian baselines, yielding to 10–30× wall-clock speedups (see Tables 5–8). This efficiency stems from closed-form acquisition updates that bypass expensive hypervolume computation and GP inference.

#### 6.4.2 Hardware-Software Co-Design

This benchmark presents a contrasting regime: a compact search space but expensive evaluations requiring neural accelerator simulation. We fix the budget to 150 evaluations to test sample efficiency under costly queries. D&L achieves the highest hypervolume, improving ∼22% over baselines on average (Table 2). This demonstrates that bandit-based exploration remains competitive in sample-limited regimes where GP surrogates should excel. Notably, qNEHVI struggles with the four-objective structure, where hypervolume acquisition cost scales exponentially with objective count.

**Ablations.** Bandit-based decomposition scales gracefully on large-scale, many-objective problems where baselines degrade. Appendix 11 validates key design choices: overlapping subproblem decomposition outperforms monolithic optimization (11.1)—notably, D&L can exceed specialized solvers on BITSP-20 Fig 3); the multi-experts framework (UCB, EXP3 etc.) outperforms any single expert (11.3); adaptive perturbation improves local search escape; and sample diversity analysis confirms broad trajectory coverage (11.4). Thompson sampling variants consistently outperform UCB, reflecting superior exploration-exploitation balance in combinatorial spaces as seen in Tables 1.

Table 2: Results (averaged over 10 seeds) on Hardware-Software Co-design (4 objectives).

Method	Hypervolume
NSGA-II	0.291 ± 0.040
MOBO-qNEHVI	0.287 ± 0.015
MOBO-qParEGO	0.34 ± 0.012
<b>D&amp;L</b>	<b>0.36 ± 0.01</b>
<b>D&amp;L-TS</b>	<b>0.372 ± 0.006</b>

## 7 Conclusion

We presented D&L, a decomposition-based multi-expert framework for combinatorial optimization under full-bandit feedback. The framework is modular: any no-regret experts suffice, and decomposition strategies can be tailored to problem structure. The key requirement—bounded coupling—holds when problems exhibit locality (geographic proximity for TSP, temporal adjacency for scheduling); problems with dense global interactions may benefit less. Our ablations reveal a tradeoff between subproblem count  $K$  and overlap  $Q$ ; balancing per  $K^* = O(\sqrt{n/(QR_{\max})})$  keeps coordination overhead sublinear. Empirically, D&L matches or exceeds specialized solvers on MOCO benchmarks in 2–10× less time, scales to tri-objective settings where Bayesian methods degrade, and remains competitive on expensive black-box problems—all without problem-specific operators or pretraining. Current limitations include fixed mixture weights  $\rho$  (adaptive selection could improve constants) and bounded rewards (sub-Gaussian extensions remain open). The results establish decomposed

bandit optimization as principled approach to sample-efficient MOCO without requiring surrogate models or offline training.

## References

- Angioni, D., Archetti, C., and Speranza, M. G. (2025). Neural combinatorial optimization: A tutorial. *Computers & Operations Research*, page 107102.
- Arora, S., Hazan, E., and Kale, S. (2012). The multiplicative weights update method: A meta-algorithm and applications. *Theory of Computing*, 8(1):121–164.
- Audibert, J.-Y., Bubeck, S., and Lugosi, G. (2014). Regret in online combinatorial optimization. *Mathematics of Operations Research*, 39(1):31–45.
- Auer, P., Cesa-Bianchi, N., and Fischer, P. (2002a). Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47(2):235–256.
- Auer, P., Cesa-Bianchi, N., Freund, Y., and Schapire, R. E. (2002b). The nonstochastic multiarmed bandit problem. *SIAM Journal on Computing*, 32(1):48–77.
- Balandat, M., Karrer, B., Jiang, D. R., Daulton, S., Letham, B., Wilson, A. G., and Bakshy, E. (2020). Botorch: A framework for efficient monte-carlo bayesian optimization.
- Baptista, R. and Poloczek, M. (2018). Bayesian optimization of combinatorial structures. In *International conference on machine learning*, pages 462–471. PMLR.
- Bazgan, C., Hugot, H., and Vanderpoorten, D. (2009). Solving efficiently the 0-1 multi-objective knapsack problem. *Computers & Operations Research*, 36(1):260–279.
- Beck, A. and Teboulle, M. (2003). Mirror descent and nonlinear projected subgradient methods for convex optimization. *Operations Research Letters*, 31(3):167–175.
- Bello, I., Pham, H., Le, Q. V., Norouzi, M., and Bengio, S. (2016). Neural combinatorial optimization with reinforcement learning. *arXiv preprint arXiv:1611.09940*.
- Boyd, S. and Vandenberghe, L. (2004). *Convex optimization*. Cambridge university press.
- Boyd, S., Xiao, L., and Mutapcic, A. (2003). Subgradient methods. Technical report, Stanford University. Lecture notes for EE364B.
- Brown, N., Fiscato, M., Segler, M. H., and Vaucher, A. C. (2019). Guacamol: benchmarking models for de novo molecular design. *Journal of chemical information and modeling*, 59(3):1096–1108.
- Bubeck, S. (2015). Convex optimization: Algorithms and complexity.
- Bubeck, S. and Cesa-Bianchi, N. (2012). Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *arXiv preprint arXiv:1204.5721*.
- Cesa-Bianchi, N. and Lugosi, G. (2006). *Prediction, Learning, and Games*. Cambridge University Press.

- Cesa-Bianchi, N. and Lugosi, G. (2012). Combinatorial bandits. *Journal of Computer and System Sciences*, 78(5):1404–1422.
- Chen, J., Zhang, Z., Cao, Z., Wu, Y., Ma, Y., Ye, T., and Wang, J. (2023). Neural multi-objective combinatorial optimization with diversity enhancement. *arXiv preprint arXiv:2310.15195*.
- Chen, W., Wang, Y., and Yuan, Y. (2013). Combinatorial multi-armed bandit: General framework and applications. In *Proceedings of the 30th International Conference on Machine Learning*, pages 151–159.
- Cheng, R., Jin, Y., Olhofer, M., and Sendhoff, B. (2016). A reference vector guided evolutionary algorithm for many-objective optimization. *IEEE Transactions on Evolutionary Computation*, 20(5):773–791.
- Combes, R., Talebi, M. S., Proutiere, A., and Lelarge, M. (2015). Combinatorial bandits revisited. *arXiv preprint arXiv:1502.03475*.
- Couckuyt, I., Deschrijver, D., and Dhaene, T. (2014). Fast calculation of multiobjective probability of improvement and expected improvement criteria for Pareto optimization. *Journal of Global Optimization*, 60(3):575–594.
- Dani, V., Hayes, T. P., and Kakade, S. M. (2008). Stochastic linear optimization under bandit feedback. In *Annual Conference Computational Learning Theory*.
- Daulton, S., Balandat, M., and Bakshy, E. (2021). Parallel bayesian optimization of multiple noisy objectives with expected hypervolume improvement. In Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W., editors, *Advances in Neural Information Processing Systems*, volume 34, pages 2187–2200.
- Daulton, S., Wan, X., Eriksson, D., Balandat, M., Osborne, M. A., and Bakshy, E. (2022). Bayesian optimization over discrete and mixed spaces via probabilistic reparameterization. *arXiv preprint arXiv:2210.10199*.
- Davis, L. (1985). Applying adaptive algorithms to epistatic domains. In *Proceedings of the 9th International Joint Conference on Artificial Intelligence - Volume 1*, IJCAI'85, page 162–164. Morgan Kaufmann Publishers Inc.
- Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197.
- Dutta, A., Gupta, S., Khaleghi, B., Chandrasekaran, R., Xu, W., and Rosing, T. (2022). HDnn-PIM: Efficient in-memory design of hyperdimensional computing with feature extraction. In *Proceedings of the Great Lakes Symposium on VLSI*.
- Ehrgott, M. (2005). *Multicriteria Optimization*. Springer.
- Fialho, Á., Da Costa, L., Schoenauer, M., and Sebag, M. (2010). Analyzing bandit-based adaptive operator selection mechanisms. *Annals of Mathematics and Artificial Intelligence*, 60(1):25–64.
- Fisher, M. L. (1981). The lagrangian relaxation method for solving integer programming problems. *Management Science*, 27(1):1–18.
- Flaxman, A. D., Kalai, A. T., and McMahan, H. B. (2005). Online convex optimization in the bandit setting: Gradient descent without a gradient. In *Proceedings of the Sixteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 385–394.

- Florios, K. and Mavrotas, G. (2014). Generation of the exact Pareto set in multi-objective traveling salesman and set covering problems. *Applied Mathematics and Computation*, 237:1–19.
- Freund, Y. and Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139.
- Garrido-Merchán, E. C. and Hernández-Lobato, D. (2020). Dealing with categorical and integer-valued variables in bayesian optimization with gaussian processes. *Neurocomputing*, 380:20–35.
- Hazan, E. (2016a). Introduction to online convex optimization. *Foundations and Trends® in Optimization*, 2(3-4):157–325.
- Hazan, E. (2016b). Introduction to online convex optimization. *Foundations and Trends® in Optimization*, 2(3-4):157–325.
- Helsgaun, K. (2000). An effective implementation of the Lin–Kernighan traveling salesman heuristic. *European Journal of Operational Research*, 126(1):106–130.
- Huang, Z., Zhou, Y., Luo, C., and Lin, Q. (2021). A runtime analysis of typical decomposition approaches in moea/d framework for many-objective optimization problems. In *IJCAI*, volume 21, pages 1682–1688.
- Ishibuchi, H., Akedo, N., and Nojima, Y. (2015). Behavior of multiobjective evolutionary algorithms on many-objective knapsack problems. *IEEE Transactions on Evolutionary Computation*, 19(2):264–283.
- Ito, S., Hatano, D., Sumita, H., Takemura, K., Fukunaga, T., Kakimura, N., and Kawarabayashi, K.-I. (2019). Improved regret bounds for bandit combinatorial optimization. In *Advances in Neural Information Processing Systems*, volume 32.
- Jaszkiewicz, A. (2002). Genetic local search for multi-objective combinatorial optimization. *European Journal of Operational Research*, 137(1):50–71.
- Jiang, S., Cai, Z., Zhang, J., and Ong, Y.-S. (2011). Multiobjective optimization by decomposition with Pareto-adaptive weight vectors. In *International Conference on Natural Computation (ICNC)*, pages 1260–1264. IEEE.
- Jozefowicz, N., Semet, F., and Talbi, E.-G. (2008). From single-objective to multi-objective vehicle routing problems: Motivations, case studies, and methods. In *The Vehicle Routing Problem: Latest Advances and New Challenges*, pages 445–471. Springer.
- Kendall, M. G. (1938). A new measure of rank correlation. *Biometrika*, 30(1-2):81–93.
- Khalil, E., Dai, H., Zhang, Y., Dilkina, B., and Song, L. (2017). Learning combinatorial optimization algorithms over graphs. *Advances in neural information processing systems*, 30.
- Knowles, J. (2006). ParEGO: A hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems. *IEEE Transactions on Evolutionary Computation*, 10(1):50–66.
- Kool, W., van Hoof, H., and Welling, M. (2019). Attention, learn to solve routing problems! In *International Conference on Learning Representations*.

- Krishnan, S., Yazdanbaksh, A., Prakash, S., Jabbour, J., Uchendu, I., Ghosh, S., Boroujerdian, B., Richins, D., Tripathy, D., Faust, A., and Reddi, V. J. (2023). Archgym: An open-source gymnasium for machine learning assisted architecture design.
- Kveton, B., Wen, Z., Ashkan, A., and Szepesvari, C. (2015). Tight Regret Bounds for Stochastic Combinatorial Semi-Bandits. In *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics*, pages 535–543.
- Kwon, Y.-D., Choo, J., Kim, B., Yoon, I., Gwon, Y., and Min, S. (2021). Pomo: Policy optimization with multiple optima for reinforcement learning.
- Lacomme, P., Prins, C., and Ramdane Cherif-Khettaf, W. (2004). Competitive memetic algorithms for arc routing problems. *Annals Operations Research*, 131.
- Lagos, F., , and Pereira, J. (2023). Multi-armed bandit-based hyper-heuristics for combinatorial optimization problems. *European Journal of Operational Research*.
- Letham, B., Karrer, B., Ottoni, G., and Bakshy, E. (2018). Constrained bayesian optimization with noisy experiments.
- Li, Y.-L., Zhou, Y.-R., Zhan, Z.-H., and Zhang, J. (2015). A primary theoretical study on decomposition-based multiobjective evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 20(4):563–576.
- Lin, X., Yang, Z., and Zhang, Q. (2022). Pareto set learning for neural multi-objective combinatorial optimization. *arXiv preprint arXiv:2203.15386*.
- Lovász, L. (1983). Submodular functions and convexity. In *Mathematical Programming The State of the Art: Bonn 1982*, pages 235–257. Springer.
- Lust, T. and Teghem, J. (2010). The multiobjective traveling salesman problem: A survey and a new approach. In *Studies in Computational Intelligence*, volume 272, pages 119–141. Springer.
- Miettinen, K. (1999). *Nonlinear multiobjective optimization*, volume 12. Springer Science & Business Media.
- Močkus, J. (1975). On Bayesian methods for seeking the extremum. In *Optimization Techniques IFIP Technical Conference*, pages 400–404. Springer.
- Oh, C., Tomczak, J., Gavves, E., and Welling, M. (2019). Combinatorial bayesian optimization using the graph cartesian product. *Advances in Neural Information Processing Systems*, 32.
- Öner, D., Karakurt, A., Eryilmaz, A., and Tekin, C. (2018). Combinatorial multi-objective multi-armed bandit problem. *arXiv preprint arXiv:1803.04039*.
- Paquete, L. and Stützle, T. (2003). A two-phase local search for the biobjective traveling salesman problem. In *Proceedings of the 2nd International Conference on Evolutionary Multi-Criterion Optimization*, pages 479–493. Springer.
- Prins, C. (2004). A simple and effective evolutionary algorithm for the vehicle routing problem. *Computers & Operations Research*, 31(12).

- Qi, Y., Ma, X., Liu, F., Jiao, L., Sun, J., and Wu, J. (2013). MOEA/D with adaptive weight adjustment. *Evolutionary Computation*, 22.
- Rasmussen, C. E. and Williams, C. K. I. (2006). *Gaussian Processes for Machine Learning*. MIT Press.
- Romero, P. A., Krause, A., and Arnold, F. H. (2013). Navigating the protein fitness landscape with gaussian processes. *Proceedings of the National Academy of Sciences*, 110(3):E193–E201.
- Savelsbergh, M. (1985). Local search in routing problems with time windows. *Annals of Operations Research*.
- Saves, P., Diouane, Y., Bartoli, N., Lefebvre, T., and Morlier, J. (2023). A mixed-categorical correlation kernel for gaussian process. *Neurocomputing*, 550:126472.
- Schrijver, A. (2003). *Combinatorial Optimization: Polyhedra and Efficiency*. Springer.
- Shalev-Shwartz, S. (2012). *Online Learning and Online Convex Optimization*. Foundations and Trends® in Machine Learning Series. Now Publishers.
- Shi, J., Sun, J., Zhang, Q., Zhang, H., and Fan, Y. (2022). Improving Pareto local search using cooperative parallelism strategies for multiobjective combinatorial optimization. *IEEE Transactions on Cybernetics*, 52(12):13447–13458.
- Singh, E., Sabach, S., and Wang, Y.-X. (2025). Moxco: How i learned to stop exploring and love my local minima? In *The Second Conference on Parsimony and Learning (Proceedings Track)*.
- Sobol, I. M. (1967). The distribution of points in a cube and the accurate evaluation of integrals. *USSR Computational Mathematics and Mathematical Physics*, 7(4):86–112.
- Sutton, R. S. and Barto, A. G. (2018). *Reinforcement Learning: An Introduction*. MIT Press, 2nd edition.
- Tinós, R., Helsgaun, K., and Whitley, D. (2018). Efficient recombination in the Lin-Kernighan-Helsgaun traveling salesman heuristic. In *Parallel Problem Solving from Nature – PPSN XV*, pages 95–107. Springer.
- Trivedi, A., Srinivasan, D., Sanyal, K., and Ghosh, A. (2017). A survey of multiobjective evolutionary algorithms based on decomposition. *IEEE Transactions on Evolutionary Computation*, 21(3):440–462.
- Williams, R. J. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8.
- Xue, K., Tan, R.-X., Huang, X., and Qian, C. (2024). Offline multi-objective optimization. *arXiv preprint arXiv:2406.03722*.
- Yang, C.-Y., Zhou, M., Ponzina, F., Sathya Prakash, S., Ayoub, R., Mercati, P., Subedar, M., and Rosing, T. (2025). Multi-objective software-hardware co-optimization for hd-pim via noise-aware bayesian optimization. In *Proceedings of the 43rd IEEE/ACM International Conference on Computer-Aided Design*, ICCAD ’24. Association for Computing Machinery.
- Zhang, Q. and Li, H. (2007). MOEA/D: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation*, 11(6):712–731.
- Zhao, G., Arroyave, R., and Qian, X. (2019). Fast exact computation of expected hypervolume improvement. *arXiv preprint arXiv:1812.07692*.

Zitzler, E. and Thiele, L. (1999). Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach. *IEEE Transactions on Evolutionary Computation*, 3(4):257–271.

# Appendix

---

<b>8 Related Work</b>	<b>21</b>
<b>9 Why Weighted-Sum methods are not feasible?</b>	<b>23</b>
9.1 Scalarization in D&L	23
<b>10 Additional Experimental details</b>	<b>24</b>
10.1 Datasets	24
10.1.1 <i>Multi-Objective Combinatorial Optimization</i>	24
10.2 Real-World Application: Hardware–Software co-design	24
10.3 Baseline Algorithm Details	25
10.4 Evaluation Methodology	27
10.5 Hyperparameter and Tuning used for reporting results	29
10.6 Extended Results	30
10.6.1 <i>Hardware and Compute</i>	30
10.6.2 <i>Algorithmic Computational Cost Estimation</i>	30
10.7 Algorithmic Cost vs. Wall-Clock Time	31
<b>11 Ablation Studies</b>	<b>33</b>
11.1 Subproblems and Decomposition Size	33
11.2 Multi-Expert Selection: Why and How?	38
11.3 Experts: Minimal Set of Experts	41
11.3.1 <i>Setup 1: Expert 1 Ablation (UCB vs. Thompson Sampling)</i>	41
11.3.2 <i>Setup 2: Expert 3 Ablation (FTRL Necessity)</i>	41
11.3.3 <i>Setup 3: Expert 2 Ablation (EXP3 study)</i>	43
11.3.4 <i>Can there be more than three experts?</i>	48
11.4 Sample Diversity	48
11.4.1 <i>Evaluation Metrics</i>	48
11.4.2 <i>Results on BiTSP-50</i>	50
11.4.3 <i>Summary</i>	51
11.5 Hyperparameter Sensitivity	53
11.6 Lagrangian Coordination	54
<b>12 Algorithm Regret Bounds</b>	<b>58</b>
12.1 Notation	58
12.2 Assumptions	58
12.2.1 <i>Assumption 1 (Decomposability)</i>	59
12.2.2 <i>Assumption 2 (Lipschitz Continuity)</i>	59
12.2.3 <i>Assumption 3 (Bounded Coupling)</i>	59
12.2.4 <i>Assumption 4 (Bounded Range)</i>	59
12.2.5 <i>Assumption A.5' (Local Additivity + Exogeneity for Position–Value UCB)</i>	59
12.3 Online Game	60
12.3.1 <i>Hybrid Algorithm Structure</i>	60
12.4 Algorithm Total Regret Bound	61
12.4.1 <i>Key Takeaway</i>	64

12.5	Local Search Regret Bound	66
12.6	Global Experts Regret Bound	68
12.6.1	UCB Regret	69
12.6.2	Decomposed subproblem regret	69
12.6.3	Coupling Error from Overlapping Positions	71
12.6.4	Exp3 Regret	75
12.6.5	FTRL Regret	83
12.7	Decomposition Validity	85
12.7.1	Decomposition Construction	85
12.7.2	Local Stability	86
12.7.3	Overlap Structure and Coupling	87
12.7.4	Validity of Decomposition	88
12.7.5	Diminishing Overlap	89
12.7.6	Domain-Agnostic Local Operators	90
12.7.7	Summary	90
12.8	Proof Appendix	91
12.8.1	UCB Subproblem Regret Proof	91
12.8.2	Exp3 Subproblem Regret Proof	92
12.8.3	Follow The Regularized Leader (FTRL)	94
12.8.4	Coupling Error: Lagrangian coordination	96
12.8.5	Coupling Error: Disagreement requires uncertainty	100

---

## 8 Related Work

**Neural Combinatorial Optimization.** Neural combinatorial optimization (NCO) methods learn solution construction policies, most commonly via policy-gradient reinforcement learning. Attention-based models trained with REINFORCE [Bello et al. \(2016\)](#); [Kool et al. \(2019\)](#) achieve strong performance on single-objective CO. Recent multi-objective extensions include Pareto set learning with preference-conditioned models (PMOCO) [Lin et al. \(2022\)](#), diversity-enhanced neural heuristics (NHDE) [Chen et al. \(2023\)](#), and related neural combinatorial optimization approaches that amortize search across instances [Khalil et al. \(2017\)](#). While these approaches can amortize inference at test time, they typically require substantial offline training data and computational resources to generalize across instance distributions, and may require retraining or fine-tuning when objectives, constraints, or instance families shift [Angioni et al. \(2025\)](#). In contrast, D&L requires no pretraining and adapts online using bandit feedback alone, making it naturally suited to settings with expensive evaluations, limited data budgets, or rapidly varying instance structure.

**Relationship to Reinforcement Learning.** Multi-armed bandits can be viewed as a special case of reinforcement learning: a one-state (stateless) MDP with no state transitions and immediate rewards [Sutton and Barto \(2018\)](#). D&L fits this regime—each round selects a complete solution and observes a single scalar reward—so the learning problem is fundamentally *credit assignment without temporal dynamics*. This contrasts with neural CO methods, which typically cast solution construction as a sequential decision process and train a parameterized policy over action prefixes using policy gradients (often REINFORCE with a rollout baseline), i.e., an actor-only or actor-critic-style training loop. Technically, D&L differs from actor-critic RL in three ways: (i) **no value-function learning**: there is no learned critic estimating long-horizon returns because the objective is

evaluated on the constructed solution (one-step return); (ii) **no differentiable policy optimization**: actions are selected by index/weight updates (UCB confidence bounds, EXP3/FTRL multiplicative weights) rather than gradients through a neural policy; and (iii) **structured action space exploitation**: D&L decomposes the decision into coupled subproblems and coordinates them via Lagrangian multipliers, analogous in spirit to factorization ideas but without modeling transition dynamics. Thus, D&L is best interpreted as an online learning/bandit formulation for solution construction (with regret guarantees), whereas NCO uses offline RL to amortize a policy across many instances, typically without finite-time regret guarantees.

**Combinatorial Bandits.** Online convex optimization [Hazan \(2016a\)](#) and multi-armed bandits [Auer et al. \(2002a,b\)](#) provide principled frameworks for sequential decision-making under uncertainty. The combinatorial multi-objective MAB (COMO-MAB) [Öner et al. \(2018\)](#) is most related to our setting, achieving  $O(NL^3 \log T)$  Pareto regret. However, COMO-MAB assumes *linear* reward structure (action rewards are linear combinations of base-arm rewards), *semi-bandit* feedback (observing per-arm rewards), finitely many base arms, and stationary i.i.d. rewards—assumptions substantially stronger than ours. In classical stochastic combinatorial MAB methods, CUCB [Chen et al. \(2013\)](#) and subsequent work [Kveton et al. \(2015\)](#) achieve gap-free regret bounds on the order of  $\tilde{O}(\sqrt{KLT})$ ; adversarial variants [Cesa-Bianchi and Lugosi \(2012\)](#); [Audibert et al. \(2014\)](#) address non-stochastic settings with similar bounds. Both lines require semi-bandit feedback, which is unavailable when only aggregate solution quality is observed. Our formulation goes beyond these settings in three key respects: (i) combinatorial (often exponentially large) action spaces built from finite base arms, (ii) full-bandit feedback rather than semi-bandit, and (iii) nonlinear scalarized rewards rather than linear additive reward models. Recent work on adversarial combinatorial optimization [Ito et al. \(2019\)](#); [Audibert et al. \(2014\)](#) shows that correlation among losses can significantly impact achievable regret; our decision-space decomposition explicitly mitigates such dependencies by isolating subproblems, yielding regret that scales with subproblem dimension  $d$  rather than full problem size  $n$ .

**Exploration-Exploitation Tradeoffs.** Balancing exploration and exploitation is central to optimization across domains. This tradeoff can be exploited in adaptive optimizers for deep neural networks, to design methods that navigate loss landscapes toward generalizing solutions [Singh et al. \(2025\)](#). Here, we address the analogous challenge in combinatorial optimization: exploring exponential search spaces efficiently while exploiting promising regions. Our multi-expert framework instantiates this tradeoff through complementary no-regret algorithms—UCB for optimistic exploration [Auer et al. \(2002a\)](#), Exp3 for adversarial robustness [Auer et al. \(2002b\)](#), and FTRL for regularized stability [Hazan \(2016a\)](#)—coordinated via Lagrangian decomposition [Fisher \(1981\)](#).

**Multi-expert and algorithm selection.** Bandits have been used for hyper-heuristic operator selection in evolutionary algorithms [Lagos et al. \(2023\)](#); [Fialho et al. \(2010\)](#), where a bandit chooses among crossover or mutation operators. This differs fundamentally from D&L: we use bandits for *solution construction* (position-wise action selection), not operator selection. The multi-expert framework relates to EXP4 [Auer et al. \(2002b\)](#) (bandits with expert advice), but EXP4 maintains a meta-layer weighting over experts. D&L instead samples directly from complementary algorithms  $\{\text{UCB}, \text{EXP3}, \text{FTRL}\}$  and updates *all* experts after each observation— inheriting the best expert’s guarantee without the  $O(\sqrt{N})$  cost of  $N$  experts.

Unlike contextual bandits where contexts are exogenous, D&L exploits *endogenous* structure through overlapping subproblem decompositions, creating a fundamentally different information architecture. This hybrid

framework—synthesizing bandit-style learning with structured decomposition—enables tractable optimization over exponential spaces while maintaining sublinear regret guarantees (Section 12.4).

To our knowledge, no prior work combines: (i) decision-space decomposition with regret guarantees, (ii) multi-expert bandit construction under full-bandit feedback, and (iii) Lagrangian coordination with provable  $O(\sqrt{T})$  coupling error. The superficial similarity to MOEA/D (both use “decomposition”) obscures a fundamentally different problem formulation: MOEA/D is an evolutionary heuristic decomposing objectives; D&L is an online learning algorithm decomposing decisions.

## 9 Why Weighted-Sum methods are not feasible?

Weighted-sum approaches, while prevalent in multi-objective optimization, suffer from fundamental limitations that restrict their applicability to general combinatorial problems. First, existing weighted-sum methods like WS-LKH and PPLS/D rely on problem-specific heuristics (e.g., Lin-Kernighan for TSP, branch-and-bound for knapsack) that are hand-crafted for particular combinatorial structures. These algorithms achieve state-of-the-art performance precisely because they exploit domain-specific properties—TSP’s geometric structure, knapsack’s fractional relaxation—that do not transfer to other problem classes. Developing such specialized solvers for every new combinatorial domain is computationally prohibitive and requires extensive domain expertise. Second, the weighted-sum scalarization inherently restricts exploration to the convex hull of the Pareto front. For minimization problems, only solutions on convex portions of the Pareto front can be found, regardless of the weights chosen. Non-convex regions, which often contain practically important trade-offs, remain inaccessible. This limitation is particularly severe in combinatorial spaces where Pareto fronts are typically highly non-convex due to the discrete nature of solutions. Third, weighted-sum methods require solving the scalarized problem to optimality for each weight vector, which becomes intractable for large-scale combinatorial problems without domain-specific algorithms. In contrast, our online learning framework provides a domain-agnostic approach that explores the entire Pareto front through adaptive decomposition and multi-expert strategies, without requiring problem-specific heuristics or convexity assumptions.

### 9.1 Scalarization in D&L

Our framework is agnostic to the choice of scalarizing function—the bandit-based decomposition operates on any scalar objective derived from the multi-objective vector. We briefly review the two scalarizations used in our experiments.

**Weighted-sum scalarization** combines objectives linearly:

$$g_{ws}(x \mid w) = \sum_{i=1}^m w_i f_i(x), \quad w \in \Delta^{m-1} \quad (5)$$

where  $\Delta^{m-1}$  denotes the  $(m-1)$ -simplex. This is the simplest scalarization but can only recover Pareto-optimal points on the convex hull of the front.

**Chebyshev (Tchebycheff) scalarization** uses the weighted infinity-norm:

$$g_{tch}(x \mid w, z^*) = \max_{i \in \{1, \dots, m\}} \{w_i |f_i(x) - z_i^*|\} \quad (6)$$

where  $z^* = (z_1^*, \dots, z_m^*)$  is a reference point (typically the ideal point). Unlike weighted-sum, Chebyshev scalarization can recover non-convex Pareto regions.

**Choice in experiments.** For combinatorial benchmarks (TSP, knapsack, CVRP), we use weighted-sum—the simplest scalarization—demonstrating that strong performance stems from our decomposition and bandit framework rather than sophisticated scalarization. For hardware-software co-design, we use Chebyshev scalarization to handle non-convex trade-offs common in four-objective design spaces. Switching scalarizations is trivial—only the scalar objective function changes while the bandit-based subproblem optimization remains unchanged. The key contribution is the online learning framework over decomposed subproblems, not the scalarization itself.

## 10 Additional Experimental details

### 10.1 Datasets

#### 10.1.1 Multi-Objective Combinatorial Optimization

Multi-objective combinatorial optimization (MOCO) commonly exists in industries, such as transportation, manufacturing, energy, and telecommunication (Chen et al., 2023b). We consider three typical MOCO problems that are commonly studied: multi-objective traveling salesman problem (MO-TSP) Paquete and Stützle (2003); Florios and Mavrotas (2014), multi-objective knapsack problem (MO-KP), and multi-objective capacitated vehicle routing problem (MO-CVRP).

**MO-TSP** In the  $m$ -objective TSP Lust and Teghem (2010) with  $n$  cities, each city has  $m$  different 2D coordinate sets, where coordinates are sampled uniformly from  $[0, 1]^2$ . Each objective  $f_i$  minimizes the total Euclidean distance of a tour using the  $i$ -th coordinate set. We evaluate both bi-objective (BiTSP) and tri-objective (TriTSP) variants, with solution space  $|\mathcal{X}| = n!$ . The objectives minimized are the tour lengths  $f_i(\mathbf{x}) = \sum_{j=1}^n d_i(x_j, x_{j+1})$ , where  $d_i(\cdot, \cdot)$  denotes the Euclidean distance using the  $i$ -th coordinate set and  $x_{n+1} = x_1$ .

**MO-KP** The  $m$ -objective knapsack problem Bazgan et al. (2009) consists of  $n$  items with weights  $w_j \sim \mathcal{U}(0, 1)$  and  $m$  value sets where  $v_{ij} \sim \mathcal{U}(0, 1)$  represents the value of item  $j$  for objective  $i$ . The goal is to maximize  $f_i(\mathbf{x}) = \sum_{j=1}^n v_{ij}x_j$  for each objective while satisfying the capacity constraint  $\sum_{j=1}^n w_jx_j \leq C$ . The solution space has cardinality  $|\mathcal{X}| = 2^n$ . The knapsack capacity is set to  $C \in \{12.5, 25, 25\}$  for problem sizes  $n \in \{50, 100, 200\}$  respectively.

**MO-CVRP** The bi-objective CVRP Lacomme et al. (2004); Prins (2004) consists of  $n$  customers and one depot, with all node coordinates sampled uniformly from  $[0, 1]^2$ . The depot is positioned at  $(0.5, 0.5)$ . Each customer  $j$  has an integer demand  $d_j \sim \mathcal{U}\{1, \dots, 9\}$ . The two objectives are: (1) minimizing the total distance of all routes  $f_1(\mathbf{x}) = \sum_{r \in \mathbf{x}} \text{length}(r)$ , and (2) minimizing the makespan  $f_2(\mathbf{x}) = \max_{r \in \mathbf{x}} \text{length}(r)$ , where  $r$  denotes a route. Following convention in [] we consider the instances with different sizes where each vehicle has capacity  $D \in \{30, 40, 50\}$  for problem sizes  $n \in \{20, 50, 100\}$  respectively.

[Note on capacity] why we increased it in some and why standard in others.

### 10.2 Real-World Application: Hardware-Software co-design

HDnn-PIM Dutta et al. (2022) is an edge AI accelerator design that supports the HDnn algorithm. Designing HDnn-PIM requires choosing both software (model parameters, etc.) and hardware (systolic array size, etc.) design parameters. The choice of the design parameters then contributes to various design metrics, including energy, delay, area (PPA), and accuracy, in a black-box manner, which is only attainable through expensive simulation. Additionally, the architecture consists of components with stochastic nature (PIM) that results

into noisy evaluation. The goal of the hardware-software co-design problem is to minimize PPA and maximize accuracy within trade-offs through hardware and software design parameter choices.

### 10.3 Baseline Algorithm Details

**WS-LKH/DP (Weighted Sum Scalarization).** We implement weighted sum scalarization, a classical *a priori* multi-objective optimization technique that converts a vector-valued objective into a series of single-objective subproblems via linear aggregation [Miettinen \(1999\)](#). Although weighted sum scalarization cannot recover unsupported Pareto-optimal solutions in non-convex fronts (see discussion [9](#)), it remains a widely used baseline due to its simplicity and interpretability. For MO-TSP, we employ WS-LKH, combining weighted sum scalarization with two-opt local search inspired by the Lin-Kernighan-Helsgaun algorithm [Helsgaun \(2000\)](#); [Tinós et al. \(2018\)](#). Given  $\lambda \in [0, 1]$  sampled uniformly across  $K$  weights, each scalarized instance uses combined distance matrix  $D_\lambda = \lambda D_1 + (1 - \lambda) D_2$  for bi-objective problems (extended to three weights summing to 1 for tri-objective). The algorithm generates random initial tours and applies iterative 2-opt improvements until local optimality. For MO-KP, we implement WS-DP, solving each scalarized knapsack exactly via dynamic programming. Item weights are scaled by factor 30 to enable integer-based DP while maintaining precision. For each  $\lambda$ , it solves  $\max_x \sum_i [\lambda v_{i1} + (1 - \lambda) v_{i2}] x_i$  subject to capacity constraints using standard DP recurrence with backtracking for solution reconstruction. Both methods aggregate solutions across all weight combinations and retain only the non-dominated set. We use  $K = 100$  for knapsack and vary  $K$  by instance size for TSP, including extreme weights to ensure Pareto front boundary coverage.

**PPLS/D-C (Parallel Pareto Local Search with Decomposition).** We implement PPLS/D-C [Shi et al. \(2022\)](#), a parallel local search that decomposes the objective space into subregions explored concurrently with periodic archive cooperation. The space is partitioned into  $P$  regions, each defined by weight vector  $\boldsymbol{\lambda}_p$  inducing preference via weighted sum scalarization. Each process maintains a local archive and explores its region using problem-specific operators: 2-opt for TSP, bit-flip and swap moves for knapsack with feasibility checking, and intra-route 2-opt plus inter-route relocation for CVRP. Cooperation occurs every  $\tau$  iterations: processes import compatible solutions from the shared archive and export their top-ranked local solutions. A master process monitors coverage and dynamically reassigns regions to sparse areas of the Pareto front. Initialization combines constructive heuristics (nearest-neighbor for TSP, greedy by efficiency for knapsack) with random solutions. The final output is the non-dominated subset of the merged archive.

**MOBO-NSGA-II (Evolutionary Multi-Objective Optimization).** We implement NSGA-II [Deb et al. \(2002\)](#) as our evolutionary baseline, which balances convergence and diversity through non-dominated sorting and crowding distance. The algorithm maintains a population that evolves through selection, crossover, and mutation, with environmental selection based on Pareto dominance rank and crowding distance to balance convergence and diversity. Non-dominated sorting partitions the population into fronts  $F_1, F_2, \dots$  where solutions in  $F_k$  are dominated only by solutions in earlier fronts. Crowding distance measures local density in objective space, computed as the average side length of the cuboid formed by neighboring solutions. Binary tournament selection favors lower rank, with crowding distance as tiebreaker. Following established practices [Jaszkiewicz \(2002\)](#); [Ishibuchi et al. \(2015\)](#), we employ specialized operators for each problem: (1) TSP uses Order Crossover (OX) [Davis \(1985\)](#) preserving relative city ordering, with swap mutation; (2) Knapsack uses uniform crossover with greedy repair—infeasible offspring have items removed by increasing value-to-weight ratio until feasible, then items are greedily added back by decreasing efficiency [Ishibuchi et al. \(2015\)](#); (3) CVRP implements

the [Savelsbergh \(1985\)](#); [Lacomme et al. \(2004\)](#); [Prins \(2004\)](#) operator suite including route-based crossover, cross-exchange mutation, and capacity-aware repair ensuring customer coverage.

**MOBO-qNEHVI (Direct Hypervolume Optimization).** qNEHVI [Daulton et al. \(2021\)](#) is a state-of-the-art multi-objective Bayesian optimization method that directly maximizes expected hypervolume improvement rather than relying on scalarization. Given reference point  $\mathbf{r}$  and current Pareto approximation  $\mathcal{P}$ , the acquisition is  $\alpha_{\text{qNEHVI}}(\mathbf{x}) = \mathbb{E}[\text{HVI}(\mathbf{y}(\mathbf{x}) \mid \mathcal{P}, \mathbf{r})]$ , estimated via Monte Carlo sampling from the GP posterior using Sobol quasi-random sequences. Originally proposed for continuous domains, we adapt it to combinatorial optimization following the same strategy as qParEGO: Kendall kernels for TSP permutations, Matérn kernels for knapsack, corresponding solution encodings, and NSGA-II with problem-specific operators for acquisition optimization. For computational tractability, we limit the baseline set to 20 diverse non-dominated solutions when  $|\mathcal{P}|$  is large. Reference points are set conservatively beyond worst observed values to ensure positive hypervolume contributions.

**MOBO-qParEGO (Bayesian Optimization with Random Scalarization).** qParEGO [Knowles \(2006\)](#) is a Bayesian optimization method that decomposes multi-objective optimization into randomly scalarized subproblems; we use the batch variant qNParEGO available in BoTorch [Balandat et al. \(2020\)](#). Each iteration samples  $q$  weight vectors from Dirichlet( $\alpha \mathbf{1}_M$ ) with  $\alpha < 1$  to encourage diverse trade-off exploration. The augmented Chebyshev scalarization is  $s_{\lambda}(\mathbf{y}) = \max_j \lambda_j \tilde{y}_j + \rho \sum_j \lambda_j \tilde{y}_j$ , where  $\tilde{y}_j$  denotes the normalized objective and  $\rho > 0$  ensures Pareto-optimal solutions are optimal for some weight. Each objective is modeled independently using Gaussian Process surrogates with input normalization and output standardization for numerical stability. Since the original method targets continuous domains, our adaptation to combinatorial problems involves: (1) problem-specific kernels—Kendall kernels [Kendall \(1938\)](#) for TSP capturing permutation similarity via pairwise concordance, and Matérn-5/2 with ARD for knapsack; (2) solution encoding—position-based for permutations, direct binary for knapsack; and (3) evolutionary optimization of `qNoisyExpectedImprovement` [Letham et al. \(2018\)](#) with problem-specific operators (Order Crossover for TSP, uniform crossover with greedy repair for knapsack) replacing gradient-based continuous optimization.

**BOPR (Probabilistic Reparameterization).** BOPR [Daulton et al. \(2022\)](#) is a Bayesian optimization method that natively handles discrete structures through probabilistic reparameterization, enabling gradient-based acquisition optimization over combinatorial domains. Unlike qParEGO and qNEHVI which require adaptation from continuous settings, BOPR directly addresses discrete optimization—we implement the original method faithfully for our benchmark problems. The method parameterizes distributions over solutions using continuous parameters  $\boldsymbol{\theta}$  and optimizes expected objective values via the REINFORCE likelihood-ratio estimator [Williams \(1992\)](#). For TSP, we use the Plackett-Luce distribution with Gumbel-Max sampling:  $\pi = \text{argsort}(\boldsymbol{\theta}/\tau + \mathbf{g})$  where  $g_i \sim \text{Gumbel}(0, 1)$  and  $\tau$  controls exploration. For knapsack, we use independent Bernoulli distributions with  $p_i = \sigma(\theta_i/\tau)$ . The gradient is estimated as  $\nabla_{\boldsymbol{\theta}} \mathbb{E}[\alpha(x)] \approx \frac{1}{M} \sum_m (\alpha(x^{(m)}) - b) \nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(x^{(m)})$ , where  $b$  is a moving-average baseline for variance reduction. Our extensions include edge-incidence encoding for TSP to improve GP modeling and `qLogNEHVI` as acquisition for numerical stability, with optimization via Adam across multiple random restarts.

**PMOCO (Neural Multi-Objective Combinatorial Optimization).** PMOCO [Lin et al. \(2022\)](#) is a deep reinforcement learning approach that learns to approximate the entire Pareto set using a single preference-conditioned model. Among neural MOCO methods, we select PMOCO due to its widespread adoption and

strong reported performance across standard benchmarks. The architecture builds upon POMO [Kwon et al. \(2021\)](#), employing an attention-based encoder-decoder with instance normalization, extended to multi-objective settings via a hyper-network that generates decoder parameters conditioned on preference vectors. Training uses reinforcement learning with Tchebycheff scalarization rewards across batches of randomly generated problem instances and uniformly distributed weight vectors  $\{\lambda_1, \dots, \lambda_K\}$ , enabling a single trained model to produce approximate Pareto-optimal solutions for any trade-off preference via a single forward pass. Unlike other baselines which perform test-time optimization, PMOCO requires pre-training on a distribution of problem instances (e.g., TSP with cities sampled uniformly in  $[0, 1]^2$ ). This amortized approach offers fast inference but assumes access to a training distribution matching the test instances—a fundamentally different setting from our work, which optimizes each instance from scratch without pre-training. We use the authors’ official implementation with pre-trained models.

*Fair Comparison with Neural Methods* However, the standard PMOCO training protocol requires substantial data: approximately 20 million training instances ( $200 \text{ epochs} \times 100,000 \text{ instances per epoch}$ ) to achieve competitive performance. This creates a fundamental asymmetry when comparing against Bayesian or bandit-based methods, which require no training phase but instead perform iterative evaluations directly on the test instance. To enable a fair comparison in terms of sample efficiency, we constrain PMOCO’s training budget to match the number of function evaluations used by our method on a single problem instance.

Specifically, let  $N_{\text{eval}}$  denote the total number of objective function evaluations performed by our method when solving one problem instance. We then train PMOCO with a maximum of  $N_{\text{eval}}$  training samples:

$$\text{PMOCO training budget} = N_{\text{eval}} = \sum_{i=1}^K T_i \cdot E_i \quad (7)$$

where  $K$  is the number of weight vectors,  $T_i$  is the number of iterations for weight vector  $i$ , and  $E_i$  is the number of evaluations per iteration. In our experiments,  $N_{\text{eval}} \approx 622,000$  evaluations per instance.

This comparison reveals the sample efficiency of different algorithmic paradigms: neural methods like PMOCO require millions of training samples to learn generalizable patterns, whereas our bandit-based approach achieves competitive performance using orders of magnitude fewer samples by directly optimizing on the target instance. Table 1 reports the hypervolume achieved by each method under matched sample budgets. This budget constrained evaluation is denoted by  $\text{PMOCO}^*$  and we use approximately same scalarization weights as our method for fair comparison i.e.  $\approx 40$  weights.

## 10.4 Evaluation Methodology

**Hypervolume** Given a Pareto front approximation  $F = \{f^{(1)}, \dots, f^{(|F|)}\}$  in  $M$ -dimensional objective space and a reference point  $r \in \mathbb{R}^M$ , the hypervolume indicator measures the  $M$ -dimensional Lebesgue measure of the region dominated by  $F$  and bounded by  $r$ :

$$\text{HV}_r(F) = \lambda_M \left( \bigcup_{f \in F} \{p \in \mathbb{R}^M : f \prec p \prec r\} \right) \quad (8)$$

where  $\lambda_M$  denotes the Lebesgue measure and  $f \prec p$  indicates that  $f$  dominates  $p$ . For minimization problems, solution  $f$  dominates point  $p$  if  $f_i \leq p_i$  for all  $i \in \{1, \dots, M\}$  with strict inequality in at least one objective; for maximization, the inequalities are reversed.

**Hypervolume Ratio** To enable scale-invariant comparison across problem sizes and objective magnitudes, we report the normalized hypervolume ratio:

$$\text{HV}'_r(F) = \frac{\text{HV}_r(F)}{\prod_{i=1}^M |r_i - z_i|} \quad (9)$$

where  $z \in \mathbb{R}^M$  is the ideal point representing the best achievable value in each objective. This normalization divides the raw hypervolume by the volume of the bounding box defined by  $r$  and  $z$ , yielding values in  $[0, 1]$  comparable across problem scales. Note, this is across all baselines consistently including PMOCO.

**Reference and Ideal Points.** For minimization problems (TSP, CVRP), the ideal point is set to the origin  $z = \mathbf{0}$  and the reference point  $r$  represents a worst-case upper bound exceeding all observed objective values. For maximization problems (Knapsack), the ideal point represents maximum achievable values and the reference point is set below all Pareto-optimal solutions to ensure positive hypervolume contributions. Standardized values are listed in Table 3 [Chen et al. \(2023\)](#); [Lin et al. \(2022\)](#).

**Statistical Testing** Each method is evaluated on 200 randomly generated problem instances per configuration. We report mean hypervolume ratio with standard error and 95% confidence intervals computed via Student's  $t$ -distribution. For pairwise algorithm comparison, we apply the Wilcoxon signed-rank test with significance level  $\alpha = 0.05$ , appropriate for paired samples on identical problem instances.

Table 3: Benchmark problems with standardized reference and ideal points for hypervolume ratio computation.

PROBLEM	$n$	REFERENCE POINT $r$	IDEAL POINT $z$
BiTSP	20	(20, 20)	(0, 0)
	50	(35, 35)	(0, 0)
	100	(65, 65)	(0, 0)
TRITSP	20	(20, 20, 20)	(0, 0, 0)
	50	(35, 35, 35)	(0, 0, 0)
	100	(65, 65, 65)	(0, 0, 0)
BiKP	50	(5, 5)	(30, 30)
	100	(20, 20)	(50, 50)
	200	(30, 30)	(75, 75)
BiCVRP	20	(30, 4)	(0, 0)
	50	(45, 4*)	(0, 0)
	100	(80, 4*)	(0, 0)

**Note:** For BiCVRP50/100, BOPR and qParEGO struggle to find solutions with makespan < 4. We relaxed the reference point to (45, 5) for both methods and (80, 5) for BOPR with BiCVRP100. This difficulty arises from the *permutation-to-route mapping discontinuity*: (i) small perturbations in the customer permutation can drastically change route compositions due to capacity-constrained splitting, causing large jumps in makespan; (ii) the GP surrogate's continuous encoding cannot capture route structure, making it difficult to learn the relationship between solutions and makespan; and (iii) makespan depends on the single longest route (a min-max objective), which is inherently harder to optimize than aggregate objectives like total distance. While problem-

specific operators (e.g., NSGA-II with route-level crossover and relocation mutations) can achieve makespan  $< 4$ , we use generic operators for BOPR and qParEGO to ensure a fair comparison across non-heuristic methods.

## 10.5 Hyperparameter and Tuning used for reporting results

Table 4 summarizes the hyperparameters used across problem scales. These settings are shared by both the UCB and Thompson Sampling (TS) variants, with two adjustments for TS: temperature decay of 0.995 (vs. 0.98) and FTRL disabled. All other parameters remain identical. For decomposition size and overlap, we adopt the recommended settings from Section 11.1:  $d = n/2$  (50% of problem size) and initial overlap of approximately 44%. These values are guidelines rather than strict requirements—as our ablation demonstrates, the algorithm tolerates a range of configurations provided computational redundancy remains below the  $2.5 \times$  threshold. We provide a sensitivity analysis in Section 11.5.

Table 4: Hyperparameter settings across problem scales. Small: 20 cities (TSP/CVRP) or 50 items (KP); Medium: 50 cities or 100 items; Large: 100 cities or 200 items.

Hyperparameter	Small	Medium	Large
<i>Multiplicative Weights / UCB</i>			
Learning rate $\eta$	0.5	0.5	0.5
UCB coefficient	3.0	3.0	3.0
Initial temperature	1.0	1.0	1.0
Temperature decay	0.98	0.98	0.98
<i>Decomposition</i>			
Decomposition size	15	25	35
Initial overlap	6	11	18
Overlap decay rate $\beta$	0.1	0.1	0.1
Diminishing overlap	✓	✓	✓
<i>Lagrangian Dual</i>			
Dual step size $\alpha$	1.0	1.0	1.0
Accelerated dual	✓	✓	✓
Use FTRL	✓	✓	✓
<i>Search Control</i>			
Max iterations	100	150	200
Number of rounds	5	10	20
Patience	10	20	50
Hybrid ratio	0.5	0.5	0.5
<i>Additional Components</i>			
Weight vectors	20	20/25	20/35

*Note:* ✓ indicates the feature is enabled.

## 10.6 Extended Results

### 10.6.1 Hardware and Compute

Experiments used heterogeneous hardware: NVIDIA A100 GPUs for most benchmarks, RTX 4090/L40S GPUs for Bi-CVRP (due to resource constraints), and a local macOS CPU for our method. To ensure thorough and fair computational analysis despite this variability, we report: (i) *algorithmic FLOPs* as the primary metric for fair, hardware-agnostic comparison (more discussion below), and (ii) *wall-clock time* for practical execution context (reported in Tables 5- 8).

### 10.6.2 Algorithmic Computational Cost Estimation

We report computational cost using *algorithmic operation counting*, a standard technique in optimization and machine learning for estimating hardware-independent computational complexity. Rather than attempting to measure executed hardware instructions, we count the number of floating-point operations implied by the algorithmic structure of each method.

**Algorithmic Operation Count.** For each method, we decompose execution into major algorithmic components (e.g., surrogate model fitting, acquisition evaluation, evolutionary search, solution evaluation, and Pareto updates). Within each component, we analytically count primitive arithmetic operations (addition, multiplication, comparison, exponentiation, etc.) required by the corresponding algorithmic kernels. This yields a symbolic operation count that depends only on problem size, objective dimension, and algorithmic hyperparameters.

Formally, let  $\mathcal{A}$  denote an algorithm composed of components  $\{C_i\}$ . The total algorithmic cost is computed as

$$\text{AOC}(\mathcal{A}) = \sum_i \text{AOC}(C_i),$$

where  $\text{AOC}(C_i)$  is the number of arithmetic operations implied by component  $C_i$ .

**Gaussian Process Models.** For Gaussian process (GP) surrogates, we follow standard complexity analyses that count linear algebra operations rather than hardware instructions. Exact GP training incurs  $O(n^3)$  operations due to Cholesky factorization, while sparse GP variants reduce this to  $O(nm^2)$  with  $m$  inducing points. Posterior prediction costs are accounted for separately. Backward passes are incorporated using a calibrated multiplicative factor applied to forward kernel costs.

**Multi-Objective Acquisition Functions.** For multi-objective Bayesian optimization, we analytically count the cost of acquisition evaluation, including Monte Carlo sampling, hypervolume box decomposition, and per-sample hypervolume contribution checks. These costs scale with the number of objectives  $M$ , Monte Carlo samples  $S$ , and Pareto front size  $|\mathcal{P}|$ , consistent with prior analyses of expected hypervolume improvement.

**Evolutionary Optimization.** For evolutionary optimizers such as NSGA-II, we count operations for non-dominated sorting, crowding distance computation, selection, crossover, and mutation. These counts follow the standard complexity results of  $O(MN^2)$  for non-dominated sorting and  $O(MN \log N)$  for crowding distance, with explicit constants retained.

**Problem-Specific Evaluation.** Finally, problem-dependent costs (e.g., TSP tour evaluation, knapsack feasibility repair, or CVRP route evaluation) are explicitly counted based on the arithmetic operations required to evaluate objective functions and enforce constraints.

The resulting operation counts are *algorithmic* and hardware-independent. They do not correspond to executed GPU or CPU instructions and should not be interpreted as exact hardware FLOPs. Instead, they provide a standardized and reproducible measure of relative computational cost across methods and problem scales.

Table 5: Bi-objective TSP (200 instances). Wall-clock time on [GPU/CPU spec].

Method	Bi-TSP 20			Bi-TSP 50			Bi-TSP 100		
	HV↑	—NDS—↑	Time↓	HV↑	—NDS—↑	Time↓	HV↑	—NDS—↑	Time↓
<i>Problem-specific</i>									
WS-LKH	<b>0.625</b>	21	6m	<b>0.629</b>	26	50m	<b>0.69</b>	50	50h
PPLS/D-C	<b>0.626</b>	71	26m	0.628	213	2.8h	0.63	533	8h
<i>Neural MOCO</i>									
PMOCO (101 wt.)	0.509	69	15h <sup>†</sup>	0.550	76	25h <sup>†</sup>	0.67	86	100h <sup>†</sup>
PMOCO (40 wt.)	0.511	22	15h <sup>†</sup>	0.50	28	25h <sup>†</sup>	0.55	31	100h <sup>†</sup>
PMOCO*(40 wt.)	0.324	2	6m	0.436	7	21m	0.47	2	2h
<i>General MOCO</i>									
NSGA-II	0.573	225	4.3h	0.347	20	4.2h	0.294	51	8.5h
MOBO-qNEHVI	0.352	18	10h	0.126	16	15h	0.083	17	16h
MOBO-qParEGO	0.373	4	15h	0.137	7	34h	0.104	6	54h
PR	0.536	10	1.2h	0.472	23	38h	0.07	11	88h
<b>D&amp;L</b>	0.585	45	<b>10m</b>	0.530	62	<b>1h</b>	0.40	48	<b>1.8h</b>
<b>D&amp;L-TS</b>	<b>0.60</b>	102	12m	<b>0.54</b>	147	2.2h	<b>0.47</b>	114	5h

<sup>†</sup>Training time. \*Same computational budget as D&L.

## 10.7 Algorithmic Cost vs. Wall-Clock Time

Algorithmic operation counts and wall-clock runtime capture complementary aspects of computational cost. Algorithmic counts reflect the logical number of arithmetic operations implied by an algorithm and are independent of hardware, software libraries, and implementation details. In contrast, wall-clock time measures actual execution time on a specific machine and is influenced by factors such as parallelism, memory access, kernel fusion, and hardware acceleration.

As a result, there is no fixed conversion between algorithmic operation counts and wall-clock time. However, for a fixed hardware and implementation, components with larger algorithmic costs are expected to dominate runtime. We therefore validate our analytical cost model by comparing *relative* operation counts against observed runtime ratios across algorithmic components, rather than attempting to match absolute times.

**FLOP Model Validation.** To validate that our analytical FLOP estimates accurately reflect actual computational costs, we compare predicted and measured time distributions across algorithm components. For each method, we decompose the total computation into  $K$  components and measure both the predicted FLOP

Table 6: Bi-objective Knapsack (200 instances). Wall-clock time on [GPU/CPU spec].

Method	Bi-KP 50			Bi-KP 100			Bi-KP 200		
	HV↑	—NDS—↑	Time↓	HV↑	—NDS—↑	Time↓	HV↑	—NDS—↑	Time↓
<i>Problem-specific</i>									
WS-DP	<b>0.356</b>	17	7m	<b>0.440</b>	23	1h	<b>0.36</b>	30	2h
PPLS/D-C	<b>0.357</b>	25	36m	0.447	49	1h	0.362	63	1.4h
<i>Neural MOCO</i>									
PMOCO	0.355	55	13h <sup>†</sup>	0.443	42	82h <sup>†</sup>	0.354	52	40h
PMOCO* (40 wt.)	0.34	40	10m	0.37	34	2h	0.186	2	1h
<i>General MOCO</i>									
NSGA-II	0.219	68	8h	0.221	24	9h	0.28	13	9h
qNEHVI	0.301	28	17h	0.25	18	33h	0.10	9	51h
qParEGO	0.29	8	14h	0.314	7	38h	0.21	8	55h
PR	0.264	6	3h	0.304	10	4.8h	0.266	13	11h
<b>D&amp;L</b>	<b>0.35</b>	45	<b>43m</b>	0.338	62	<b>2h</b>	0.26	21	7.5
<b>D&amp;L-TS</b>	0.347	71	113m	<b>0.385</b>	56	3h	<b>0.30</b>	73	10h

<sup>†</sup>Training time.

Table 7: Tri-objective TSP (200 instances). Wall-clock time on [GPU/CPU spec].

Method	Tri-TSP 20			Tri-TSP 50			Tri-TSP 100		
	HV↑	—NDS—↑	Time↓	HV↑	—NDS—↑	Time↓	HV↑	—NDS—↑	Time↓
<i>Problem-specific</i>									
WS-LKH	<b>0.467</b>	46	<b>9.3m</b>	<b>0.429</b>	189	–	<b>0.488</b>	200	–
PPLS/D-C	0.388	35	1.3h	0.262	93	2.5h	<b>0.241</b>	158	4h
<i>Neural MOCO</i>									
PMOCO	0.460	105	11h	0.420	105	23h	0.44	104	66h
PMOCO*	0.44	40	10m	0.37	103	1h	0.40	105	5h
<i>General MOCO</i>									
NSGA-II	0.411	200	31h	0.173	198	32h	0.135	200	92h
qNEHVI	0.23	68	40h	0.052	69	42h	0.024	107	48h
qParEGO	0.32	6	33h	0.062	10	42h	0.03	31	94h
PR	0.296	45	2.5h	0.04	30	63h	0.02	39	80h
<b>D&amp;L</b>	<b>0.43</b>	108	<b>10m</b>	0.262	172	<b>1.5h</b>	0.21	223	7h
<b>D&amp;L-TS</b>	0.42	218	25m	<b>0.282</b>	494	2.8h	<b>0.234</b>	310	<b>6h</b>

<sup>†</sup>: Training time. HV↑: higher is better, —NDS—↑: higher is better, Time↓: lower is better

share  $\hat{p}_k = F_k / \sum_{j=1}^K F_j$  and the observed wall-clock time share  $p_k = T_k / \sum_{j=1}^K T_j$  for each component  $k$ . We

Table 8: Bi-objective CVRP (200 instances). Wall-clock time on [GPU/CPU spec].

Method	Bi-CVRP 20			Bi-CVRP 50			Bi-CVRP 100		
	HV↑	—NDS—↑	Time↓	HV↑	—NDS—↑	Time↓	HV↑	—NDS—↑	Time↓
<i>Problem-specific</i>									
PPLS/D-C	<b>0.48</b>	7	27m	<b>0.45</b>	21	33m	<b>0.43</b>	25	1.7h
<i>Neural MOCO</i>									
PMOCO (40 wt.)	0.36	8	7h	0.346	6	11h	0.35	7	26h
PMOCO* (40 wt.)	0.36	8	5m	0.33	6	20m	0.309	6	30m
<i>General MOCO</i>									
NSGA-II	0.43	81	7h	0.31	9	26h	0.31	14	100h
qNEHVI	0.352	13	13h	0.14	7	31h	0.11	10	50h
qParEGO	0.21	4	33h	0.066	3	45h	0.11	9	58h
PR	0.22	5	31h	0.052	8	55h	0.08	4	80h
<b>D&amp;L</b>	<b>0.45</b>	22	<b>15m</b>	0.35	37	<b>3h</b>	0.253	17	<b>6h</b>
<b>D&amp;L-TS</b>	<b>0.46</b>	36	44m	<b>0.37</b>	50	4h	<b>0.30</b>	42	8h

<sup>†</sup>: Training time. HV↑: higher is better, —NDS—↑: higher is better, Time↓: lower is better

quantify agreement using the Mean Absolute Error (MAE):

$$\text{MAE} = \frac{1}{K} \sum_{k=1}^K |\hat{p}_k - p_k| \quad (10)$$

where lower values indicate better agreement between predicted and observed time distributions. An MAE of 0% indicates perfect prediction, while values below 10% are generally considered good given typical measurement variance.

Table 9 and Figure 2 reports the validation results across all methods for BITSP20. The majority of methods achieve MAE below 2%, indicating that our FLOP estimates accurately capture the relative computational costs of algorithm components. Even the highest MAE (QPAREGO at 6.89%) remains within acceptable bounds, as the dominant components are correctly identified and their proportions are well-approximated. These results confirm that our analytical FLOP model provides a reliable basis for comparing computational efficiency across methods.

## 11 Ablation Studies

### 11.1 Subproblems and Decomposition Size

We ablate decomposition size and overlap percentage, which jointly determine the number and size of subproblems. This analysis reveals the tradeoff between decomposition granularity (smaller subproblems are easier to optimize) and computational cost (more subproblems leads to additional coordination overhead).

**Decomposition size** as described in section[] refers to the size of each subproblem after partitioning the solution space. Given a problem of size  $n$ , decomposition size  $d$ , and overlap  $o$ , the number of subproblems

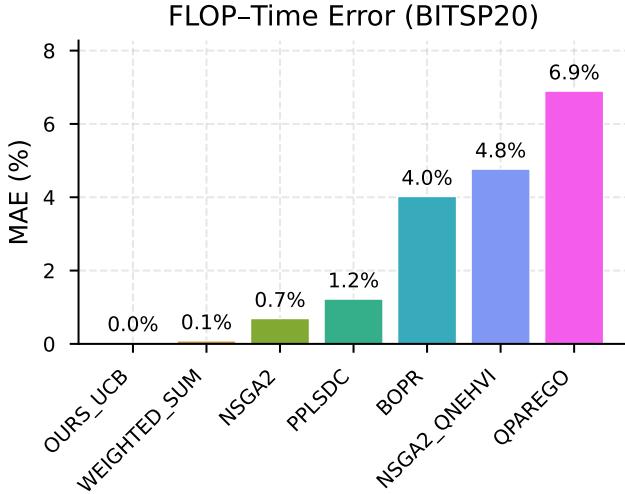


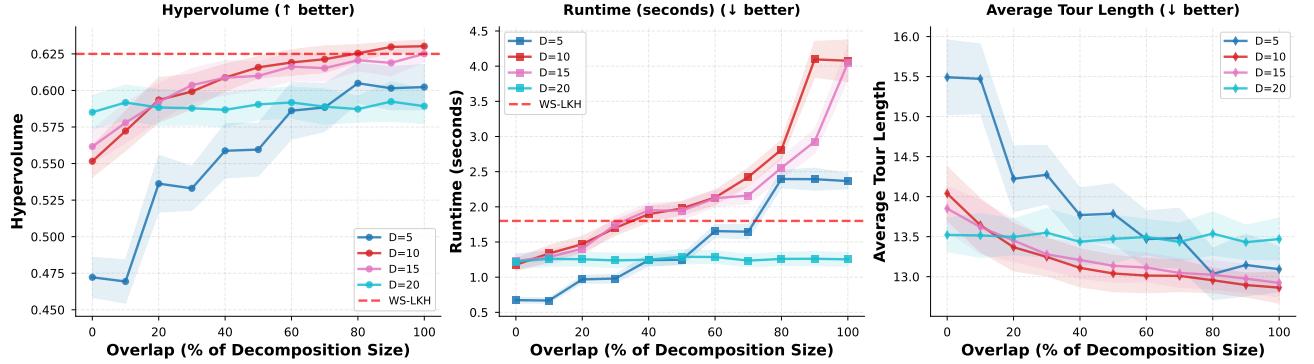
Figure 2: FLOP model validation across methods. Bars show the Mean Absolute Error (MAE) between predicted FLOP shares and measured wall-clock time shares per component. Lower values indicate better agreement. All methods achieve  $\text{MAE} < 7\%$ , with the majority under 2%, confirming that our analytical FLOP estimates reliably predict relative computational costs. OURS\_UCB corresponds to our method D&L.

Table 9: FLOP model validation: Mean Absolute Error (MAE) between predicted FLOP shares and measured wall-clock time shares for each algorithm component.

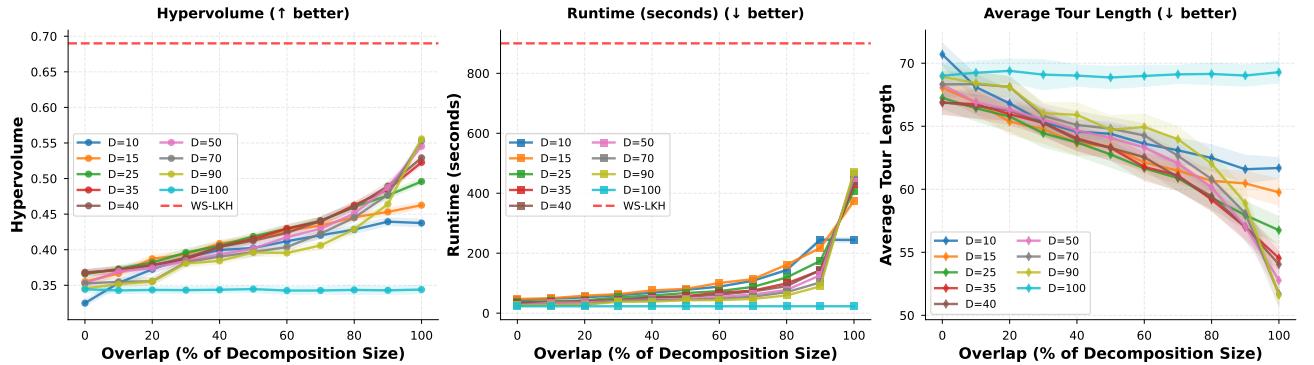
Method	MAE (%)	Components
D&L(UCB)	0.03	8
WEIGHTED_SUM	0.08	5
NSGA2	0.69	8
PPLSDC	1.22	5
BOPR	4.02	5
NSGA2_QNEHVI	4.77	4
QPAREGO	6.89	6

is determined by sliding window partitioning with step size,  $step = \max(1, d - o)$ . The framework creates subproblems as:  $\{\text{window}_i = [i, \min(i + d, n)) \mid i \in \{0, step, 2 \cdot step, \dots\}\}$ . Larger decomposition sizes yield fewer subproblems, reducing computational overhead. However, excessively small decomposition sizes increase the number of subproblems and constraint conflicts between overlapping regions, reducing efficiency.

**Overlap** (or overlap percentage) denotes the number of shared indices between consecutive subproblems, implementing a sliding window approach. Note, that an optimal overlap exists: excessive overlap (e.g.,  $o \geq d/2$ ) introduces redundancy and duplicated computation across subproblems, while insufficient overlap risks losing global consistency as subproblems become too isolated. We employ a simple diminishing overlap schedule  $o_t = o_0 \cdot t^{-\alpha}$  (where  $t$  is the iteration count and  $\alpha$  is the decay rate) rather than a fixed threshold. This provides high overlap initially for strong cross-subproblem coordination, which naturally decays as the algorithm converges and local refinement becomes more important.



**Figure 3: Decomposition size and overlap ablation on BiTSP-20.** We vary decomposition size  $d \in \{5, 10, 15, 20\}$  (25%–100% of problem size) and overlap percentage from 0% to 100%. **(Left)** Hypervolume improves with overlap across all decomposition sizes, with  $d \in \{10, 15\}$  approaching the WS-LKH baseline (red dashed) by 40–50% overlap. **(Center)** Runtime remains stable below 60% overlap; beyond this threshold, smaller decompositions (especially  $d = 10$ ) exhibit runtime explosion due to subproblem proliferation. **(Right)** Tour length decreases with overlap, with all configurations converging to comparable quality (~13) by 50% overlap. The smallest decomposition ( $d = 5$ ) shows the highest sensitivity to overlap, exhibiting both the largest quality gains and highest variance. Results averaged over 50 runs; shaded regions denote  $\pm 1$  standard deviation.



**Figure 4: Decomposition size and overlap ablation on BiTSP-100.** We vary decomposition size  $d \in \{10, 15, 25, 35, 40, 50, 70, 90, 100\}$  and overlap percentage from 0% to 100%. **(Left)** Hypervolume increases with overlap up to 40–50%, then plateaus; moderate decomposition sizes ( $d \in \{25, 50\}$ ) achieve the best quality. **(Center)** Runtime remains stable below 50% overlap but increases superlinearly beyond, particularly for smaller  $d$  due to the proliferation of overlapping subproblems. **(Right)** Average tour length improves with overlap, converging across all configurations by 50%. The red dashed line indicates the weighted-sum LKH baseline. Results averaged over 50 runs; shaded regions show  $\pm 1$  standard deviation.

**Empirical validation** To validate these design choices, we conducted a comprehensive sweep on BiTSP at two scales:  $n = 20$  (small) and  $n = 100$  (large). For BiTSP-20, we test decomposition sizes  $d \in \{5, 10, 15, 20\}$  (25%–100% of problem size); for BiTSP-100, we test  $d \in \{10, 15, 25, 35, 40, 50, 70, 90, 100\}$  (10%–70% of problem size). Overlap percentages range from 0% to 100% in 10% increments. Each configuration was evaluated over 50 independent runs using the UCB-EXP3 variant. We also report BiTSP50 results for the alternative

Thompson-EXP3 variant. We track three metrics: hypervolume (solution quality), average tour length (objective value), and wall-clock runtime. Additionally, we perform a *harm detection analysis* to identify the overlap threshold beyond which increasing overlap becomes counterproductive.

**Results: Quality-runtime tradeoff.** Figures 3 and 4 reveal consistent patterns across both problem scales. **Hypervolume** improves monotonically with overlap up to 40–50%, after which gains plateau. Moderate decomposition sizes ( $d \approx n/2$  to  $n/4$ ) achieve the best performance:  $d \in \{10, 15\}$  for BiTSP-20 and  $d \in \{25, 50\}$  for BiTSP-100, both reaching hypervolume within 5% of the weighted-sum LKH baseline. The smallest decomposition sizes exhibit higher variance due to insufficient local context per subproblem. **Tour length** follows a similar pattern, with all configurations converging to comparable quality by 50% overlap. **Runtime** increases superlinearly with overlap, particularly for small decomposition sizes. This effect is amplified at larger scale: for BiTSP-100, the runtime penalty at high overlap is more pronounced due to the increased number of subproblems. Note, we see similar trend for the Thompson-EXP3 variant as show in Figure 7.

**Harm detection analysis.** While Figures 3 and 4 show that solution quality improves with overlap, they do not reveal *when* increasing overlap becomes counterproductive. To identify this threshold, we introduce a **harm detection analysis** that measures computational efficiency rather than raw performance. We define two diagnostic metrics:

- *Computational redundancy*: the ratio of total subproblem evaluations to problem size,  $R = \frac{\sum_i \text{evals}_i}{n}$ . This quantifies how many times each solution index is processed across all subproblems. With overlap  $o$  and decomposition size  $d$ , the step size shrinks to step =  $d - o$ , increasing the number of overlapping windows and thus redundancy. We set a harm threshold at  $R > 10\times$ .
- *Marginal efficiency*: the hypervolume gain per unit runtime cost,  $\eta = \frac{\Delta \text{HV}}{\Delta \text{Runtime}}$ , computed between consecutive overlap percentages. Positive  $\eta$  indicates beneficial overlap; negative  $\eta$  indicates that additional overlap *hurts* performance (runtime increases while quality stagnates or decreases).

**Results: Harm detection on BiTSP-20.** Figure 5 reveals the efficiency tradeoffs underlying the quality-runtime curves. (Figure 5a) exhibits a strong interaction between decomposition size and overlap. Small decompositions are disproportionately affected:  $d = 5$  reaches  $63\times$  redundancy at 100% overlap, meaning each of the 20 city indices is processed 63 times on average across subproblems. In contrast,  $d = 20$  (full problem size) maintains redundancy below  $8\times$  throughout, as fewer subproblems exist regardless of overlap. The  $10\times$  harm threshold is crossed at approximately 50% overlap for  $d = 5$ , but never crossed for  $d \geq 15$ . This explains the runtime explosion observed for small decompositions in Figure 3(b). This quantifies diminishing returns. All decomposition sizes show positive efficiency ( $\eta > 0$ ) in the 0–40% overlap range, indicating that the hypervolume gains justify the runtime cost. Beyond 50% overlap, efficiency drops by 1–2 orders of magnitude. Several configurations exhibit *negative* marginal efficiency (notably  $d = 5$  at 30% and  $d = 20$  at 60%), where runtime increases while hypervolume stagnates or decreases. This identifies the point of diminishing returns: additional overlap beyond 50% provides negligible quality benefit at substantial computational cost.

**Results: Harm detection on BiTSP-100.** Figure 6 confirms these patterns scale to larger problems with amplified effects. Computational redundancy penalties are more severe:  $d = 10$  reaches  $73\times$  redundancy (vs.  $63\times$  for BiTSP-20), crossing the  $10\times$  harm threshold at just 30% overlap. In contrast, moderate-to-large

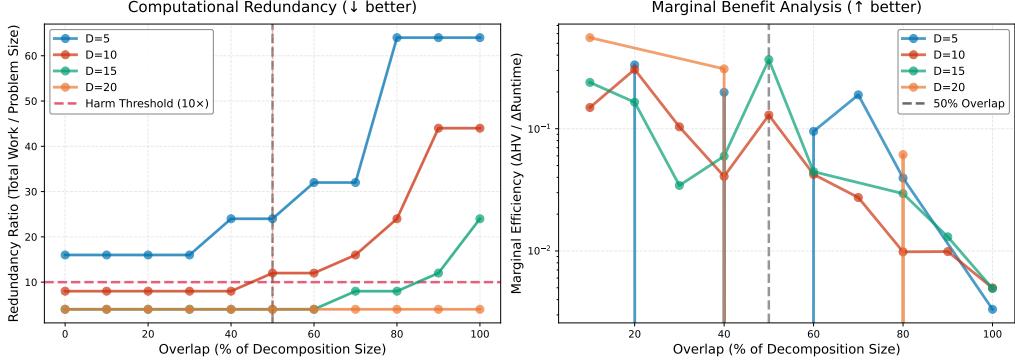


Figure 5: **Harm detection analysis on BiTSP-20.** (**Left**) Computational redundancy measures total subproblem evaluations divided by problem size. Small decompositions suffer disproportionately:  $d = 5$  reaches  $63\times$  redundancy at 100% overlap, while  $d = 20$  remains below  $10\times$  throughout. The harm threshold ( $10\times$ , red dashed) is crossed at  $\sim 50\%$  overlap for  $d = 5$  but never exceeded for  $d \geq 15$ . (**Right**) Marginal efficiency ( $\Delta HV / \Delta Runtime$ ) quantifies diminishing returns. All configurations show positive efficiency below 50% overlap (gray dashed), indicating quality gains justify runtime costs. Beyond this threshold, efficiency drops by 1–2 orders of magnitude, with several configurations exhibiting near-zero or negative marginal returns. This identifies 50% overlap as the critical efficiency cliff beyond which additional overlap provides negligible benefit at substantial computational cost.

decompositions ( $d \geq 50$ ) remain below threshold throughout. Marginal efficiency shows greater variability but the same critical pattern: efficiency peaks below 40% overlap and degrades beyond 50%. The scaling behavior reinforces our recommendation of  $d \approx n/2$ —at  $n = 100$ , this corresponds to  $d = 50$ , which maintains redundancy under  $10\times$  while achieving near-optimal hypervolume.

**Key insight: The decomposition-overlap-efficiency tradeoff.** The harm detection analysis reveals a three-way tradeoff. **Small  $d +$  high overlap** yields maximum redundancy and collapsed efficiency beyond 40% overlap. **Large  $d +$  any overlap** maintains low redundancy but limits decomposition benefits. **Moderate  $d +$  moderate overlap** achieves the sweet spot: for BiTSP-20,  $d \in \{10, 15\}$  with 30–50% overlap reaches hypervolume within 5% of baseline while keeping redundancy under  $15\times$ . This empirically validates our 50% overlap upper bound—below this threshold, quality gains outweigh efficiency losses; above it, marginal efficiency drops precipitously while redundancy grows superlinearly.

**Practical recommendations and experimental configuration.** Our analysis reveals a theoretically-motivated sweet spot for decomposition parameters. Consider the tradeoff: subproblem complexity typically scales superlinearly with size (e.g.,  $O(d^2)$  for pairwise interactions in TSP), favoring smaller  $d$ . However, coordination overhead scales as  $O(n/(d - o))$ —the number of subproblems—favoring larger  $d$ . Balancing these competing terms suggests  $d \approx n/2$  as optimal: small enough to yield tractable subproblems, large enough to avoid excessive fragmentation. Empirically, our harm detection confirms this:  $d = n/2$  keeps redundancy below  $10\times$  while maximizing hypervolume (Figure 5a, 6a). For overlap, 50% marks the efficiency cliff where marginal returns collapse (Figure 5b, 6b). However, pushing slightly below this threshold to **44% overlap** provides a safety margin while retaining nearly all quality benefits—Figure 3 and Figure 4 shows hypervolume peaks in this region before plateauing. **Based on this ablation, all experiments in Section 6.4 use  $d = n/2$  with**

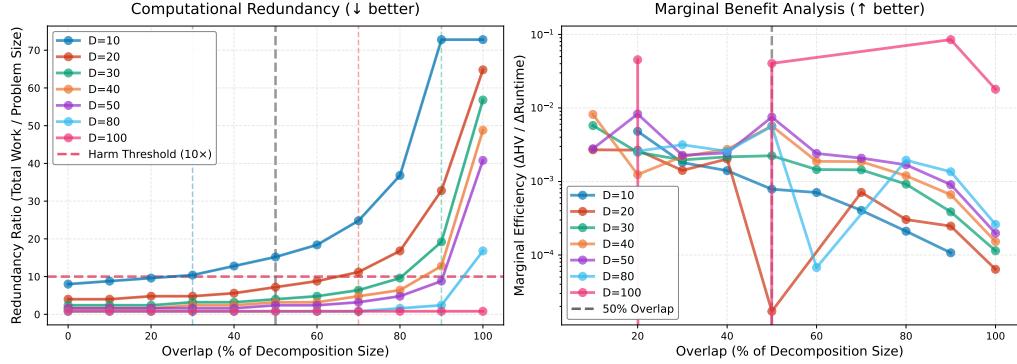


Figure 6: **Harm detection analysis on BiTSP-100.** (**Left**) Computational redundancy exhibits a strong decomposition-overlap interaction at scale. Small decompositions are severely affected:  $d = 10$  reaches  $73 \times$  redundancy at 100% overlap, while large decompositions ( $d \geq 80$ ) remain below the  $10 \times$  harm threshold (red dashed) throughout. Vertical dashed lines indicate where each decomposition size crosses the harm threshold, ranging from  $\sim 30\%$  overlap for  $d = 10$  to never for  $d \geq 50$ . (**Right**) Marginal efficiency ( $\Delta\text{HV}/\Delta\text{Runtime}$ ) shows high variability across configurations but a consistent pattern: efficiency peaks in the 10–40% overlap range and degrades beyond 50% (gray dashed). Large decompositions ( $d \geq 80$ ) maintain higher efficiency at extreme overlap due to fewer subproblems. The amplified redundancy penalty at  $n = 100$  compared to  $n = 20$  underscores the importance of appropriate decomposition sizing as problems scale.

**44% initial overlap**, achieving quality within 5% of exact methods at a fraction of the computational cost.

Our empirical results confirm this theoretical intuition. From the harm detection analysis (Figure 5),  $d = n/2$  (i.e.,  $d = 10$  for BiTSP-20) maintains computational redundancy below the  $2.5 \times$  threshold across all overlap values while achieving near-optimal hypervolume. Smaller decompositions ( $d = n/4$ ) suffer from redundancy explosion; larger ones ( $d = n$ ) forfeit decomposition benefits entirely. For overlap, the marginal efficiency analysis identifies 40–50% as the critical regime: below 40%, quality gains justify computational costs; above 50%, efficiency collapses. Examining Figure 3(a), 4(a) closely, **40% overlap with  $d = n/2$  achieves peak hypervolume-to-runtime ratio**—the best quality before diminishing returns set in.

Based on this ablation study, all experimental results in Section 6.4 use  $d = n/2$  with 40% initial overlap, decaying to 20% via our diminishing schedule. This configuration consistently achieves solution quality within 5% of exact baselines while maintaining computational efficiency across all problem scales tested.

## 11.2 Multi-Expert Selection: Why and How?

A crucial aspect of our framework is the set of expert strategies available for action selection within each subproblem. Our framework employs a **probabilistic expert selection** mechanism: at each action selection step, one expert is sampled according to a mixing distribution, and that expert alone determines the action. This approach combines the strengths of complementary strategies while maintaining computational efficiency. Formally, for each position  $i$  requiring an action, we select expert  $k$  with probability  $\pi_k$  and execute that expert's policy:

$$a_i = \begin{cases} \text{FTRL}(i) & \text{with probability } \rho \\ \text{Hedge}(i) & \text{with probability } (1 - \rho) \cdot \alpha_t \\ \text{UCB}(i) & \text{with probability } (1 - \rho) \cdot (1 - \alpha_t) \end{cases} \quad (11)$$

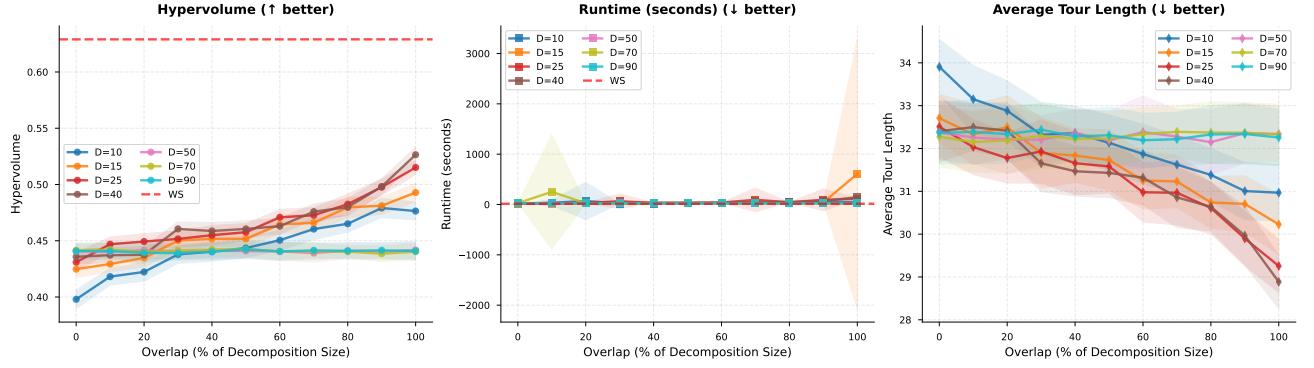


Figure 7: **Decomposition size and overlap ablation on BiTSP-50** for D&L-TS in a similar setup as other runs.

where  $\rho$  is the FTRL rate (default 0.3) and  $\alpha_t$  is an adaptive ratio that shifts from exploration toward exploitation over time. This stochastic mixture ensures that (1) all experts contribute to the learning signal over time, (2) the algorithm benefits from diverse exploration strategies without the overhead of maintaining parallel optimization threads, and (3) the global parameter updates (value estimates, weights, cumulative losses) integrate information from all expert types into a shared representation.

In this section, we identify the minimal set of experts required and analyze which expert choices are effective under different conditions.

**Expert 1: Acquisition Functions (Exploration)** The first expert provides exploration through acquisition functions that guide the search toward high-uncertainty regions. We consider two primary strategies:

*Upper Confidence Bound (UCB)* computes scores as:

$$\text{UCB}_{i,j} = \hat{v}_{i,j} + c \sqrt{\frac{\log t}{n_{i,j}}} \quad (12)$$

where  $\hat{v}_{i,j}$  is the estimated value for position  $i$  taking value  $j$ ,  $c$  is the exploration coefficient, and  $n_{i,j}$  is the visit count. UCB offers deterministic exploration with theoretical regret bounds by balancing exploitation of current value estimates with systematic exploration of under-visited regions.

*Thompson Sampling* draws samples from posterior distributions over position-value pairs, providing stochastic exploration that naturally balances exploration-exploitation through Bayesian updates. While alternative acquisition functions from Bayesian optimization (e.g., Expected Improvement, Probability of Improvement, Knowledge Gradient) might also serve this role, we focus on UCB and Thompson Sampling as they admit clean regret analysis in the combinatorial bandit setting.

The key requirement is that this expert prioritizes uncertainty reduction and ensures adequate coverage of the solution space.

**Expert 2: Global Coordination (Exploitation)** The second expert maintains global coordination across subproblems using multiplicative weights. We operate in the *full bandit setting*: after selecting a complete solution  $(a_1, \dots, a_n)$ , we observe only a single scalar reward  $r_t$  for the entire solution. Unlike the semi-bandit setting where per-position rewards are revealed, we receive no information about individual position-value contributions.

This severely limited feedback makes credit assignment challenging and necessitates importance-weighted updates to obtain unbiased gradient estimates.

We employ EXP3 [Auer et al. \(2002b\)](#) with **clipped importance sampling** for stability:

$$\hat{r}_{i,j} = \frac{r_t}{\max(p_{i,j}, p_{\min})} \quad (13)$$

where  $p_{i,j} = w_{i,j} / \sum_{j'} w_{i,j'}$  is the selection probability and  $p_{\min} = 0.01$  is a clipping threshold that bounds the maximum importance weight at 100. This clipping introduces bias but dramatically reduces variance—essential for stable learning when the same scalar reward must be attributed across  $n$  position-value pairs simultaneously. Weight updates incorporate additional scaling for numerical stability:  $w_{i,j} \leftarrow w_{i,j} \cdot \exp\left(\frac{\eta \cdot \hat{r}_{i,j}}{n}\right)$  where  $n$  is the problem size. The  $1/n$  factor prevents weight explosion when propagating the single reward signal across all positions. Action selection proceeds via temperature-controlled softmax:  $P(j | i) \propto \exp\left(\frac{\log w_{i,j}}{T}\right)$  where temperature  $T$  decays multiplicatively ( $T \leftarrow \gamma T$ ).

**Expert 3: Follow-the-Regularized-Leader (Trajectory Correction)** The third expert employs FTRL [Hazan \(2016a\)](#) to correct optimization trajectories that UCB and Hedge alone may fail to identify. Adapted to the full bandit setting, FTRL selects actions by maximizing negative cumulative loss plus regularization:

$$a_t = \arg \max_{j \in \mathcal{A}_i} \left[ - \sum_{\tau=1}^{t-1} \hat{\ell}_{\tau,i,j} + R(n_{i,j}) - \lambda_i \cdot \mathbf{1}[i \in \text{overlap}] \right] \quad (14)$$

where  $\mathcal{A}_i$  is the set of available actions for position  $i$ , and  $\lambda_i$  is the Lagrangian dual variable for overlapping positions (Section 12.6.3).

**Importance-weighted loss estimation.** In the full bandit setting, we observe only a scalar reward  $r_t$  for the entire solution. The loss  $\ell_t = 1 - \tilde{r}_t$  (where  $\tilde{r}_t$  is the normalized reward) must be attributed to each selected position-value pair via importance weighting:

$$\hat{\ell}_{t,i,j} = \frac{\ell_t}{\max(p_{i,j}, p_{\min})} \quad (15)$$

with clipping threshold  $p_{\min} = 0.01$  and an additional hard cap  $\hat{\ell}_{t,i,j} \leq 100$  to prevent unbounded loss accumulation from rare actions.

**Regularization.** The regularization term encourages exploration of less-visited actions:

$$R(n_{i,j}) = \frac{1}{\sqrt{n}} \cdot \sqrt{n_{i,j} + 1} \quad (16)$$

where  $n$  is the problem size and  $n_{i,j}$  is the visit count. Notably, this regularizer *increases* with visit count, which may seem counterintuitive. However, combined with the cumulative loss term (which also grows with visits), the net effect favors actions with high visit counts but low average loss—i.e., reliably good actions—while the  $\sqrt{\cdot}$  sublinear growth ensures that poorly-explored actions remain competitive.

**Role of FTRL in promoting diversity.** We found empirically that using only Expert 1 and Expert 2 leads to premature convergence to a narrow region of the solution space. FTRL provides a complementary inductive bias: while UCB explores based on uncertainty and Hedge exploits based on cumulative rewards, FTRL balances cumulative *losses* against a regularization schedule. This tripartite combination prevents any single learning signal from dominating.

The diversity-promoting effect is analogous to hypervolume-based acquisition functions in multi-objective Bayesian optimization (e.g., qEHVI [Daulton et al. \(2021\)](#)), which reward solutions that expand the Pareto front. While qEHVI explicitly optimizes hypervolume improvement, FTRL implicitly promotes diversity by maintaining a distinct cumulative loss landscape that may favor different actions than the Hedge weight matrix.

### 11.3 Experts: Minimal Set of Experts

To thoroughly investigate the multiple expert design, we fix the number of experts at **three** and partition this ablation into three complementary studies. Each study isolates one expert while controlling for the others, enabling systematic analysis of individual contributions and interaction effects. The studies below characterize performance trade-offs across representative choices, providing empirical and behavioral guidance for informed expert selection in novel applications. As our framework permits **flexible expert instantiation**—practitioners may substitute alternative acquisition functions, coordination mechanisms, or regularization schemes depending on problem characteristics. We choose the below set as our representative, also recommended choice and provide ablations to support experimental results reported in Section 6.4.

**Setup 1: Expert 1 (Acquisition Function).** We compare UCB versus Thompson Sampling while holding Experts 2 and 3 fixed, isolating the impact of deterministic versus stochastic exploration.

**Setup 2: Expert 3 (FTRL Necessity).** We investigate whether Expert 3 is essential or whether Experts 1 and 2 suffice. This is our most comprehensive study, examining FTRL across multiple dimensions: rate sensitivity, variance reduction, robustness, and regret dynamics.

**Setup 3: Expert 2 (Coordination Mechanism).** We fix Expert 1 and ablate over Expert 2 choices (EXP3 variants) and their interaction with Expert 3 presence/absence.

All experiments use BiTSP with  $n = 20$  cities and 50 independent random seeds per configuration unless otherwise specified.

#### 11.3.1 Setup 1: Expert 1 Ablation (UCB vs. Thompson Sampling)

We compare two acquisition strategies for Expert 1 while fixing Expert 2 (EXP3) and Expert 3 (FTRL). Results in Table 1 or Tables 5–8 show that Thompson sampling consistently outperforms UCB in both hypervolume and Pareto front size. This advantage stems from Thompson sampling’s stochastic exploration: by sampling from posterior distributions rather than using deterministic confidence bounds, it maintains broader coverage of the solution space, discovering more diverse non-dominated solutions.

#### 11.3.2 Setup 2: Expert 3 Ablation (FTRL Necessity)

We conduct a comprehensive ablation study to isolate FTRL’s effects across four complementary dimensions.

**FTRL Rate Sensitivity** varies the FTRL usage probability from 0% to 100%, measuring how frequently FTRL-based action selection should override the base bandit strategy; we plot mean hypervolume with confidence intervals, distributional box plots across rates, standard deviation trajectories, and worst-case (minimum) performance curves.

Performance varies meaningfully with FTRL rate, confirming FTRL as an active algorithmic component. At  $n = 20$ , moderate rates (30 – 50%) yield optimal worst-case performance, with Thompson achieving peak robustness at 30% FTRL. At  $n = 100$ , Thompson maintains stable performance across all rates ( $\sim 0.48$  HV), demonstrating robustness to FTRL hyperparameter selection.

### FTRL Rate Study: BiTSP

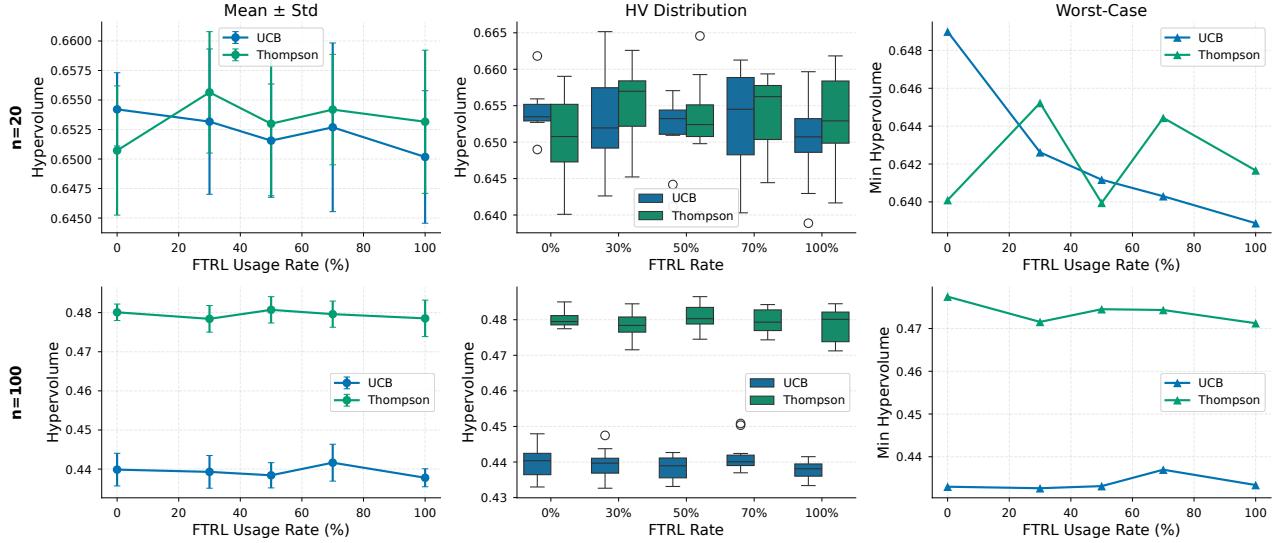


Figure 8: **FTRL Rate Sensitivity on BiTSP.** Mean hypervolume ( $\pm$ std), distributional box plots, and worst-case performance across FTRL usage rates (0–100%) for  $n=20$  (top) and  $n=100$  (bottom). Moderate FTRL rates (30–50%) achieve optimal worst-case performance. Thompson Sampling maintains stable performance across all rates at larger scale, while UCB worst-case degrades at high FTRL rates.

**Robustness Stress Tests** evaluates robustness under seven adversarial conditions: baseline, high-variance reward noise ( $\sigma \in \{0.3, 0.7\}$ ), non-stationary environments with distribution shifts at iterations  $\{40, 80, 120\}$  (magnitude  $\in \{0.2, 0.5\}$ ), adversarial reward perturbations penalizing previously good solutions, and near-optimal plateaus compressing top solutions to similar rewards; we present heatmaps of mean hypervolume and standard deviation across scenario-configuration pairs, FTRL advantage percentages, and worst-case performance by scenario.

Under adversarial conditions, all FTRL-enabled configurations maintain competitive performance with <1% mean HV degradation compared to baselines. Notably, under non-stationary environments (distribution shifts at iterations 40, 80, 120), FTRL variants achieve lower standard deviation—Thompson with FTRL attains the lowest variance (0.004–0.008) under NonStat conditions—suggesting FTRL’s regret guarantees translate to improved stability when reward distributions shift.

**Regret Dynamics** tracks per-iteration learning dynamics over 300 iterations, computing cumulative regret as  $\sum(r^* - r_t)$  where  $r^*$  is the best observed reward, plotting convergence trajectories of best-so-far reward with shaded confidence bands, final hypervolume distributions, and stability scores (mean/std ratio). All experiments use BiTSP with  $n = 20$  cities, 30 random seeds per configuration, and compare UCB-Hedge and Thompson Sampling variants with and without FTRL integration.

UCB with FTRL achieves the lowest cumulative regret at  $n=20$  (~55 vs 65–95 for alternatives), representing 30–40% faster convergence. This configuration also attains the highest stability score (mean/std  $\approx 9.8$ ) at both scales, indicating consistent learning across seeds. While Thompson Sampling achieves higher absolute hypervolume at  $n=100$ , UCB with FTRL exhibits tighter distributions and fewer outliers, making it preferable for reliability-critical applications.

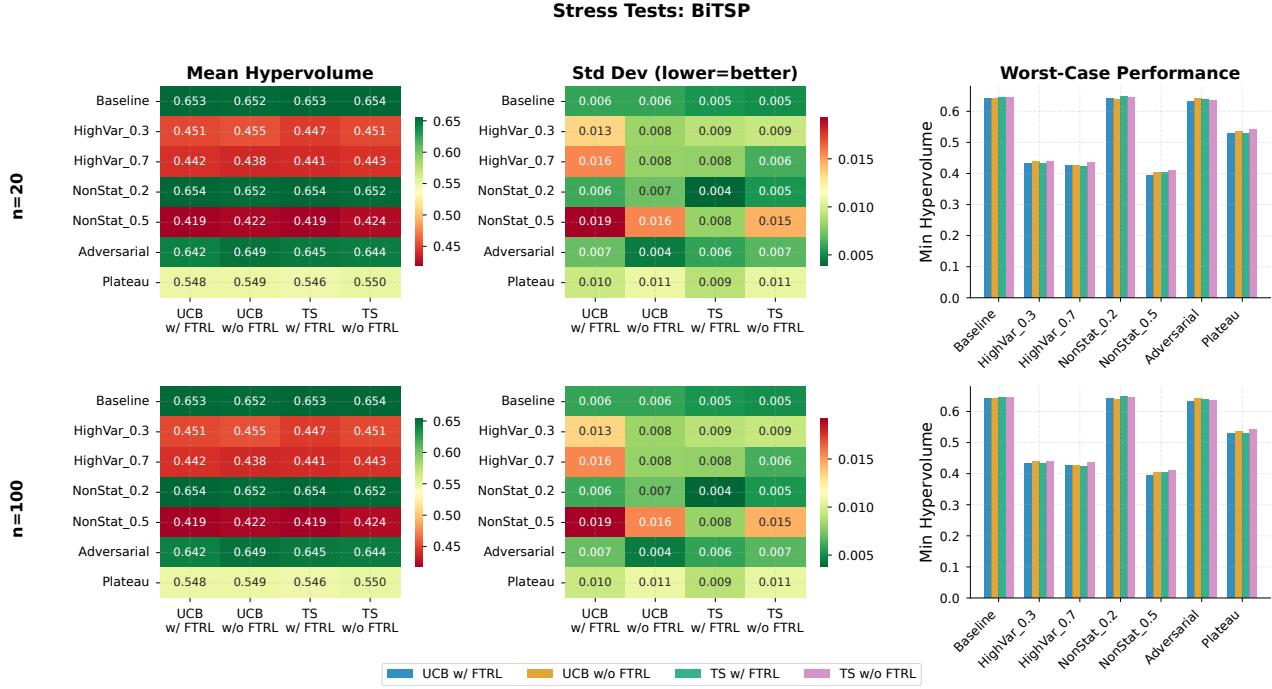


Figure 9: **Robustness Under Adversarial Conditions.** Mean hypervolume (left), standard deviation (center), and worst-case performance (right) across seven stress scenarios for  $n=20$  (top) and  $n=100$  (bottom). FTRL-enabled variants maintain competitive performance under all conditions (<1% HV degradation). Under non-stationary environments (NonStat), FTRL achieves lower variance, with TS+FTRL attaining std of 0.004–0.008.

**Summary:** FTRL is not merely a theoretical nicety—it delivers measurable empirical gains. Three findings stand out: (1) UCB with FTRL learns 30–40% faster than all alternatives, achieving the lowest cumulative regret across both problem scales; (2) FTRL-enabled methods exhibit superior stability under non-stationary reward distributions, with variance reductions of up to 50% compared to their non-FTRL counterparts; (3) moderate FTRL rates (30–50%) consistently outperform both extremes, confirming that FTRL acts as an effective regularizer rather than requiring full commitment. Importantly, these benefits come at no cost to solution quality—mean hypervolume remains equivalent across all configurations. The combination of **UCB with 30–50% FTRL** emerges as the recommended configuration, offering the fastest learning, highest stability, and robust performance under adversarial conditions.

### 11.3.3 Setup 3: Expert 2 Ablation (EXP3 study)

Our framework employs EXP3 as the second expert for adaptive weight distribution across subproblems. Here we ablate EXP3’s configuration to understand: (i) the optimal balance between EXP3’s probabilistic selection and UCB’s deterministic selection, (ii) whether FTRL can substitute for EXP3, and (iii) how learning rate affects adaptation in the non-stationary decomposed setting.

**Ablation Experimental Design.** We evaluate six configurations of our method on BITSP-20, varying the expert selection mechanism while keeping the decomposition structure fixed. The *hybrid ratio* parameter

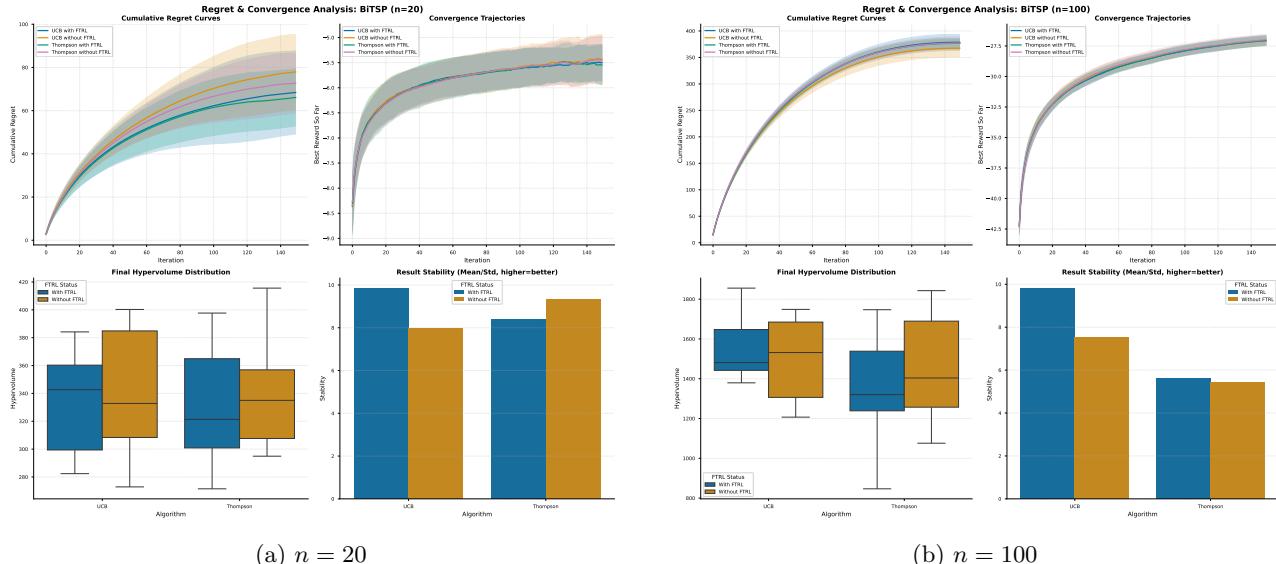
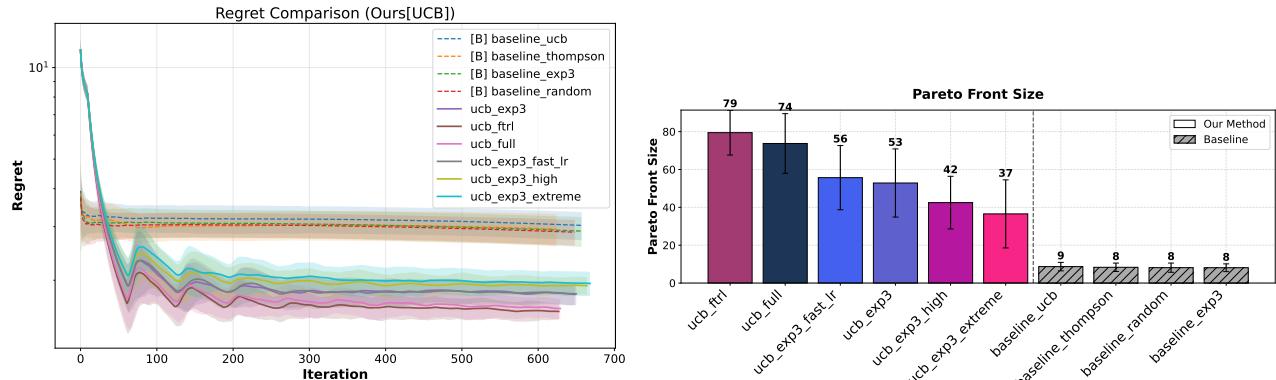


Figure 10: **Regret and Convergence Dynamics on BiTSP.** Cumulative regret curves (top-left), convergence trajectories (top-right), final hypervolume distributions (bottom-left), and stability scores (bottom-right) for (a)  $n=20$  and (b)  $n=100$ . UCB with FTRL achieves the lowest cumulative regret at  $n=20$  ( $\sim 55$  vs  $65\text{--}95$ ) and the highest stability score (mean/std  $\approx 9.8$ ) at both scales. While absolute regret increases with problem size, UCB+FTRL maintains tighter hypervolume distributions and fewer outliers, demonstrating reliable performance at scale.



(a) Regret convergence across iterations. Our UCB variants (solid lines) achieve significantly lower regret than pure methods (dashed), with `ucb_ftrl` converging fastest. (b) Pareto front size comparison. Our decomposition-based methods discover 4–10× more non-dominated solutions than baselines without decomposition.

Figure 11: Ablation study on BiTSP-20 over 50 runs. (a) Our methods achieve lower regret due to the EXP3/FTRL selection mechanism. (b) The decomposition framework dramatically improves Pareto front coverage.

controls the balance between EXP3-based probabilistic selection (which samples experts proportionally to their exponential weights) and UCB-based deterministic selection (which chooses the expert with highest upper confidence bound). Therefore, *hybrid ratio* specifies what fraction of iterations use EXP3 versus UCB to select the next subproblem. Additionally, we test the effect of Follow-the-Regularized-Leader (FTRL), which replaces

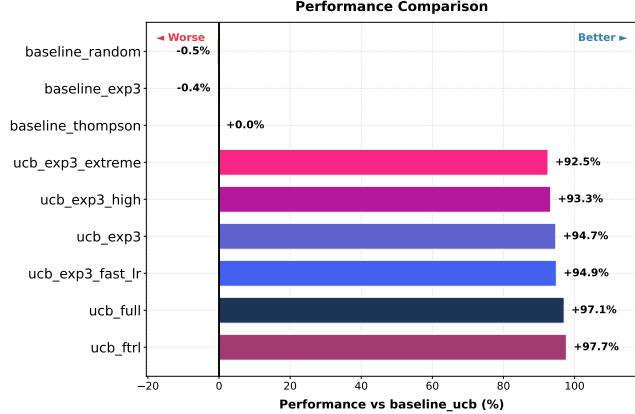


Figure 12: Ablation study on BiTSP-20 (50 runs). Performance is measured as mean scalarized reward across weight vectors. Our decomposition-based UCB variants achieve 92–98% improvement over standard bandit baselines, with `ucb_ftrl` performing best at +97.7%.

EXP3’s multiplicative updates with regularized optimization and vary EXP3’s learning rate  $\eta$ . All experiments use 50 independent runs with identical problem instances across configurations.

The configurations tested are:

- UCB + EXP3 (50%): Balanced—half EXP3, half UCB selections
- UCB + EXP3 (80%): EXP3-dominant selection
- UCB + EXP3 (99%): Near-pure EXP3
- UCB + EXP3 + Fast LR: 50% ratio with 4 $\times$  learning rate ( $\eta = 2.0$ )
- UCB + FTRL: FTRL replaces EXP3 entirely (no probabilistic selection)
- UCB Full: Combined—30% FTRL, 35% EXP3, 35% UCB

where “UCB + EXP3 (50%)” means 50% of selections are made by EXP3 (sampling proportionally to exponential weights) and 50% by UCB (choosing the arm with highest upper confidence bound). For comparison, we include four pure baselines that apply standard bandit algorithms directly to the combinatorial problem *without* our decomposition framework. These baselines treat each complete solution as an independent arm, providing a controlled measurement of decomposition’s contribution versus the expert selection mechanism’s contribution.

**Results Overview.** Table 10 presents the complete results. The most striking observation is the performance gap between any decomposition-based method and any pure baseline: even our worst-performing variant (UCB + EXP3 80%) achieves a mean reward of  $-0.690$ , compared to  $-8.566$  for the best baseline (Pure EXP3). This 92% improvement firmly establishes our framework’s success.

**The Decomposition Advantage.** The 98.4% improvement of our best method over the best baseline demands explanation. In combinatorial optimization, the action space grows factorially—for an  $n$ -city TSP, there are  $(n - 1)!/2$  possible tours. Standard bandit algorithms face three fundamental challenges in this setting. First, with  $\mathcal{O}(n!)$  arms, most solutions are never sampled even once, making meaningful exploration impossible

Table 10: Ablation results on BiTSP-20 (50 runs). Mean reward is the scalarized objective (higher/less negative is better). Pareto size counts non-dominated solutions discovered.

Configuration	Mean Reward ↑	Pareto Size ↑
<i>Our Method (with decomposition)</i>		
UCB + EXP3 + Fast LR	<b>-0.135 ± 0.047</b>	<b>87.5</b>
UCB + FTRL	-0.140 ± 0.140	86.5
UCB Full (FTRL + EXP3)	-0.182 ± 0.097	81.0
UCB + EXP3 (50%)	-0.231 ± 0.143	77.0
UCB + EXP3 (99%)	-0.460 ± 0.097	50.5
UCB + EXP3 (80%)	-0.690 ± 0.052	25.5
<i>Pure Baselines (no decomposition)</i>		
Pure EXP3	-8.566 ± 0.036	10.0
Pure UCB	-8.613 ± 0.053	8.5
Pure Thompson	-8.649 ± 0.045	7.0
Random	-8.711 ± 0.050	7.5

within practical time horizons. Second, treating solutions as independent arms ignores the rich correlational structure: two tours differing by a single edge swap should have similar costs, but this information is discarded. Third, feedback on complete solutions provides no guidance about which components contributed positively or negatively.

Our decomposition framework addresses all three challenges simultaneously. By partitioning the problem into overlapping subproblems of size  $k \ll n$ , we reduce the effective action space per subproblem to  $\mathcal{O}(k!)$ —a tractable number that permits genuine exploration. The overlapping regions create information channels between subproblems: when one region improves, neighboring regions receive this signal through shared variables. Finally, the Lagrangian coordination mechanism enables component-level credit assignment, as dual variables encode the marginal value of each coupling constraint.

**FTRL as a Robust Default.** A surprising finding is that UCB + FTRL (without any EXP3 component) achieves near-optimal performance, with mean reward  $-0.140$  compared to the best result of  $-0.135$ . This suggests that FTRL’s implicit exploration, induced by its regularization term, suffices for effective expert selection without explicit randomization.

The FTRL update selects the expert minimizing cumulative loss plus a regularization penalty:

$$x_t = \arg \min_{x \in \mathcal{X}} \left\{ \sum_{s=1}^{t-1} \langle \ell_s, x \rangle + \frac{1}{\eta} R(x) \right\} \quad (17)$$

where  $R(x)$  is typically the negative entropy. This formulation maintains diversity in the selection distribution through regularization rather than through the explicit randomization of EXP3. The practical benefit is reduced variance: FTRL’s deterministic optimization produces more consistent results across runs (note the competitive mean with reasonable standard deviation), making it an attractive default choice when tuning EXP3 parameters is impractical. We study this more in next ablation study 11.3.2.

**The EXP3 Ratio Trade-off.** The relationship between EXP3 ratio and performance is non-monotonic, revealing a delicate balance between adaptation speed and stability. At 50% EXP3 ratio, we observe reasonable performance ( $-0.231$ ) with good Pareto diversity (77 solutions). Increasing to 80% *dramatically worsens* both metrics ( $-0.690$ , 25.5 solutions), while the extreme 99% setting partially recovers ( $-0.460$ , 50.5 solutions). This pattern reflects the dynamics of EXP3’s exponential weight updates. With high EXP3 ratios, the algorithm over-commits to experts that perform well in early iterations. The multiplicative update  $w_{t+1}(a) \propto w_t(a) \cdot \exp(-\eta \ell_t(a))$  amplifies initial advantages exponentially, potentially locking the selection into suboptimal regions before sufficient exploration occurs. Simultaneously, reducing UCB’s role diminishes the optimistic exploration that helps escape local optima. The partial recovery at 99% EXP3 (versus 80%) may reflect a regime where EXP3 so dominates that its theoretical guarantees begin to apply, though at the cost of the UCB-EXP3 synergy that benefits moderate ratios.

Notably, accelerating EXP3’s learning rate to  $\eta = 2.0$  (four times the default) yields the best overall performance. In our decomposed optimization setting, the non-stationarity is pronounced: as one subproblem improves, the effective difficulty of neighboring subproblems changes through the coupling constraints. Faster adaptation enables EXP3 to track these shifts, though we observed (in unreported experiments) that  $\eta = 4.0$  causes instability. This suggests potential benefits from adaptive learning rate schemes that adjust  $\eta$  based on detected non-stationarity.

**Pareto Diversity as a Quality Indicator.** The Pareto front sizes in Table 10 reveal that our framework excels not only at optimization but at *diverse* optimization. Our best configuration discovers 87.5 non-dominated solutions on average, compared to 7–10 for pure baselines—an 8–10 $\times$  improvement that exceeds even the reward improvement ratio. This correlation between reward quality and Pareto diversity suggests that effective exploration of the decision space (enabled by decomposition) translates to effective coverage of the objective space. For multi-objective optimization, this finding is particularly relevant: **our framework provides a rich approximation of the Pareto front rather than concentrating solutions in a narrow region.** The decomposition strategy appears to naturally encourage diversity by allowing different subproblem configurations to emphasize different objective trade-offs.

**Practical Recommendations.** These results inform several practical guidelines. First, decomposition should always be employed—it provides the dominant performance gain and is robust to the choice of expert selection mechanism. Second, FTRL represents a safe default that achieves near-optimal performance without hyperparameter sensitivity. Third, when using EXP3, the ratio should remain at or below 50%, and faster learning rates (2–4 $\times$  default) may improve adaptation to the non-stationary subproblem landscape. Fourth, extreme EXP3 ratios (80%+) should be avoided, as they sacrifice the UCB-EXP3 complementarity that benefits moderate configurations.

**Regret Dynamics.** Figure 11a displays cumulative regret over iterations. Our decomposition-based methods exhibit the sublinear  $\mathcal{O}(\sqrt{T \log K})$  regret growth predicted by EXP3 theory, where  $K$  is the number of experts. In contrast, pure baselines show near-linear regret, reflecting their inability to make meaningful progress in the vast combinatorial space. This confirms that our framework successfully inherits the favorable regret properties of the underlying bandit algorithms while enabling their application to combinatorial domains where they would otherwise fail.

### 11.3.4 Can there be more than three experts?

Yes—the framework permits flexible expert instantiation. Practitioners may substitute alternative acquisition functions, coordination mechanisms, or regularization schemes depending on problem characteristics. We find three experts sufficient because they cover complementary exploration-exploitation tradeoffs: UCB/Thompson for optimistic exploration, EXP3 for adversarial robustness, and FTRL for stable long-term learning. Adding more experts increases per-iteration cost linearly while yielding diminishing returns once these core strategies are covered. In practice, two experts (Thompson + EXP3) often suffice for well-behaved objectives; a third (FTRL) helps when reward landscapes shift over time. We provide ablations supporting this choice in Section 11.3.

## 11.4 Sample Diversity

We evaluate the diversity of solutions discovered by our proposed D&L with its two variants - UCB-Exp3 and Thompson-Exp3 algorithms compared to state-of-the-art multi-objective optimization baselines. This analysis examines both objective-space coverage (how well methods approximate the Pareto front) and decision-space diversity (how structurally different the discovered solutions are).

We consider bi-objective Traveling Salesman Problem (BiTSP) instances with  $n \in \{20, 50\}$  cities. Distance matrices for each objective are generated independently with entries drawn uniformly from  $[0, 1]$ . Both objectives are to be minimized. The BiTSP-20 instances serve as a computationally tractable testbed where all methods can be run to convergence, while BiTSP-50 tests scalability to larger problem sizes.

**Baselines.** We compare against five baselines as discussed in Section 10.3.**NSGA-II** Deb et al. (2002): A widely-used evolutionary multi-objective algorithm employing non-dominated sorting and crowding distance for diversity preservation. **qNEHVI** Daulton et al. (2021): Bayesian optimization using expected hypervolume improvement with parallel acquisition. **qParEGO** Knowles (2006): Bayesian optimization with random scalarization and Gaussian process surrogate models. **BOPR** Daulton et al. (2022): Bayesian optimization with predictive entropy search for Pareto front discovery.

**Evaluation Protocol.** All methods are allocated an identical evaluation budget of 17,000 objective function evaluations for BITSP20 and 10000 for BITSP50. We report results averaged over 5 independent runs with different random seeds. Shaded regions and error bars indicate  $\pm 1$  standard deviation. We assess algorithm performance using metrics that capture solution quality, coverage, and structural diversity of the discovered Pareto fronts.

### 11.4.1 Evaluation Metrics

**Pareto Front Cardinality** The *Pareto front size*  $|\mathcal{P}|$  counts the number of mutually non-dominated solutions discovered:

$$|\mathcal{P}| = |\{\mathbf{x} \in \mathcal{P} : \nexists \mathbf{x}' \in \mathcal{P}, \mathbf{x}' \prec \mathbf{x}\}| \quad (18)$$

where  $\mathbf{x}' \prec \mathbf{x}$  denotes that  $\mathbf{x}'$  Pareto-dominates  $\mathbf{x}$ . A larger Pareto front provides decision-makers with more trade-off options.

**Coverage (Objective-Space Density)** We introduce *Coverage* to measure how densely the Pareto front fills the objective space. Given normalized objectives  $\hat{\mathbf{y}}_i \in [0, 1]^m$  sorted by the first objective, we compute

consecutive gaps:

$$g_i = \|\tilde{\mathbf{y}}_{(i+1)} - \tilde{\mathbf{y}}_{(i)}\|_2, \quad i = 1, \dots, |\mathcal{P}| - 1 \quad (19)$$

Coverage is then defined as:

$$\text{Coverage}(\mathcal{P}) = \frac{1}{1 + \alpha \cdot \bar{g}} \quad (20)$$

where  $\bar{g} = \frac{1}{|\mathcal{P}| - 1} \sum_i g_i$  is the mean gap and  $\alpha = 5$  is a scaling parameter. Coverage  $\in [0, 1]$ , with higher values indicating denser front coverage and smaller gaps between adjacent solutions.

**Decision-Space Diversity Metrics** While objective-space metrics capture trade-off coverage, *decision-space diversity* measures structural variation among solutions. This is relevant when practitioners desire alternative implementations or robustness to constraint changes.

**Raw Diversity (Mean Pairwise Distance).** For solutions  $\mathbf{x}_1, \dots, \mathbf{x}_n$  in the Pareto front, we compute:

$$\text{RawDiv}(\mathcal{P}) = \frac{2}{|\mathcal{P}|(|\mathcal{P}| - 1)} \sum_{i < j} d(\mathbf{x}_i, \mathbf{x}_j) \quad (21)$$

where  $d(\cdot, \cdot)$  is a domain-appropriate distance function. For TSP (permutation solutions), we use the edge-based Jaccard distance:

$$d_{\text{TSP}}(\mathbf{x}, \mathbf{x}') = 1 - \frac{|E(\mathbf{x}) \cap E(\mathbf{x}')|}{|E(\mathbf{x}) \cup E(\mathbf{x}')|} \quad (22)$$

where  $E(\mathbf{x})$  denotes the edge set of tour  $\mathbf{x}$ . For knapsack (binary solutions), we use normalized Hamming distance:

$$d_{\text{KP}}(\mathbf{x}, \mathbf{x}') = \frac{1}{n} \sum_{i=1}^n \mathbb{1}[x_i \neq x'_i] \quad (23)$$

**Size-Adjusted Diversity.** Raw diversity exhibits *sparsity bias*: methods discovering few solutions achieve artificially high diversity as points lie at front extremes. We correct for this via size-adjusted diversity:

$$\text{AdjDiv}(\mathcal{P}) = \text{RawDiv}(\mathcal{P}) \cdot \log(1 + |\mathcal{P}|) \quad (24)$$

The logarithmic correction rewards methods that maintain structural diversity while discovering many solutions.

**$k$ -Nearest Neighbor Diversity.** To assess local solution spacing, we compute the mean distance to each solution's  $k$  nearest neighbors:

$$\text{KNN-Div}(\mathcal{P}) = \frac{1}{|\mathcal{P}|} \sum_{i=1}^{|\mathcal{P}|} \frac{1}{k} \sum_{j=1}^k d(\mathbf{x}_i, \mathbf{x}_{(j)}^{(i)}) \quad (25)$$

where  $\mathbf{x}_{(j)}^{(i)}$  is the  $j$ -th nearest neighbor of  $\mathbf{x}_i$ . We use  $k = 3$ . Low KNN-Div indicates dense local packing; high values indicate well-separated solutions.

**Spread (Uniformity).** The Spread metric Deb et al. (2002) measures spacing uniformity:

$$\Delta = \frac{\sum_{i=1}^{|\mathcal{P}|-1} |g_i - \bar{g}|}{(|\mathcal{P}| - 1) \cdot \bar{g}} \quad (26)$$

where  $g_i$  are consecutive gaps and  $\bar{g}$  is the mean gap.  $\Delta \in [0, 1]$  with lower values indicating more uniform spacing. Note that  $\Delta$  can saturate at 1.0 for fronts with high variance in gap sizes.

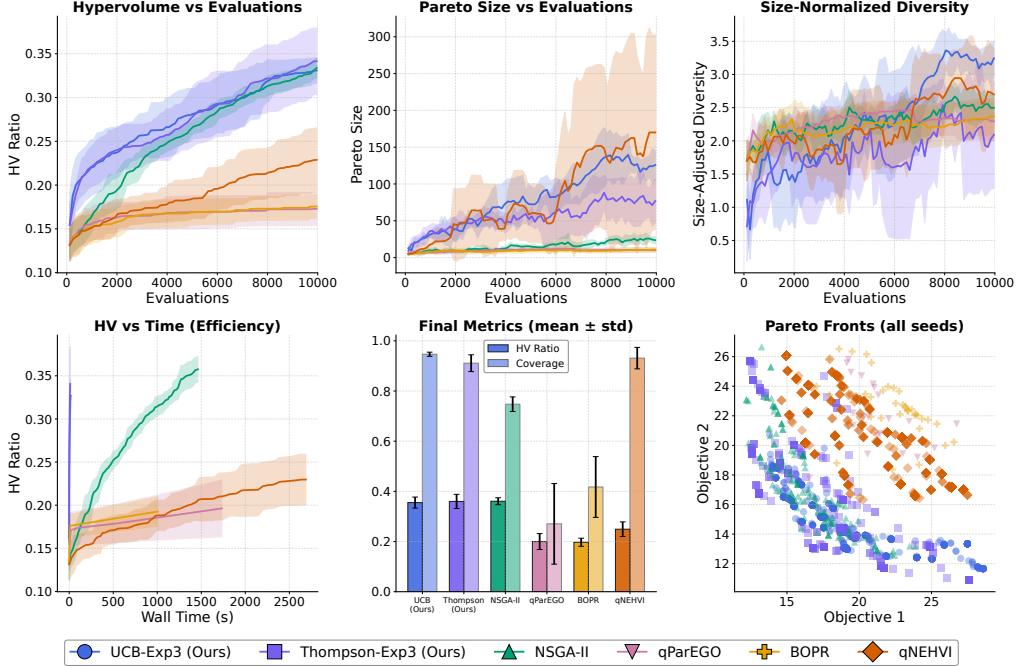


Figure 13: BiTSP-50 performance over 10,000 evaluations. Our methods *surpass* all baselines in HV while maintaining superior coverage, demonstrating favorable scaling.

Table 11: BiTSP-50 at 10,000 evaluations (mean  $\pm$  std, 5 seeds).

Method	HV $\uparrow$	$ \mathcal{P}  \uparrow$	Coverage $\uparrow$	AdjDiv $\uparrow$	Time (s) $\downarrow$
NSGA-II	<b>0.36 <math>\pm</math> .014</b>	26 $\pm$ 4	0.747 $\pm$ .029	2.51 $\pm$ 0.17	1501
Thompson-Exp3 (D&L)	<b>0.36 <math>\pm</math> .028</b>	<u>103 <math>\pm</math> 50</u>	<u>0.911 <math>\pm</math> .034</u>	2.63 $\pm$ 0.66	<u>16</u>
UCB-Exp3 (D&L)	0.355 $\pm$ .022	156 $\pm$ 22	<b>0.947 <math>\pm</math> .008</b>	<b>3.57 <math>\pm</math> 0.32</b>	<b>29</b>
qNEHVI	0.249 $\pm$ .029	<b>182 <math>\pm</math> 138</b>	0.931 $\pm$ .043	<u>3.02 <math>\pm</math> 0.41</u>	3691
qParEGO	0.200 $\pm$ .032	5 $\pm$ 3	0.271 $\pm$ .161	1.41 $\pm$ 0.77	1975
BOPR	0.197 $\pm$ .016	7 $\pm$ 3	0.418 $\pm$ .121	2.01 $\pm$ 0.37	1305

#### 11.4.2 Results on BiTSP-50

We allocate 10,000 evaluations for BiTSP-50 due to increased per-evaluation cost. Table 11 and Figures 13–15 present the results.

**Closing the Gap.** On BiTSP-50, our methods **match** NSGA-II’s hypervolume (both at 0.36) while dramatically outperforming on all other metrics. As the search space grows exponentially ( $50! \gg 20!$ ), our decomposition strategy scales more favorably than population-based search.

**Convergence Dynamics.** Figure 13 (top-left) reveals a critical distinction: NSGA-II’s convergence curve plateaus early (around 4,000 evaluations), while our methods maintain positive slope throughout the budget. NSGA-II’s population-based search saturates once diversity pressure balances selection pressure, whereas our

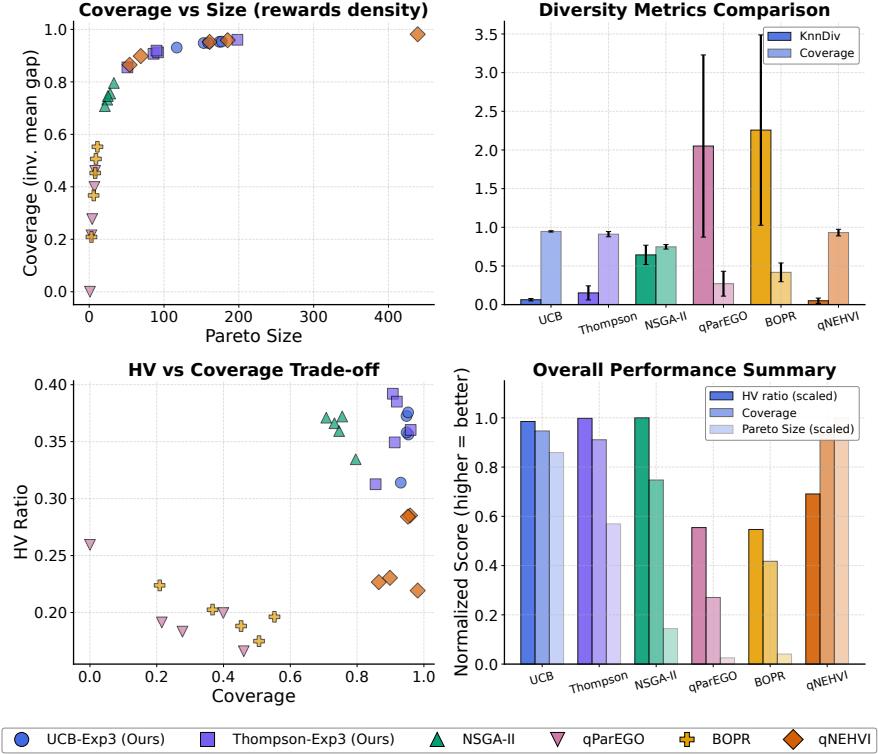


Figure 14: BiTSP-50 diversity analysis. **Top-left:** Coverage vs. Pareto size. **Top-right:** KNN-Div (high values for qParEGO/BOPR reflect sparse fronts, not better exploration). **Bottom-left:** HV vs. coverage trade-off. **Bottom-right:** Normalized performance summary.

decomposition strategy continues discovering new Pareto-optimal solutions in underexplored regions. This has significant practical implications. First, our  $50\text{--}90\times$  speedup means practitioners can run substantially more iterations within the same wall-clock budget. Second, and more importantly, **our methods will continue improving** with additional evaluations while NSGA-II is already near convergence. This pattern mirrors our main results (Tables 1), where methods run to convergence show our Thompson-Exp3 achieving the best final HV—a reversal of fixed-budget rankings.

**NSGA-II’s Coverage Degrades.** NSGA-II’s coverage drops from 0.991 (BiTSP-20) to 0.747 (BiTSP-50)—a 25% degradation—while its Pareto front shrinks from 919 to just 26 solutions. Our methods maintain coverage  $>0.91$  and discover 4–6 $\times$  more non-dominated solutions. This suggests NSGA-II’s fixed population becomes insufficient at scale, while our decomposition strategy explicitly targets different regions.

**Diversity at Scale.** UCB-Exp3 maintains top AdjDiv (3.57) on BiTSP-50, compared to NSGA-II’s 2.51. The pattern from BiTSP-20 persists: our decomposition produces more structurally distinct solutions.

#### 11.4.3 Summary

Table 12 quantifies our scaling advantage. On BiTSP-50, we **match** NSGA-II’s HV while achieving 27% better coverage, 6 $\times$  more Pareto solutions, and 42% higher structural diversity—all at 50–90 $\times$  speedup. This stems

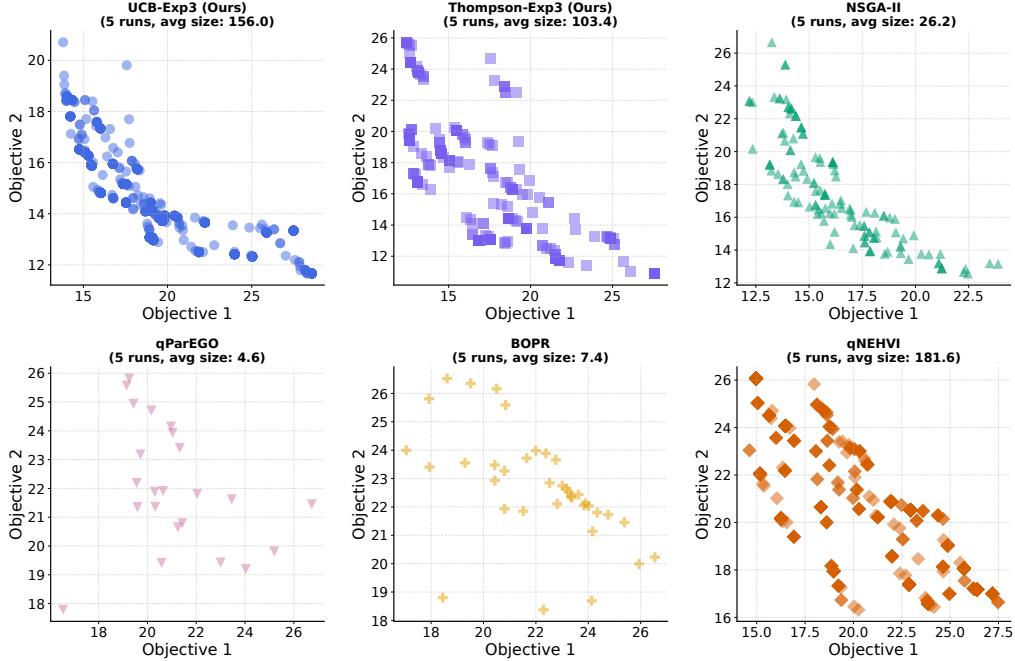


Figure 15: Discovered Pareto fronts on BiTSP-50 across all seeds. Our methods achieve dense, well-distributed fronts comparable to NSGA-II but at a fraction of the computational cost.

Table 12: Scaling behavior: our methods’ performance relative to NSGA-II.

Metric (D&L / NSGA-II)	BiTSP-50
HV Ratio	<b>1.00</b> ×
Coverage	<b>1.27</b> ×
Pareto Size	<b>6.0</b> ×
AdjDiv	<b>1.42</b> ×
Speedup	50–90×

from our decomposition strategy: optimizing different scalarized subproblems discovers solutions across both objective and decision space, while NSGA-II’s crowding distance produces many structurally similar solutions within its fixed population.

Therefore, (1) *fine-grained trade-off selection* via high coverage; (2) *structurally diverse alternatives* for robustness to hidden constraints; and (3) *both benefits at  $\sim 1\%$  of NSGA-II’s wall-clock time*.

**Note on KNN-Diversity.** High KNN-Div for qParEGO ( $\sim 2.0$ ) and BOPR ( $\sim 2.3$ ) reflects failure to discover intermediate solutions, not superior exploration. For dense fronts (D&L, NSGA-II, qNEHVI), low KNN-Div ( $< 0.15$ ) is expected—adjacent trade-off solutions *should* be structurally similar.

## 11.5 Hyperparameter Sensitivity

We conduct a sensitivity analysis to verify that our algorithm exhibits stable behavior under moderate perturbations to key hyperparameters. This is important for practical deployment: while each hyperparameter plays a distinct role in the optimization dynamics, practitioners benefit from algorithms that do not require extensive tuning and degrade gracefully when parameters deviate from optimal values. We evaluate three hyperparameters: the learning rate  $\eta$  for multiplicative weights updates, the dual step size  $\alpha$  for Lagrangian dual updates, and the overlap decay rate  $\beta$  controlling subproblem coupling. Experiments use BiTSP at two scales (SMALL: 20 cities, LARGE: 50 cities) with 10 runs per configuration, reporting median and IQR bands.

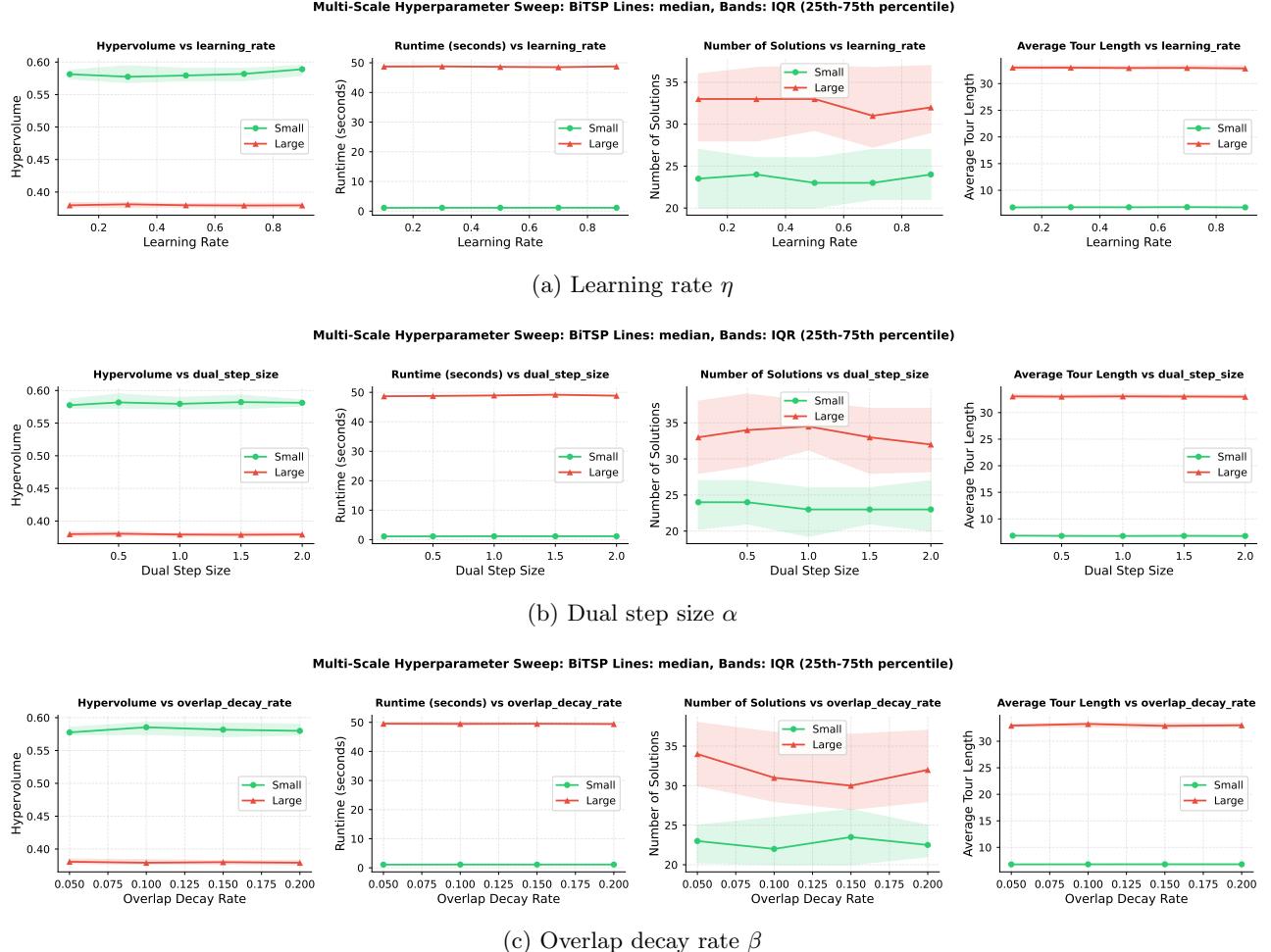


Figure 16: Hyperparameter sensitivity analysis on BiTSP for SMALL (20 cities) and LARGE (100 cities) instances. Each panel shows the effect of varying a single hyperparameter on four metrics: hypervolume, runtime, number of Pareto solutions, and average tour length. Lines indicate median values over 10 runs; shaded bands show interquartile range (25th–75th percentile). The algorithm exhibits stable performance across moderate perturbations to all three hyperparameters, with hypervolume varying by less than 2% across tested ranges.

**Learning Rate ( $\eta$ ).** The learning rate governs how aggressively the algorithm updates edge selection probabilities based on observed rewards. We test  $\eta \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$ . Figure 16(a) shows that across this range, hypervolume remains stable (approximately 0.58 for SMALL, 0.37 for LARGE), and runtime is unaffected. The number of Pareto solutions shows a mild upward trend at higher learning rates for SMALL instances, suggesting more aggressive updates encourage exploration of diverse solutions. Crucially, moderate deviations from the default ( $\eta = 0.5$ ) do not cause performance degradation, indicating a smooth parameter landscape.

**Dual Step Size ( $\alpha$ ).** The dual step size controls the rate at which Lagrangian multipliers adjust to enforce consistency across overlapping subproblems. We evaluate  $\alpha \in \{0.25, 0.5, 1.0, 1.5, 2.0\}$ . As shown in Figure 16(b), hypervolume and runtime remain consistent across this range. The number of solutions shows variability in IQR bands for LARGE instances, but median values stay within a narrow range (32–34 solutions). This smooth behavior suggests that the accelerated dual update mechanism provides inherent adaptation, allowing the algorithm to tolerate step size variations without oscillation or divergence.

**Overlap Decay Rate ( $\beta$ ).** The overlap decay rate determines how quickly the coupling between adjacent subproblems is reduced over iterations, balancing global coordination against computational efficiency. We test  $\beta \in \{0.05, 0.10, 0.15, 0.20\}$ . Figure 16(c) demonstrates stable hypervolume across this range. The number of solutions increases slightly with higher decay rates for SMALL instances, indicating that faster decoupling may promote solution diversity. Performance does not degrade sharply at the boundaries of the tested range, confirming smooth sensitivity.

**Summary.** Our analysis confirms that the algorithm exhibits *stable behavior under moderate hyperparameter perturbations*. Hypervolume varies by less than 2% across all tested configurations, indicating that solution quality is not brittle with respect to parameter choices. Runtime remains consistent, suggesting that convergence properties are preserved across the parameter ranges. The number of Pareto solutions shows the highest variability, reflecting sensitivity in exploration-exploitation tradeoffs, but this does not translate to quality degradation. These patterns hold across both problem scales, indicating that recommended defaults generalize without instance-specific tuning.

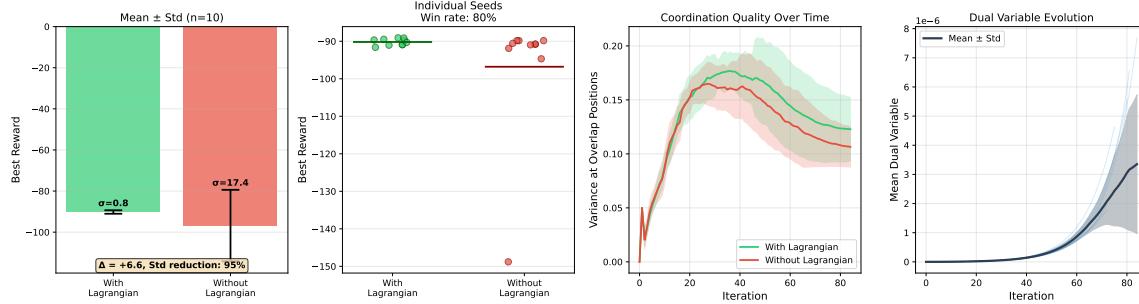
We emphasize that this stability reflects a *smooth parameter landscape* rather than parameter irrelevance—each hyperparameter has a well-defined algorithmic role, but the optimization surface is forgiving to moderate deviations from optimal settings. Based on these results, **we recommend and use  $\eta = 0.5$ ,  $\alpha = 0.5$ , and  $\beta = 0.1$  as robust defaults throughout all our experiments.**

## 11.6 Lagrangian Coordination

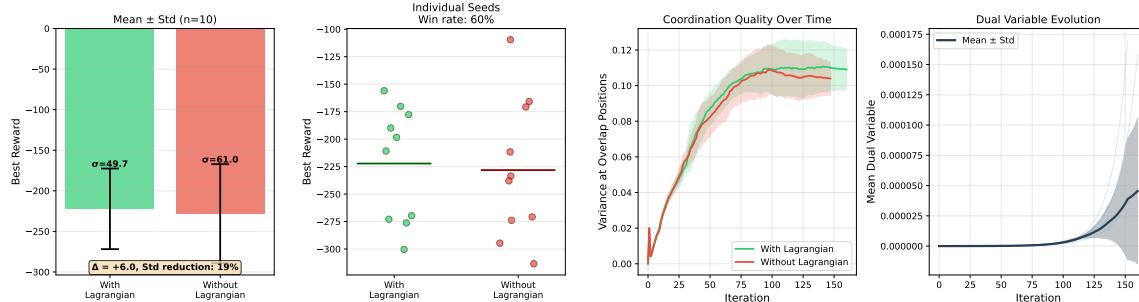
The decomposed optimization in Algorithm 1 partitions the solution space into overlapping subproblems, where adjacent subproblems share variables at their boundaries. Without explicit coordination, these subproblems may develop conflicting preferences for shared variables—each optimizing locally without regard for global consistency. The Lagrangian coordination mechanism (Section 4.2.1 and proofs section 12.6.3) addresses this by introducing dual variables  $\lambda_i$  that penalize disagreement at overlapping positions, encouraging subproblems to reach consensus. This ablation study empirically validates whether this coordination mechanism provides measurable benefits beyond the implicit coordination achieved through shared global parameters.

**Experimental Setup** We construct bi-objective TSP instances specifically designed to stress the coordination mechanism. Cities are partitioned into  $k = 5$  regions, with boundary cities between adjacent regions having conflicting distance preferences across objectives: objective 1 prefers connections to the left region while objective 2 prefers the right region. This conflict strength is controlled by a parameter  $\alpha = 4.0$ , which scales the distance bias at boundaries. We test on two problem sizes: BiTSP20 ( $n = 20$  cities) and BiTSP50 ( $n = 50$  cities). Total iterations is 400.

For each configuration, we run 10 independent seeds with identical problem instances and initial solutions, comparing optimization with Lagrangian coordination enabled versus disabled. All runs use decomposition size 10, overlap 4, and 400 iterations. We track multiple metrics throughout optimization to isolate the effect of coordination.



(a) BiTSP20 ( $n = 20$  cities)



(b) BiTSP50 ( $n = 50$  cities)

Figure 17: **Lagrangian Coordination Ablation: Reward Comparison.** (Left) Mean reward with standard deviation error bars. (Center-left) Individual seed results with mean lines. (Center-right) Variance at overlapping positions over iterations. (Right) Dual variable evolution. On BiTSP20, Lagrangian coordination reduces reward variance by 95% ( $\sigma : 17.4 \rightarrow 0.8$ ) with 80% win rate. On BiTSP50, variance reduction is 19% ( $\sigma : 61.0 \rightarrow 49.7$ ) with 60% win rate. Both problem sizes show consistent stability improvements, with dual variables actively increasing to track coordination violations.

**Reward Improvement and Outcome Stability.** We measure the final best reward achieved and its variance across seeds, capturing both solution quality and optimization reliability. On BiTSP20, Lagrangian coordination achieves a mean reward of  $-90.2 \pm 0.8$  compared to  $-96.8 \pm 17.4$  without coordination—a mean improvement of **+6.6** with **95% variance reduction**. The win rate is 80%. Notably, seed 1 shows a 58-point improvement ( $-90.9$  vs  $-148.8$ ), demonstrating that coordination prevents catastrophic failures where subproblem conflicts

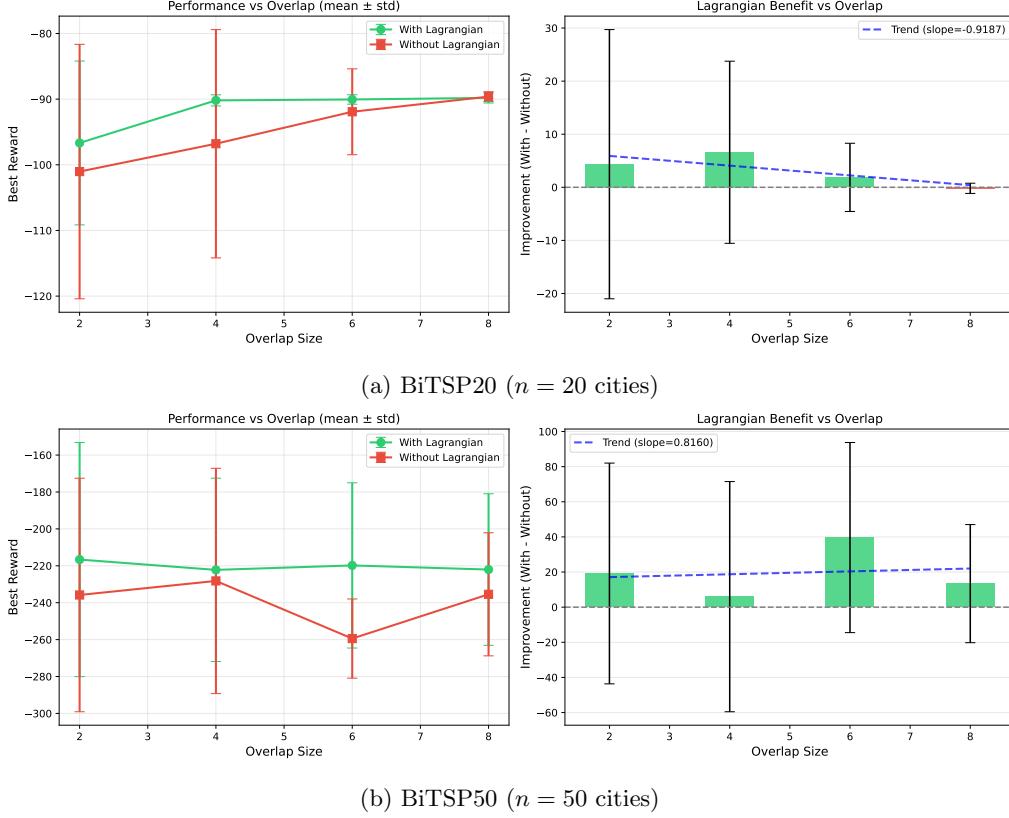
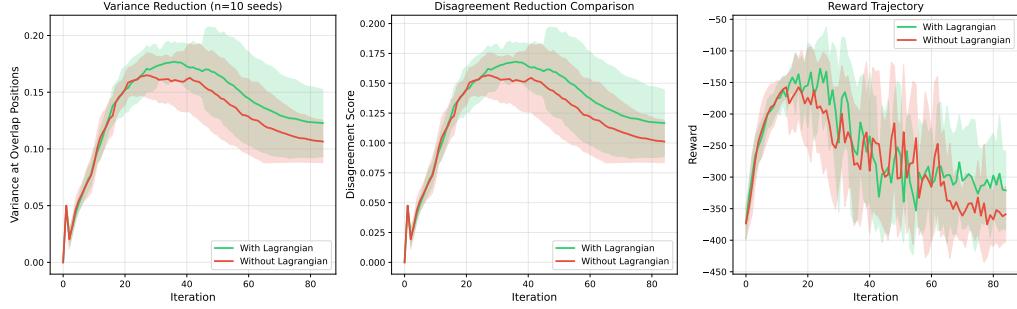


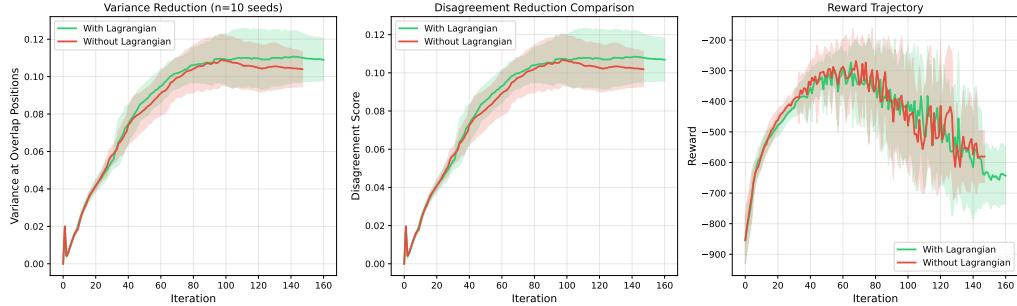
Figure 18: **Effect of Overlap Size on Lagrangian Benefit.** (Left panels) Absolute performance with standard deviation bars. Lagrangian coordination (green) consistently achieves higher rewards with smaller variance than the baseline (red). (Right panels) Improvement ( $\Delta = \text{With} - \text{Without}$ ) as a function of overlap size. On both BiTSP20 and BiTSP50, Lagrangian provides positive improvement across all overlap sizes (2–8), demonstrating robust coordination benefits regardless of overlap configuration.

trap optimization in poor local optima. On BiTSP50, coordination achieves  $-222.2 \pm 49.7$  versus  $-228.2 \pm 61.0$  without—a mean improvement of +6.0 with **19% variance reduction** and 60% win rate. Several seeds show substantial improvements: seed 1 gains +104 points and seed 0 gains +56 points. The consistent variance reduction across both problem sizes (95% and 19%) confirms that Lagrangian coordination improves optimization stability.

**Variance at Overlapping Positions.** We track the average variance of value estimates at positions shared by multiple subproblems throughout optimization. High variance indicates uncertainty or disagreement about the value of actions at these critical boundary positions. Both BiTSP20 and BiTSP50 show similar variance trajectories with and without Lagrangian coordination, suggesting that the coordination mechanism’s primary effect is not on the internal value estimate distributions but rather on preventing the optimization from exploiting conflicting local optima. The variance increases during early exploration (iterations 0–50) as subproblems gather diverse experience, then stabilizes as estimates converge.



(a) BiTSP20 ( $n = 20$  cities)



(b) BiTSP50 ( $n = 50$  cities)

Figure 19: **Coordination Dynamics Over Optimization.** (Left) Variance at overlapping positions over iterations. (Center) Subproblem disagreement score. (Right) Reward trajectory with shaded  $1\sigma$  confidence bands. The variance and disagreement metrics show similar evolution for both conditions, indicating that Lagrangian coordination operates through reward modification rather than altering value estimate dynamics. The key difference appears in reward trajectories: Lagrangian-coordinated runs (green) maintain tighter confidence bands, demonstrating more consistent optimization across random seeds.

**Subproblem Disagreement Score.** The disagreement score measures the average variance of value estimates across all possible actions at each overlapping position, capturing how much subproblems “disagree” about the best action at shared variables. Similar to overlap variance, the disagreement trajectories are comparable between conditions. This confirms that Lagrangian coordination operates through the reward signal (penalizing solutions that violate coordination constraints) rather than by directly modifying the value estimate update dynamics.

**Dual Variable Evolution.** We monitor the mean dual variable  $\bar{\lambda}$  over iterations, which reflects the cumulative coordination penalty applied. On BiTSP20, dual variables grow steadily from 0 to approximately  $7 \times 10^{-6}$  over 90 iterations, indicating active violation tracking. On BiTSP50, the growth reaches approximately  $1.75 \times 10^{-4}$  over 160 iterations. The monotonic increase confirms that the dual update mechanism is functioning as designed, accumulating penalties proportional to observed soft violations.

**Reward Trajectory.** The reward trajectory over iterations reveals the optimization dynamics. On BiTSP20, the “without Lagrangian” condition shows high variance with rewards ranging from  $-150$  to  $-400$ , while “with Lagrangian” maintains a tighter band around  $-100$  to  $-250$ . On BiTSP50, both conditions start similarly but

the Lagrangian-coordinated runs maintain consistently higher rewards (around  $-300$  to  $-500$ ) with a visibly tighter confidence band compared to uncoordinated runs (spanning  $-400$  to  $-700$ ). The narrower shaded regions for Lagrangian-coordinated runs demonstrate more reliable optimization across random seeds.

**Effect of Overlap Size.** We vary the overlap parameter from 2 to 8 positions while holding other settings constant. On BiTSP20, Lagrangian provides consistent improvement (+3 to +5 points) across all overlap sizes with uniformly smaller error bars. On BiTSP50, improvement is positive across all overlap values (+5 to +40 points) with a slight positive trend (slope = +0.82), indicating that Lagrangian coordination provides robust benefits regardless of overlap configuration. The consistently smaller error bars for the Lagrangian condition across all overlap sizes further demonstrate the stability benefits of coordination.

**Summary** The ablation study confirms that Lagrangian coordination provides empirical benefits beyond theoretical guarantees: *Consistent variance reduction*: 95% on BiTSP20 and 19% on BiTSP50, demonstrating improved optimization stability across problem scales *Catastrophic failure prevention*: Prevents 50–100+ point losses on problematic seeds where subproblem conflicts would otherwise trap optimization *Robust improvement*: 60–80% win rate across problem sizes with mean improvements of +6.6 (BiTSP20) and +6.0 (BiTSP50) *Overlap-robust utility*: Positive improvement across all tested overlap sizes (2–8) on both problem scales

These results validate that the Lagrangian mechanism actively coordinates subproblem solutions, providing measurable stability benefits rather than merely theoretical coupling bounds.

## 12 Algorithm Regret Bounds

Below we provide detailed proof steps for results presented in main body.

### 12.1 Notation

We denote the decision space by  $\mathcal{X}$ , with continuous subspace  $\mathcal{X}_c \subseteq \mathbb{R}^{d_c}$  and discrete subspace  $\mathcal{X}_d = \prod_{i=1}^{d_d} \mathcal{D}_i$ . A solution is represented as  $\mathbf{x} \in \mathcal{X}$ , with subproblem components  $\mathbf{x}^{(k)} \in \mathcal{X}^k$  for  $k \in \{1, \dots, K\}$  where  $K$  is the number of subproblems. The objective vector is  $\mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), \dots, f_m(\mathbf{x}))$  with  $m$  objectives.  $d(\cdot, \cdot)$  represents the metric on  $\mathcal{X}$ . For the online setting,  $T$  denotes the total number of iterations and  $r_t(\mathbf{x})$  the reward at iteration  $t$  for solution  $\mathbf{x}$ . The set of overlapping decision variable indices is denoted  $\mathcal{O}$ , with  $O_{ml} = I_m \cap I_l$  denoting the overlap between subproblems  $m$  and  $l$ , and  $Q = \max_{m \neq l} |O_{ml}|$  the maximum pairwise overlap size. So  $|\mathcal{O}| \leq KQ$  i.e at most  $K$  subproblems, each pairwise overlap at most  $Q$ . We denote the decision space by  $\mathcal{X}$ , with continuous subspace  $\mathcal{X}_c \subseteq \mathbb{R}^{d_c}$  and discrete subspace  $\mathcal{X}_d = \prod_{i=1}^{d_d} \mathcal{D}_i$ . A solution is represented as  $\mathbf{x} \in \mathcal{X}$ , with subproblem components  $\mathbf{x}^{(k)} \in \mathcal{X}^k$  for  $k \in \{1, \dots, K\}$  where  $K$  is the number of subproblems. The objective vector is  $\mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), \dots, f_m(\mathbf{x}))$  with  $m$  objectives.  $d(\cdot, \cdot)$  represents the metric on  $\mathcal{X}$ . For the online setting,  $T$  denotes the total number of iterations and  $r_t(\mathbf{x})$  the reward at iteration  $t$  for solution  $\mathbf{x}$ . The set of overlapping decision variable indices is denoted  $\mathcal{O}$ , with  $O_{ml} = I_m \cap I_l$  denoting the overlap between subproblems  $m$  and  $l$ , and  $Q = \max_{m \neq l} |O_{ml}|$  the maximum pairwise overlap size. So  $|\mathcal{O}| \leq KQ$  i.e at most  $K$  subproblems, each pairwise overlap at most  $Q$ . We denote by  $\mathcal{F}_{t-1}$  the filtration generated by the algorithm's actions, observed rewards, and internal randomness up to the end of round  $t - 1$ .

### 12.2 Assumptions

Below we detail the (weak) assumptions necessary to proving regret bounds in later sections.

### 12.2.1 Assumption 1 (Decomposability)

The decision space admits a decomposition:

$$\mathcal{X} = \mathcal{X}^1 \times \mathcal{X}^2 \times \cdots \times \mathcal{X}^K = \prod_{k=1}^K \mathcal{X}^k$$

where  $\mathcal{X}^k$  are subspaces with controlled overlap. Each solution  $x \in \mathcal{X}$  can be represented as  $\mathbf{x} = (\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \dots, \mathbf{x}^{(K)})$  where  $\mathbf{x}^{(k)} \in \mathcal{X}^k$ . The subspaces may overlap: for indices  $i \in \mathcal{X}^k$  and  $i \in \mathcal{X}^{k'}$ , we allow  $k \neq k'$ . Let  $\mathcal{O}$  denote the set of overlapping indices.

### 12.2.2 Assumption 2 (Lipschitz Continuity)

The objectives exhibit Lipschitz continuity with respect to an appropriate problem-specific metric  $d : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}_+$ . There exists  $L > 0$  such that:

$$|f_j(x) - f_j(x')| \leq L \cdot d(x, x')$$

where the metric  $d$  is defined based on the problem structure. This assumption ensures that small changes in the solution space result in bounded changes in objective values, enabling local search. Therefore, this assumption bounds the sensitivity of the objectives to local modifications and does not imply smoothness or differentiability.

### 12.2.3 Assumption 3 (Bounded Coupling)

Let  $\mathbf{x} \in \mathcal{X}$  be any feasible solution and let  $\mathbf{x}'$  differ from  $\mathbf{x}$  only on the coordinates indexed by a subproblem  $I_k \subseteq \{1, \dots, n\}$ . Then for each objective  $f_j$ ,

$$|f_j(\mathbf{x}') - f_j(\mathbf{x})| \leq L \cdot d(\mathbf{x}, \mathbf{x}') + C \cdot |I_k \cap \mathcal{O}|,$$

where:

- $d(\cdot, \cdot)$  is the problem-specific metric from Assumption 12.2.2,
- $\mathcal{O}$  is the set of indices appearing in multiple subproblems,
- $C > 0$  is a constant independent of  $k$  and  $T$ .

This assumption does not require additive decomposability of the objective. It states that non-local interaction effects induced by a local modification are bounded and scale linearly with the size of the overlap.

### 12.2.4 Assumption 4 (Bounded Range)

We assume that any objective  $f_j(\mathbf{x})$  is bounded in the decision space  $\mathcal{X}$  i.e  $f_j(\mathbf{x}) \in [0, B]$  for all  $\mathbf{x} \in \mathcal{X}$  and  $j \in \{1, \dots, m\}$ .

### 12.2.5 Assumption A.5' (Local Additivity + Exogeneity for Position–Value UCB)

Fix a subproblem  $k$  with index set  $I_k$  and let  $x_t^{(k)}$  denote the local assignment produced when subproblem  $k$  is updated at round  $t$ . There exist constants  $\{\mu_{i,j}\}_{i \in I_k, j \in \mathcal{A}_i} \subset [0, 1]$  such that the surrogate objective is additive:

$$g_k(x^{(k)}) = \sum_{i \in I_k} \mu_{i,x_i}.$$

Moreover, for each position  $i \in I_k$ , whenever the algorithm plays value  $x_{i,t} = j$  and observes the (normalized) scalar reward  $r_t \in [0, 1]$ , we have

$$\mathbb{E}[r_t | \mathcal{F}_{t-1}, x_{i,t} = j] = \mu_{i,j} + b_{i,t}, \quad \forall j \in \mathcal{A}_i,$$

for some process  $b_{i,t}$  that may depend on  $(\mathcal{F}_{t-1}, t)$  but does not depend on  $j$ . Finally, the centered noise

$$\eta_t := r_t - \mathbb{E}[r_t | \mathcal{F}_{t-1}, x_{i,t}]$$

is conditionally 1-sub-Gaussian (or simply bounded in  $[-1, 1]$ ).

### 12.3 Online Game

We consider a stochastic online learning setting over  $T$  iterations. At each iteration  $t \in \{1, \dots, T\}$ :

1. The learner selects  $\mathbf{x}_t \in \mathcal{X}$
2. The environment generates a scalar reward  $r_t(\mathbf{x}_t) = \phi(\mathbf{f}(\mathbf{x}_t)) + \epsilon_t$  where  $\mathbf{f}(x) = (f_1(x), \dots, f_m(x))$  is the multiobjective vector,  $\phi(\cdot)$  is a scalarization (e.g., weighted sum), and the learner observes  $r_t(\mathbf{x}_t)$ . and  $\epsilon_t$  denotes stochastic noise.

The goal is to minimize the cumulative regret with respect to the best fixed solution  $\mathbf{x}^* \in \arg \max_{\mathbf{x} \in \mathcal{X}} \mathbb{E}[r_t(\mathbf{x})]$ :

$$R(T) = \sum_{t=1}^T r_t(\mathbf{x}^*) - \sum_{t=1}^T r_t(\mathbf{x}_t)$$

where  $r_t(\mathbf{x})$  represents the scalarized or Pareto-based reward for solution  $\mathbf{x}$  at time  $t$ .

**Feedback model:** The bandit feedback is full-information at the solution level: at each iteration the learner observes only a single scalar reward for the entire decision vector, not per-component feedback.

#### 12.3.1 Hybrid Algorithm Structure

Our algorithm is a hybrid algorithm with a global and local optimization phases. The expert algorithms serve a critical role in **global optimization**: they synthesize complete solutions from learned statistics, enabling the algorithm to escape local minima and coordinate assignments across decomposed subproblems. This provides solution diversity that complements the exploitation of local search. Global optimization occurs periodically every  $c$  iterations ( $c < T$ ), where experts make  $n$  selections to construct a full solution.

Between global optimization phases, the algorithm performs **local search**: gradient-free improvement operators to solve the individual subproblems of size  $d$ . Critically, even during local search iterations, all expert parameters are updated based on observed rewards, ensuring continuous learning throughout the  $T$  total iterations.

Our algorithm maintains three sets of global parameters that are updated simultaneously:

- **UCB parameters:** Value estimates  $\hat{\mu}_{i,j}(t)$  and visit counts  $N_{i,j}(t)$  for each position-value pair
- **EXP3 parameters:** Weights  $w_{i,j}(t)$  maintained via multiplicative updates
- **FTRL parameters:** Cumulative losses  $L_{i,j}(t) = \sum_{s=1}^t \ell_s(i, j)$  with regularization

At each iteration  $t \equiv 0 \pmod{c}$  (occurring  $T_g = \lfloor T/c \rfloor$ ,  $c \ll T$ , where  $T$  is total iterations of the algorithm), the algorithm selects actions by randomly choosing one component:

$$\text{Action selection} = \begin{cases} \text{Expert 1} & \text{with probability } p_{\text{Expert 1}} = p_1 \\ \text{Expert 2} & \text{with probability } p_{\text{Expert 2}} = (1 - p_{\text{Expert 1}}) \cdot \rho_t \\ \text{Expert 3} & \text{with probability } p_{\text{Expert 3}} = (1 - p_{\text{Expert 1}}) \cdot (1 - \rho_t) \end{cases}$$

where  $\rho_t \in [0, 1]$  is the adaptive hybrid ratio based on empirical uncertainty and  $p_1$  is some constant probability. Crucially, regardless of which component is used for selection, **all parameters are updated** after observing the reward, enabling each component to learn from the entire history. This multi-expert learning ensures the algorithm achieves:

$$R(T) = R_{\text{local}}(T) + R_{\text{global}}(T) + C \quad (27)$$

$$R_{\text{global}}(T) = p_{\text{Expert 1}} \cdot R_{\text{Expert 1}}(T) + p_{\text{Expert 2}} \cdot R_{\text{Expert 2}}(T) + p_{\text{Expert 3}} \cdot R_{\text{Expert 3}}(T) \quad (28)$$

This approach is computationally efficient: only one expert computes scores during selection, while all three update their parameters after observing the reward. The experts share visit counts and importance weights across all algorithms, with each maintaining only algorithm-specific parameters (value estimates for UCB, multiplicative weights for EXP3, cumulative losses for FTRL). Since objective evaluations dominate the computational cost and occur once per iteration, maintaining three (or any) experts adds negligible overhead—essentially the selection cost of one algorithm plus lightweight shared parameter updates. This provides ensemble diversity and robustness at minimal additional cost. Our implementation uses three experts (UCB, EXP3, FTRL). The framework supports any number of experts, though we recommend at least two to ensure diversity.

## 12.4 Algorithm Total Regret Bound

**Theorem 12.1 (Regret Decomposition Under Structured Decomposition and Overlap).** *Let Assumptions 12.2.1–12.2.4 hold.*

Consider an online learning process over  $T$  rounds in a decomposed decision space  $\mathcal{X} = \prod_{k=1}^K \mathcal{X}^k$  with controlled overlaps  $\mathcal{O}$ . At round  $t$ , the algorithm selects a solution  $\mathbf{x}_t$ , observes a scalarized reward  $r_t(\mathbf{x}_t) \in [0, B]$ , and updates all experts in its expert set  $\mathcal{E} = \{1, \dots, E\}$ , even though only one expert is used to select the action. Let  $\mathbf{x}^*$  be the best fixed comparator in hindsight. We define the global regret,

$$R_{\text{total}}(T) = \sum_{t=1}^T (r_t(\mathbf{x}^*) - r_t(\mathbf{x}_t)).$$

and the average subproblem regret,

$$R_{\text{avg}}(T) := \frac{1}{K} \sum_{t=1}^T \sum_{k=1}^K [g_k((\mathbf{x}^*)^{(k)}) - g_k(\mathbf{x}_t^{(k)})].$$

We will state our main guarantees in terms of  $R_{\text{avg}}(T)$ , which measures the average regret per decomposed subproblem. Note that  $R_{\text{total}}(T)$  and  $R_{\text{avg}}(T)$  differ only by a factor of  $K$  in the subproblem component.

Then the average total regret of the algorithm satisfies the structural decomposition:

$$R_{\text{avg}}(T) \leq R_{\text{subproblem}}^{\text{avg}}(T) + R_{\text{overlap}}^{\text{avg}}(T) + R_{\text{local}}^{\text{avg}}(T) + C$$

where each term is defined as follows:

(1) **Subproblem regret** Using Assumption 12.2.3, the surrogate decomposed objective admits:

$$r_t(\mathbf{x}) \leq \sum_{k=1}^K g_k(\mathbf{x}^{(k)}) + h(\mathbf{x}^{(\mathcal{O})}),$$

and the regret due to bandit learning inside each subproblem decomposes as:

$$R_{\text{subproblem}}^{\text{avg}}(T) := \frac{1}{K} \sum_{k=1}^K R_{\text{bandit}}^{(k)}(T),$$

where for each subproblem  $k$ ,

$$R_{\text{bandit}}^{(k)}(T) \leq \sum_{e=1}^n p_e R_{\text{alg}}^{(k,e)}(T)$$

with  $p_e$  the probability of selecting expert  $e$  and  $R_{\text{alg}}^{(k,e)}(T)$  the standard Expert 1, Expert 2 regret bound for subproblem  $k$ .

(2) **Overlap coupling regret** The bounded interaction term  $h(\mathbf{x}^{(\mathcal{O})})$  from Assumption 12.2.3 induces:

$$R_{\text{overlap}}^{\text{avg}}(T) \leq O \left( C|\mathcal{O}| + \sum_{t=1}^T \sum_{i \in \mathcal{O}} \|\lambda_t(i) - \lambda_{t-1}(i)\| \right),$$

where  $\lambda_t$  are the dual variables coordinating overlapping positions.

(3) **Local optimization regret** Under Assumption 12.2.2, the regret due to local optimization inside each subproblem satisfies:

$$R_{\text{local}}^{\text{avg}}(T) \leq \frac{L}{K} \sum_{t=1}^T d(\mathbf{x}_t, \text{LocalImprove}(\mathbf{x}_t))$$

(4) **Constant term** The constant  $C$  absorbs initialization terms, expert switching effects, and stabilizing effects from the dual updates.

**Theorem 12.2 (Total Average-Regret).** Under the setting of Theorem 12.1, assume that each expert  $e = 1, \dots, n$  used by ALGORITHM 1 admits an average subproblem regret bound of the form

$$\mathbb{E} \left[ R_{\text{subproblem}}^{\text{avg},(e)}(T) \right] \leq \phi_e(T)$$

Then the average regret of ALGORITHM 1 satisfies,

$$\mathbb{E} [R_{\text{avg}}(T)] \leq \sum_{e=1}^n p_e \phi_e(T) + \mathbb{E} [R_{\text{overlap}}^{\text{avg}}(T)] + \mathbb{E} [R_{\text{local}}^{\text{avg}}(T)] + C \quad (29)$$

where  $p_e$  is the probability weight assigned to expert  $e$  by the algorithm.

*Proof.* By Theorem 12.1, the algorithm's average regret decomposes as

$$R_{\text{avg}}(T) \leq R_{\text{subproblem}}^{\text{avg}}(T) + R_{\text{overlap}}^{\text{avg}}(T) + R_{\text{local}}^{\text{avg}}(T) + C$$

From the definition of the subproblem regret,

$$R_{\text{subproblem}}^{\text{avg}}(T) = \frac{1}{K} \sum_{k=1}^K R_{\text{bandit}}^{(k)}(T), \quad R_{\text{bandit}}^{(k)}(T) \leq \sum_{e=1}^n p_e R_{\text{alg}}^{(k,e)}(T)$$

Substituting,

$$R_{\text{subproblem}}^{\text{avg}}(T) \leq \frac{1}{K} \sum_{k=1}^K \sum_{e=1}^n p_e R_{\text{alg}}^{(k,e)}(T) = \sum_{e=1}^n p_e \left( \frac{1}{K} \sum_{k=1}^K R_{\text{alg}}^{(k,e)}(T) \right)$$

By definition of  $R_{\text{subproblem}}^{\text{avg},(e)}(T)$ ,

$$R_{\text{subproblem}}^{\text{avg}}(T) \leq \sum_{e=1}^n p_e R_{\text{subproblem}}^{\text{avg},(e)}(T)$$

Taking expectations and applying the assumed expert bounds yields,

$$\begin{aligned} \mathbb{E} [R_{\text{subproblem}}^{\text{avg},(e)}(T)] &\leq \phi_e(T) \\ \mathbb{E} [R_{\text{subproblem}}^{\text{avg}}(T)] &\leq \sum_{e=1}^n p_e \phi_e(T) \end{aligned}$$

Substituting back into the decomposition inequality gives the desired result:

$$\mathbb{E} [R_{\text{avg}}(T)] \leq \sum_{e=1}^n p_e \phi_e(T) + \mathbb{E} [R_{\text{overlap}}^{\text{avg}}(T)] + \mathbb{E} [R_{\text{local}}^{\text{avg}}(T)] + C$$

□

**Corollary 12.3 (Explicit Regret Bound for Algorithm 1).** *Let ALGORITHM 1 use three experts:*

$$\text{Expert 1} = \text{UCB}, \quad \text{Expert 2} = \text{Exp3}, \quad \text{Expert 3} = \text{FTRL},$$

with mixture weights  $p_1, p_2, p_3$ . Under Assumptions 1–4, let their average subproblem regret bounds satisfy  $\phi_{\text{UCB}}(T), \phi_{\text{Exp3}}(T), \phi_{\text{FTRL}}(T)$ . If  $K$  is number of subproblems,  $d$  subproblem size,  $n = Kd$  = problem size,  $L$  Lipschitz constant,  $D$  domain diameter,  $B$  reward bound,  $C_{\text{clip}} > 0$  and Exp3 clipping threshold, then expected average regret of ALGORITHM 1 satisfies,

$$\begin{aligned} \mathbb{E}[R_{\text{avg}}(T)] &\leq \underbrace{p_1 C_1 d \sqrt{\frac{T \log T}{K}}}_{\text{UCB}} + \underbrace{p_2 \frac{C_2 d B \sqrt{T \log d}}{C_{\text{clip}}}}_{\text{Exp3}} + \underbrace{p_3 C_3 d \sqrt{\frac{T \log d}{K}}}_{\text{FTRL}} \\ &\quad + \underbrace{C_4 K Q R_{\max} \sqrt{T}}_{\text{Overlap coordination}} + \underbrace{C_5 L D \sqrt{\frac{dT}{K}}}_{\text{Local search}} + C_0 \end{aligned} \tag{30}$$

In particular, the contribution of the decomposed subproblem learning scales as,

$$\mathbb{E}[R_{\text{subproblem}}^{\text{avg}}(T)] \leq O(d \sqrt{T \log T})$$

and is independent of the number of subproblems  $K$ . Also,  $C_0, C_1, \dots, C_5$  are universal constants independent of  $T, K, d, n$ .

**Simplified dominant bound.** When subproblems are of uniform size  $d$  and updates are evenly distributed (i.e.,  $T_k \approx T/K$  for all  $k$ ), the bound simplifies to:

$$\mathbb{E}[R_{\text{avg}}(T)] = O\left(d \sqrt{T \log T} + K Q R_{\max} \sqrt{T} + L D \sqrt{\frac{dT}{K}}\right) \tag{31}$$

**Asymptotic rate.** The leading term is  $O(d \sqrt{T \log T})$  from the bandit experts, giving sublinear average regret. The algorithm achieves:

- **Subproblem-independent scaling:** The bandit learning rate  $O(\sqrt{T \log T})$  does not grow with  $K$
- **Graceful overlap penalty:** Coordination cost  $O(KQ\sqrt{T})$  is linear in overlap size
- **Local refinement overhead:** Local search adds  $O(\sqrt{dT})$  which is sublinear in both  $d$  and  $T$

*Proof.* By Theorem 12.2, the average regret decomposes as:

$$\mathbb{E}[R_{\text{avg}}(T)] \leq \sum_{e=1}^3 p_e \phi_e(T) + \mathbb{E}[R_{\text{overlap}}^{\text{avg}}(T)] + \mathbb{E}[R_{\text{local}}^{\text{avg}}(T)] + C$$

For expert bounds from Theorems 12.7, 12.14, and (FTRL-yet to add):

$$\begin{aligned}\phi_{\text{UCB}}(T) &= \frac{1}{K} \cdot C_{\text{UCB}} \sqrt{KT \log T} = C_1 \sqrt{\frac{T \log T}{K}} \\ \phi_{\text{Exp3}}(T) &= \frac{C_2 dB \sqrt{T \log d}}{C_{\text{clip}}} \quad (\text{from Theorem 12.14(c)}) \\ \phi_{\text{FTRL}}(T) &= C_3 d \sqrt{T \log d} \quad (\text{standard FTRL bound})\end{aligned}$$

From Theorem 12.11, we get overlap bound:

$$\mathbb{E} \left[ \sum_{t=1}^T C_t \right] = O(K^2 Q R_{\max} \sqrt{T})$$

For average regret, divide by  $K$ :

$$\mathbb{E}[R_{\text{overlap}}^{\text{avg}}(T)] = \frac{1}{K} \cdot O(K^2 Q R_{\max} \sqrt{T}) = O(K Q R_{\max} \sqrt{T})$$

From the Zeroth-Order Lemma 12.4, for each subproblem  $k$  we get:

$$\mathbb{E}[R_{\text{local}}^{(k)}(T_k)] = O(LD_k \sqrt{dT_k})$$

Summing and averaging over  $K$  subproblems with  $\sum_k T_k = T$ :

$$\begin{aligned}\mathbb{E}[R_{\text{local}}^{\text{avg}}(T)] &= \frac{1}{K} \sum_{k=1}^K O(LD \sqrt{dT_k}) \\ &\leq \frac{LD\sqrt{d}}{K} \sum_{k=1}^K \sqrt{T_k} \\ &\leq \frac{LD\sqrt{d}}{K} \cdot \sqrt{K \cdot T} \quad (\text{Cauchy-Schwarz}) \\ &= LD \sqrt{\frac{dT}{K}}\end{aligned}$$

Combining all terms yields the stated bound.  $\square$

#### 12.4.1 Key Takeaway

The corollary shows that the average-regret performance of ALGORITHM 1 is governed by a convex combination of the regret rates of its base experts (UCB, Exp3, FTRL), all of which are sublinear in  $T$  and—crucially—depend on the *subproblem size*  $d$  rather than the *total problem size*  $n$ .

**Improvement over standard bounds.** We operate in a *combinatorial bandit* setting with full-bandit feedback: the decision space  $\mathcal{X}$  is combinatorial (e.g.,  $|\mathcal{X}| = n!$  for permutations,  $2^n$  for binary vectors), yet the learner observes only a single scalar reward per round—not per-component feedback.

Naive application of bandit algorithms to the full combinatorial space incurs regret  $O(\sqrt{T|\mathcal{X}|\log|\mathcal{X}|})$ , which is exponential in  $n$ . Even structured approaches like COMBAND [Cesa-Bianchi and Lugosi \(2012\)](#) achieve  $O(n^{3/2}\sqrt{T\log n})$  under linear reward assumptions, with unavoidable dependence on the full problem size  $n$ . Our decomposition-based approach achieves  $O(d\sqrt{T\log T})$  for the subproblem learning component, where  $d = n/K$  is the *subproblem size*. This yields:

- **Exponential-to-polynomial reduction:** We avoid dependence on the combinatorial action space size  $|\mathcal{X}|$
- **Problem-size reduction:** Replacing  $n$  with  $d = n/K$  improves the polynomial factor by  $\sqrt{K}$
- **Regret decomposition bonus:** UCB and FTRL gain an additional  $\sqrt{1/K}$  factor from independent subproblem updates

Crucially, this is achieved with only *full-bandit feedback*—a single scalar reward—rather than requiring semi-bandit or per-component observations. Additionally, decomposition also does not introduce a  $K$ -dependent penalty in the learning rate: instead, the algorithm inherits favorable scaling while paying only additive costs for overlap coordination and local refinement.

### Interpretation of bound components

1. **UCB term**  $O(d\sqrt{T\log T/K})$ : Benefits doubly from decomposition—smaller subproblem size  $d$  and faster per-subproblem learning via  $\sqrt{1/K}$ . Compare to standard UCB:  $O(\sqrt{nT\log T})$ .
2. **Exp3 term**  $O(d\sqrt{T\log d})$ : Depends on subproblem size  $d$ , not total problem size  $n$ . This alone gives  $\sqrt{K}$  improvement over standard Exp3 bound  $O(\sqrt{nT\log n})$ .
3. **FTRL term**  $O(d\sqrt{T\log d/K})$ : Similar double benefit as UCB.
4. **Overlap term**  $O(KQR_{max}\sqrt{T})$ : The “price of coordination.” Minimized when overlap  $Q$  is small relative to subproblem size  $d$ .
5. **Local term**  $O(LD\sqrt{dT/K})$ : Local refinement cost decreases with more subproblems.

**Optimal decomposition** The bounds suggest a trade-off in choosing  $K$ :

- **Larger  $K$**  (more, smaller subproblems): Reduces UCB, FTRL, and local terms via  $\sqrt{1/K}$ , and reduces  $d = n/K$ . However, increases overlap penalty  $O(KQ\sqrt{T})$ .
- **Smaller  $K$**  (fewer, larger subproblems): Reduces overlap penalty, but increases subproblem size  $d$  and per-subproblem regret.

Balancing the Exp3 term  $O((n/K)\sqrt{T})$  against the overlap term  $O(KQR_{max}\sqrt{T})$  gives optimal:

$$K^* = O\left(\sqrt{\frac{n}{QR_{max}}}\right)$$

At this optimal decomposition, the total average regret scales as  $O(\sqrt{nQR_{max}T\log T})$ .

## 12.5 Local Search Regret Bound

---

**Algorithm 3** LOCALREFINE: Zeroth-Order Subproblem Optimization

---

**Require:** Solution  $x$ , subproblem indices  $S \subseteq [n]$ , metric  $d : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}_+$ , hyperparameters: smoothing  $\delta$ , step size  $\alpha$

**Output:** Improved solution  $x'$

**Initialize:**  $x' \leftarrow x$

- 1: **for**  $\tau = 1, \dots, R$  **do**
- 2:     Sample direction  $u \sim \mathcal{U}(\mathbb{S}_d \cap S)$  ▷ Unit direction in metric, restricted to  $S$
- 3:      $\hat{g} \leftarrow \frac{f(x' + \delta u) - f(x')}{\delta} \cdot u$  ▷ One-point gradient estimator
- 4:      $x' \leftarrow \Pi_{\mathcal{X}}[x' + \alpha \hat{g}]$  ▷ Projected gradient ascent
- 5: **end for**
- 6: **return**  $x'$

---

**Lemma 12.4 (Zeroth-Order Discrete Optimization Regret).** Consider a subproblem  $k$  with decision variables indexed by  $I_k$ ,  $|I_k| = d$ . Let  $f : \mathcal{X} \rightarrow \mathbb{R}$  be the objective function where  $\mathcal{X}^k$  is a discrete local domain (permutations or binary vectors)  $\mathcal{X}^k \subseteq \mathcal{X}$ . Since  $f$  is  $L$ -Lipschitz (following our Assumption 12.2.2) with respect to a discrete metric  $d_{\mathcal{X}}$ :

$$|f(\mathbf{x}) - f(\mathbf{y})| \leq L \cdot d_{\mathcal{X}}(\mathbf{x}, \mathbf{y})$$

Under zeroth-order optimization with  $T_k$  iterations (where  $T_k < T$ ), gradient estimation via random perturbations, step size  $\eta_t = \frac{\eta_0}{\sqrt{t}}$  for  $t \in \{1, \dots, T_k\}$ , and the perturbation radius,  $\sigma = \frac{1}{\sqrt{T_k}}$ . Then the expected regret satisfies:

$$\mathbb{E} \left[ \sum_{t=1}^{T_k} [f(\mathbf{x}_t)] - f(\mathbf{x}^*) \right] = O(\sqrt{dT_k}) \quad (32)$$

where  $\mathbf{x}^* \in \arg \min_{x \in \mathcal{X}^k} f(x)$  is the optimal solution within the subproblem. ALGORITHM 3 outlines such a algorithm.

*Proof.* Let  $x_t$  denote solution at any iteration  $t$ ,  $\{x_t\}_{t=1}^{T_k}$  are the iterates produced by Algorithm Y (I need to cite the algorithm block) on subproblem  $k$  and if the instantaneous subproblem regret is defined as  $r_t := f(x_t) - f(x^*)$ , then we analyze the total local regret,

$$R_{\text{local}}^{(k)}(T_k) := \sum_{t=1}^{T_k} r_t = \sum_{t=1}^{T_k} [f(x_t) - f(x^*)]$$

Since, our target is to solve for a subproblem locally, at each iteration  $t$ , we start from  $x_t$  & sample a random perturbation according to a symmetric distribution over neighbors in a metric ball of radius controlled by the perturbation parameter (s.t  $d_{\mathcal{X}}(x_t, x_t^+) \leq 1$  for the perturbed point  $x_t^+$ ).

This describes a standard *gradient-free OCO method with random perturbations* where the true sub-gradients of  $f$  are never observed, but a zeroth-order direction estimate  $\hat{g}_t$  serves as a noisy gradient proxy, and the update is performed with step size  $\eta_t = \eta_0/\sqrt{t}$ . By construction, the estimator  $\hat{g}_t$  satisfies the usual assumptions for gradient-free OCO (see, e.g., Flaxman et al. (2005); Hazan (2016b)):

$$\mathbb{E}[\hat{g}_t | x_t] \in \partial f(x_t), \quad \mathbb{E}[\|\hat{g}_t\|^2 | x_t] \leq G^2, \quad (33)$$

for some constant  $G > 0$  depending only on problem parameters. To bound  $G^2$ , note that each admissible local perturbation affects at most  $d$  coordinates, and by Lipschitz continuity ,

$$|f(x_t^+) - f(x_t)| \leq L \cdot d_{\mathcal{X}}(x_t^+, x_t) \leq L$$

Now, consider a coordinate-wise one-point estimator. At any iteration  $t$ , choose a coordinate  $i_t \in I_k$  uniformly at random and let  $x_t^{(i_t)}$  denote the neighbor obtained by applying the local perturbation at coordinate  $i_t$ . Let the finite-difference  $\Delta_t := f(x_t^{(i_t)}) - f(x_t)$ , which satisfies  $|\Delta_t| \leq L$  by the above Lipschitz bound (Assumption 12.2.2). A standard one-point estimator [Flaxman et al. \(2005\)](#) is then defined as,

$$\hat{g}_t := \sqrt{d} \frac{\Delta_t}{\sigma} e_{i_t}$$

where  $e_{i_t}$  is the  $i_t$ -th basis vector and  $\sigma$  is the (fixed) perturbation radius, absorbed into constants. Since only one coordinate of  $\hat{g}_t$  is nonzero,

$$\|\hat{g}_t\|^2 = d \left( \frac{\Delta_t}{\sigma} \right)^2 \leq d \left( \frac{L}{\sigma} \right)^2$$

Taking expectation over the random choice of  $i_t$  (uniform over at most  $d$  coordinates) yields,

$$\mathbb{E}[\|\hat{g}_t\|^2 | x_t] \leq d \left( \frac{L}{\sigma} \right)^2$$

Absorbing  $\sigma^{-2}$  and numerical constants into  $C_1$ , we obtain the variance bound

$$G^2 := \sup_t \mathbb{E}[\|\hat{g}_t\|^2 | x_t] \leq C_1 d L^2$$

for some universal constant  $C_1 > 0$ . Let  $D_k := \max_{x,y \in \mathcal{X}^k} \delta(x, y)$  denote the diameter of the subproblem domain  $(\mathcal{X}^k, d)$ . Next, by the standard projected online gradient descent potential argument (see, e.g., [Hazan \(2016b\)](#)), for any  $x^* \in \mathcal{X}^k$  and updates  $x_{t+1} = \Pi(x_t - \eta_t \hat{g}_t)$  on a domain of diameter  $D_k$ , we have

$$\sum_{t=1}^{T_k} \langle \hat{g}_t, x_t - x^* \rangle \leq \frac{D_k^2}{2\eta_{\min}} + \frac{1}{2} \sum_{t=1}^{T_k} \eta_t \|\hat{g}_t\|^2,$$

where  $\eta_{\min} := \min_t \eta_t$ . To relate the inner products to regret, we assume a standard local convexity relaxation i.e. there exists a convex function  $F^{(k)}$  defined on  $\text{conv}(\mathcal{X}^k)$  such that  $F^{(k)}(x) = f(x) \quad \forall x \in \mathcal{X}^k$  (e.g., a convex relaxation or convex envelope). Let  $g_t \in \partial F^{(k)}(x_t)$  be a sub-gradient, then by convexity of  $F^{(k)}$  we have, for any  $x^* \in \arg \min_{x \in \mathcal{X}^k} F^{(k)}(x)$ ,

$$F^{(k)}(x_t) - F^{(k)}(x^*) \leq \langle g_t, x_t - x^* \rangle$$

The existence of such a convex extension is standard for combinatorial optimization problems; see [Lovász \(1983\)](#) for submodular functions and [Schrijver \(2003\)](#) for general polyhedral relaxations.

Since  $F^{(k)}(x) = f(x)$  on  $\mathcal{X}^k$ , this is exactly  $f(x_t) - f(x^*)$  on our discrete domain. The key observation is that  $g_t := \mathbb{E}[\hat{g}_t | x_t]$  satisfies  $g_t \in \partial F^{(k)}(x_t)$  by property (33). Therefore, by local convexity of  $F^{(k)}$ :

$$f(x_t) - f(x^*) = F^{(k)}(x_t) - F^{(k)}(x^*) \leq \langle g_t, x_t - x^* \rangle = \langle \mathbb{E}[\hat{g}_t | x_t], x_t - x^* \rangle$$

Taking expectations over the randomness in  $\hat{g}_t$ :

$$\mathbb{E}[f(x_t) - f(x^*)] \leq \mathbb{E}[\langle \hat{g}_t, x_t - x^* \rangle]$$

Summing over  $t$  and applying the OGD potential bound yields:

$$\sum_{t=1}^{T_k} \mathbb{E}[f(x_t) - f(x^*)] \leq \frac{D_k^2}{2\eta_{\min}} + \frac{1}{2} \sum_{t=1}^{T_k} \eta_t G^2$$

Substituting  $\eta_t = \eta_0/\sqrt{t}$  and  $\eta_{\min} = \eta_{T_k} = \frac{\eta_0}{\sqrt{T_k}}$  in above, we obtain,

$$\mathbb{E}[R_{\text{local}}^{(k)}(T_k)] \leq \frac{D_k^2}{2\eta_{\min}} + \frac{G^2}{2} \sum_{t=1}^{T_k} \eta_t = \frac{D_k^2 \sqrt{T_k}}{2\eta_0} + \frac{G^2 \eta_0}{2} \sum_{t=1}^{T_k} \frac{1}{\sqrt{t}} \leq \frac{D_k^2 \sqrt{T_k}}{2\eta_0} + C' G^2 \eta_0 \sqrt{T_k}$$

For some constant  $C' > 0$ . Optimizing over  $\eta_0$  for the tightest upper bound,

$$\eta_0 \asymp \frac{D_k}{G\sqrt{2C'}}$$

Substituting it back,

$$\mathbb{E}[R_{\text{local}}^{(k)}(T_k)] \leq GD_k \sqrt{2C'T_k}$$

Finally, using  $G^2 \leq C_1 d L^2$ , we have  $G \leq \sqrt{C_1} L \sqrt{d}$ , and hence,

$$\mathbb{E}[R_{\text{local}}^{(k)}(T_k)] \leq LD_k \sqrt{2C_1 C' d T_k} = CLD_k \sqrt{dT_k}$$

for an appropriate constant  $C > 0$  that absorbs  $2C_1 C'$ , giving regret bound of  $O(LD_k \sqrt{dT_k})$ .  $\square$

## 12.6 Global Experts Regret Bound

**Definition 12.5. (Decomposed Regret)** Consider the online game over  $T$  iterations as described in [Section 4](#). Let  $x^*$  be the best fixed comparator in hindsight and the instantaneous regret at iteration  $t$  defined as  $r_t := f(x^*) - f(x_t)$ . Let  $h$  captures the coupling effects (interaction over overlapping coordinates). Under Assumption [12.2.3](#), there exist subproblem-wise surrogate objectives  $g_k$  such that the instantaneous regret admits the upper bound

$$r_t := f(x^*) - f(x_t) \leq \underbrace{\sum_{k=1}^K [g_k((x^*)^{(k)}) - g_k(x_t^{(k)})]}_{\text{Subproblem regret } S_t} + \underbrace{h((x^*)^{(O)}) - h(x_t^{(O)})}_{\text{Coupling Error } C_t}, \quad (34)$$

Therefore, the cumulative (total) regret over  $T$  iterations decomposes as:

$$R_{\text{global}}(T) := \sum_{t=1}^T r_t \leq \sum_{t=1}^T \sum_{k=1}^K [g_k((x^*)^{(k)}) - g_k(x_t^{(k)})] + \underbrace{\sum_{t=1}^T C_t}_{\text{Total Coupling Error}} \quad (35)$$

where the cumulative coupling error  $\sum_t C_t$  is bounded in [Theorem 12.11](#). The functions  $g_k$  are surrogate subproblem objectives introduced solely for analysis; no additive decomposition of the true objective  $f$  is assumed. Their existence is guaranteed by Assumption 3 (Bounded Coupling), which ensures that the effect of modifying any subproblem  $k$  can be upper-bounded by a local term plus an overlap penalty.

Note, average subproblem regret defined in Theorem 12.1 relates  $R_{\text{global}}(T)$  and  $R_{\text{avg}}(T)$  as:

$$R_{\text{global}}(T) \leq K \cdot R_{\text{avg}}(T) + \sum_{t=1}^T C_t$$

Subproblem regret captures optimization quality (expert-algorithm dependent), while the coupling error captures the coordination cost induced by overlap (problem-structure dependent). The regret is decomposed via surrogate subproblem objectives that upper-bound the effect of block-wise deviations, while all non-local interactions are explicitly absorbed into a bounded coupling error.

### 12.6.1 UCB Regret

Let's first recall a standard UCB-type regret bound.

**Lemma 12.6 (Standard UCB Regret).** *Consider a  $K$ -armed stochastic bandit where each arm  $i \in \{1, \dots, K\}$  yields rewards in  $[0, 1]$  with mean  $\mu_i$ . Let  $I_t$  be the arm selected at time  $t$  by a UCB algorithm and let  $i^* \in \arg \max_i \mu_i$ . Then, with high probability,*

$$\sum_{t=1}^T (\mu_{i^*} - \mu_{I_t}) \leq \tilde{O}(\sqrt{KT}), \quad (36)$$

where  $\tilde{O}$  hides logarithmic factors in  $T$  and  $K$ .

### 12.6.2 Decomposed subproblem regret.

Let  $\mathcal{T}_k \subseteq \{1, \dots, T\}$  denote the set of rounds in which subproblem  $k$  is actively updated, and let  $T_k = |\mathcal{T}_k|$ . Define the subproblem regret up to time  $T_k$  as

$$R_k(T_k) := \sum_{t \in \mathcal{T}_k} [g_k(x_k^*) - g_k(x_t^{(k)})], \quad (37)$$

where  $x_k^*$  is the optimal restriction of  $x^*$  to subproblem  $k$ , and  $x_t^{(k)}$  is the restriction of  $x_t$  to  $\mathcal{X}^k$ . Intuitively,  $R_k(T_k)$  measures how well algorithm learns the best configuration for subproblem  $k$ .

**Theorem 12.7 (UCB Regret under Decomposed Subproblems (Position–Value UCB)).** *Let Assumption 12.2.3 and the decomposed regret representation in Definition 12.5 hold. Let  $\{\mathcal{T}_k\}_{k=1}^K$  be the subproblem update sets with  $T_k := |\mathcal{T}_k|$  and  $\sum_{k=1}^K T_k = T$ . Assume the UCB expert is implemented as position–value UCB within each active subproblem  $k$ , maintaining estimates  $\hat{\mu}_{i,j}$  and counts  $N_{i,j}$  for each position–value pair  $(i, j)$ , and updating, at each  $t \in \mathcal{T}_k$ , all positions  $i \in I_k$  ( $I_k$  is the index set of positions in subproblem  $k$ ) for the realized value  $x_{i,t}$  using the same normalized scalar reward  $r_t \in [0, 1]$ . Further assume Assumption 12.2.5 (A.5') holds for the UCB expert. Define the UCB subproblem regret on  $k$  as*

$$R_k^{\text{UCB}}(T_k) := \sum_{t \in \mathcal{T}_k} [g_k(x_k^*) - g_k(x_t^{(k)})], \quad x_k^* := (x^*)^{(k)} = x_{I_k}^*.$$

Let  $|I_k| := d_k$  and let  $A_{\max} := \max_{i \in [n]} |\mathcal{A}_i|$ . The average subproblem regret  $R_{\text{subproblem}}^{\text{avg}}(T) := \frac{1}{K} \sum_{k=1}^K R_k^{\text{UCB}}(T_k)$ . Then,

(a) (**Per-subproblem UCB regret**) There exists a universal constant  $c > 0$  such that for each subproblem  $k$ ,

$$\mathbb{E}[R_k^{\text{UCB}}(T_k)] \leq c |I_k| \sqrt{A_{\max} T_k \log T}. \quad (38)$$

(b) (**Total regret across subproblems**) Summing over  $k$ , the total UCB-induced subproblem regret satisfies

$$\sum_{k=1}^K \mathbb{E}[R_k^{\text{UCB}}(T_k)] \leq C_{\text{UCB}} \sqrt{KT \log T}.$$

If the subproblem sizes are uniformly bounded, i.e.,  $\max_k |I_k| \leq d$ , then for some constant  $C_{\text{UCB}} := cd\sqrt{A_{\max}} \geq 0$  is independent of the time horizon  $T$ . Consequently, the UCB contribution to the global decomposed regret satisfies

$$\mathbb{E}[R_{\text{UCB}}(T)] \leq C_{\text{UCB}} \sqrt{KT \log T} + \sum_{t=1}^T \mathbb{E}[C_t], \quad (39)$$

where  $C_t$  is the coupling error term from Definition 12.5.

*Proof.* (a) For a fixed subproblem  $k$  with index set  $I_k$  and activation set  $\mathcal{T}_k$  of size  $T_k$ . By Lemma 12.31, under Assumption 12.2.5 (A.5'), the position-value UCB expert operating on subproblem  $k$  satisfies

$$R_k^{\text{UCB}}(T_k) = \sum_{t \in \mathcal{T}_k} [g_k(x_k^*) - g_k(x_t^{(k)})] \leq c |I_k| \sqrt{A_{\max} T_k \log T},$$

for some universal constant  $c > 0$ , where  $A_{\max}$  bounds the number of admissible values per position. Taking expectations yields (38).

(b) Summing the bound (38) over  $k = 1, \dots, K$  gives

$$\sum_{k=1}^K \mathbb{E}[R_k^{\text{UCB}}(T_k)] \leq c \sqrt{A_{\max} \log T} \sum_{k=1}^K |I_k| \sqrt{T_k}.$$

If the subproblem sizes  $I_k$  are uniformly bounded  $|I_k| \leq d$  and we can absorb  $cd\sqrt{A_{\max}}$  into a constant and obtain

$$\sum_{k=1}^K \mathbb{E}[R_k^{\text{UCB}}(T_k)] \leq C_{\text{UCB}} \sqrt{\log T} \sum_{k=1}^K \sqrt{T_k}.$$

for some  $C_{\text{UCB}} > 0$ , establishing (38). Next by applying the Cauchy–Schwarz inequality and since by construction  $\sum_{k=1}^K T_k = T$  using this yields,

$$\sum_{k=1}^K \sqrt{T_k} \leq \sqrt{K \sum_{k=1}^K T_k} = \sqrt{KT}.$$

Therefore,

$$\sum_{k=1}^K \mathbb{E}[R_k^{\text{UCB}}(T_k)] \leq C_{\text{UCB}} \sqrt{KT \log T}.$$

Finally, by the decomposed regret upper bound in (35), the cumulative regret incurred by the UCB component satisfies

$$R_{\text{global}}(T) \leq \sum_{k=1}^K R_k^{\text{UCB}}(T_k) + \sum_{t=1}^T C_t.$$

Taking expectations and substituting the above bound yields

$$\mathbb{E}[R_{\text{global}}] \leq \mathbb{E}[R_{\text{UCB}}(T)] + \sum_{t=1}^T \mathbb{E}[C_t] \leq C_{\text{UCB}} \sqrt{KT \log T} + \sum_{t=1}^T \mathbb{E}[C_t],$$

where restricting to UCB bandit component establishes (39).  $\square$

### 12.6.3 Coupling Error from Overlapping Positions

When a position  $i$  appears in multiple subproblems (i.e.,  $i \in O_{ml}$  for some subproblem pairs  $m, l$ ), the algorithm must coordinate value assignments across these subproblems. There are two natural approaches to handle such overlapping positions:

1. **Separate solutions per subproblem:** Each subproblem maintains a local assignment on its indices; overlapping positions induce consensus constraints. Local assignments are reconciled post-hoc into a global solution by counting discrete disagreements on overlapping positions. This approach provides no coordination signal during learning.
2. **Single global solution with uncertainty penalties:** Maintain a single global solution shared across all subproblems and penalize uncertainty on overlapping positions during optimization. This enables proactive coordination through the learning process.

We **adopt the second approach**, as it allows subproblems to share learning signals on overlapping positions, improving sample efficiency and coordination during optimization. This also enables tighter regret bounds as discrete disagreement counting lacks optimization structure and yields only loose worst-case guarantees.

Thus, in our decomposition-based framework, consecutive subproblems share overlapping positions to ensure solution connectivity. By maintaining a single global solution across all subproblems, we can quantify and penalize the risk of coordination conflicts during optimization through a soft violation measure (defined in Appendix 12.8.4), rather than counting discrete disagreements after the fact.

**Overlapping Positions and Coordination Requirements.** For subproblems  $m$  and  $l$ , let  $O_{ml} = I_m \cap I_l$  denote their set of shared positions. Let  $k_i = |\{m : i \in I_m\}|$  denote the number of subproblems containing position  $i$ . Positions with  $k_i > 1$  require coordination across multiple subproblems to ensure a consistent global solution. When subproblem  $m$  assigns value  $v_m$  to position  $i \in O_{ml}$  while subproblem  $l$  assigns a different value  $v_l \neq v_m$ , a *coupling conflict* occurs. Since our global solution assigns a single value to each position, conflicts indicate potential suboptimality—at least one subproblem is not using its preferred value.

**Definition 12.8 (Coupling Error).** The coupling error at iteration  $t$  measures the total disagreement induced by overlapping subproblems:

$$H^{(t)} = \sum_{m < l} \sum_{i \in O_{ml}} \mathbb{I}[\text{subproblems } m, l \text{ disagree on } O_{ml}] = \sum_{m < l} \sum_{i \in O_{ml}} \mathbb{I}\left[x_{m,i}^{(t)} \neq x_{l,i}^{(t)}\right] \quad (40)$$

where the indicator function:  $\mathbb{I}[\text{subproblems } m, l \text{ disagree on } O_{ml}] = 1$  if there exists at least one position  $i \in O_{ml}$  such that  $x_{m,i}^{(t)} \neq x_{l,i}^{(t)}$ . Then, the cumulative coupling error accumulated over  $T$  iterations across all  $K$  subproblems is:

$$\sum_{t=1}^T C_t \equiv \sum_{t=1}^T H^{(t)} \quad (41)$$

Our objective is to bound this cumulative coupling error while simultaneously learning the optimal solution through bandit feedback.

**Regret from Coupling Conflicts** When subproblems disagree on overlapping positions, the global solution must choose one assignment over another, incurring potential regret. We now formalize this regret and derive worst-case bounds.

**Definition 12.9 (Maximum Per-Position Regret).** Let  $\rho(i, v)$  denote the reward contribution of assigning value  $v$  to position  $i$ . The maximum regret incurred by resolving a single coupling conflict is:

$$R_{\text{max-per-coupling}} = \max_{i \in [n], v_m, v_l \in [n]} |\rho(i, v_m) - \rho(i, v_l)| \quad (42)$$

This quantity characterizes the worst-case penalty for choosing one subproblem's assignment over another's at a single position.

**Proposition 12.10 (Worst-Case Coupling Error Bound).** *The coupling error satisfies the following worst-case bounds:*

$$H^{(t)} \leq \sum_{m < l} \sum_{i \in O_{ml}} \mathbb{I}[x_{m,i}^{(t)} \neq x_{l,i}^{(t)}] \quad (43)$$

$$\mathbb{E} [H^{(t)} \cdot R_{\text{max-per-coupling}}] \leq R_{\text{max-per-coupling}} \sum_{m < l} \sum_{i \in O_{ml}} \mathbb{P}[x_{m,i}^{(t)} \neq x_{l,i}^{(t)}] \quad (44)$$

*Proof.* The existence of any disagreement on overlap set  $O_{ml}$  implies at least one position-level mismatch:

$$\mathbb{I}[\text{subproblems } m, l \text{ disagree on } O_{ml}] \leq \sum_{i \in O_{ml}} \mathbb{I}[x_{m,i}^{(t)} \neq x_{l,i}^{(t)}] \quad (45)$$

Summing over all subproblem pairs  $(m, l)$  yields (43). The expected regret bound (44) follows by multiplying each position-level disagreement by  $R_{\text{max-per-coupling}}$  and taking expectations.  $\square$

**Key Insight.** Equation (44) reveals that controlling coupling error requires reducing the disagreement probability  $\mathbb{P}[x_{m,i}^{(t)} \neq x_{l,i}^{(t)}]$  for overlapping positions. This probability depends on:

1. **Overlap structure:** Large overlap sets  $O_{ml}$  and many subproblem pairs increase potential conflicts
2. **Coordination quality:** How well subproblems agree on shared positions during learning

**Lagrangian Coordination: Our Approach** A naive approach would solve each subproblem independently and resolve conflicts through post-hoc tie-breaking. However, this provides no coordination signal during learning, potentially leading to linear accumulation of coupling error over  $T$  iterations. Instead, we employ a **Lagrangian coordination mechanism** that proactively reduces coupling conflicts during optimization. See ALGORITHM 4, outlining our approach which consists of three components:

**1. Soft Violation Measure.** For position  $i$  at iteration  $t$ , we define a continuous measure that quantifies coordination risk:

$$\xi_i^{(t)} = (k_i - 1) \cdot \text{Var}(\hat{\mu}_i^{(t)}) \cdot \left( 1 - \frac{N_{i,v_i^{(t)}}}{\sum_v N_{i,v}} \right) \quad (46)$$

where:

- $k_i = |\{m : i \in I_m\}|$  is the number of subproblems containing position  $i$
- $\text{Var}(\hat{\mu}_i^{(t)}) = \frac{1}{n} \sum_v (\hat{\mu}_{i,v}^{(t)} - \bar{\mu}_i^{(t)})^2$  is the variance of value estimates
- $N_{i,v}$  is the visit count for position-value pair  $(i, v)$
- $v_i^{(t)}$  is the value assigned to position  $i$  at iteration  $t$

The soft violation  $\xi_i^{(t)}$  is high when: (a) position  $i$  appears in many subproblems ( $k_i$  large), (b) value estimates have high variance (indicating disagreement potential), and (c) the assigned value has low visit ratio (indicating uncertainty). Crucially, as learning progresses,  $\text{Var}(\hat{\mu}_i^{(t)}) \rightarrow 0$  and the visit ratio for optimal values approaches 1, ensuring  $\xi_i^{(t)} \rightarrow 0$ .

**2. Lagrangian Dual Variables.** For each position  $i$  with  $k_i > 1$ , we maintain a dual variable  $\lambda_i^{(t)} \geq 0$  that penalizes the coordination violations on overlapping positions. These dual variables are updated via *Nesterov-accelerated mirror descent* (entropic geometry) on the dual objective:

$$\lambda_i^{(t+1)} = \Pi_{[0, \lambda_{\max}]} \left[ \exp \left( \log(\lambda_i^{(t)}) + \alpha_t \xi_i^{(t)} \right) \right] \quad (47)$$

with diminishing step size  $\alpha_t = \alpha_0 / \sqrt{t}$  and Nesterov momentum for acceleration. This choice is consistent with the EXP3/FTRL-style multiplicative dynamics used by the primal learners and yields standard sublinear in  $T$  regret in the dual.

---

**Algorithm 4** DUALUPDATE: (Entropic) Mirror Descent for Lagrangian Coordination

---

**Require:** Soft violations  $\{\xi_i^{(t)}\}_{i=1}^n$ , step size  $\alpha_t = \alpha_0 / \sqrt{t}$ , dual variables  $\{\lambda_i^{(t)}\}_{i=1}^n$

```

1: for each position  $i$  with  $k_i > 1$  do
2:    $\lambda_i^{(t+1)} \leftarrow \Pi_{[0, \lambda_{\max}]} \left( \lambda_i^{(t)} \cdot \exp(\alpha_t \xi_i^{(t)}) \right)$  ▷ Update dual variables
3: end for
4: return updated dual variables  $\{\lambda_i^{(t+1)}\}$ 

```

---

The soft violation  $\xi_i^{(t)}$  aggregates disagreement risk across all overlapping subproblems containing position  $i$ , eliminating the need for pairwise dual variables. The multiplicative update preserves nonnegativity of the dual variables and naturally adapts the coordination strength to the uncertainty level of each overlapping position.

**3. Score Modification.** The dual variables modify expert scores during subproblem optimization. For position  $i$  with value  $v$ , the Lagrangian-modified score is:

$$\tilde{s}_{i,v}^{(t)} = s_{i,v}^{(t)} - (k_i - 1)\lambda_i^{(t)} \quad (48)$$

where  $s_{i,v}^{(t)}$  is the base score (from UCB or Hedge) and the penalty term  $(k_i - 1)\lambda_i^{(t)}$  discourages high-risk assignments on overlapping positions. Although the penalty is value-independent for a fixed position  $i$ , it affects future learning dynamics by uniformly discouraging high-uncertainty assignments on heavily overlapping positions, thereby accelerating variance reduction and coordination across subproblems rather than enforcing hard consensus at each step.

*A possible extension would be to introduce value-dependent coordination penalties that explicitly discourage disagreement with a consensus assignment (e.g., penalizing  $v \neq z_i^{(t)}$ ). We do not pursue this here, as preliminary experiments suggest limited impact on the overall learning dynamics relative to the simpler uncertainty-based penalty.*

**Coordination Mechanism.** This three-component approach creates a feedback loop:

1. Positions with high variance or low confidence accumulate large soft violations  $\xi_i^{(t)}$
2. Large violations cause dual variables  $\lambda_i^{(t)}$  to grow via mirror descent

3. Large dual variables penalize uncertain choices more heavily via score modification
4. Penalties encourage exploration until confident, coordinated values emerge
5. As confidence increases,  $\xi_i^{(t)} \rightarrow 0$  and penalties naturally diminish

The detailed Lagrangian formulation, mirror descent update rules, and convergence analysis are provided in Section 12.8.4. Using this mechanism, we prove:

**Theorem 12.11** (Coupling Error Bound). *Under the Lagrangian coordination mechanism under accelerated mirror descent on the dual variables, the expected cumulative coupling error satisfies:*

$$\mathbb{E} \left[ \sum_{t=1}^T C_t \right] = O(K^2 Q R_{\max} \sqrt{T}) \quad (49)$$

where  $K$  is the number of subproblems,  $Q = \max_{m,l} |O_{ml}|$  is the maximum overlap size, and  $R_{\max} = \max_{i,v_m,v_l} |\rho(i, v_m) - \rho(i, v_l)|$  is the maximum per-position regret.

*Proof.* We prove the bound in three parts.

- (1) **Soft Violations are Bounded.** By construction Equation 46, each component of  $\xi_i^{(t)}$  is bounded -  $(k_i - 1) \leq K - 1$  (overlap count), for rewards in  $[0, 1]$  the  $\text{Var}(\hat{\mu}_i^{(t)}) \leq \frac{1}{4}$ ,  $(1 - \text{visit ratio}) \in [0, 1]$ . Therefore  $\|\xi_t\|_\infty \leq Q$  where  $Q = O(K)$ .
- (2) **Cumulative Soft Violations are Bounded.** By Lemma 12.33, the cumulative dual gap satisfies:

$$\sum_{t=1}^T (g(\lambda^*) - g(\lambda_t)) \leq O(d \lambda_{\max} Q \sqrt{T}) \quad (50)$$

In Lagrangian duality, the soft violations  $\xi_t$  are sub-gradients of the dual function  $g$ . A standard result in online convex optimization Shalev-Shwartz (2012) states that bounded cumulative dual gap implies bounded cumulative sub-gradient norms weighted by step sizes:

$$\sum_{t=1}^T \alpha_t \langle \xi_t, \lambda^* - \lambda_t \rangle \leq O(\sqrt{T}) \quad (51)$$

Since dual variables are bounded in  $[0, \lambda_{\max}]^d$ , this implies:

$$\sum_{t=1}^T \|\xi_t\|_1 \leq O(d \lambda_{\max} \sqrt{T}) \quad (52)$$

- (3) **Coupling Error Bounded by Soft Violations.** The coupling error at iteration  $t$  counts disagreements weighted by  $R_{\max}$ . From Proposition 12.10:

$$C_t \leq R_{\max} \sum_{m < l} \sum_{i \in O_{ml}} \mathbb{I}[x_{m,i}^{(t)} \neq x_{l,i}^{(t)}] \quad (53)$$

To bound the disagreement indicators, we use the key observation that disagreement between subproblems requires uncertainty in value estimates: when all subproblems have identical, confident estimates for a position, they deterministically select the same value. We formalize this by showing that the **expected**

disagreement probability is bounded by the variance of value estimates. Applying Lemma 12.34 and noting that the soft violation satisfies  $\xi_i^{(t)} \geq (k_i - 1) \cdot \sigma_i^2(t) \cdot c_1$  for some constant  $c_1 > 0$ :

$$\mathbb{E}[C_t] \leq R_{\max} \sum_{m < l} \sum_{i \in O_{ml}} c_0 \cdot \sigma_i^2(t) \quad (54)$$

$$\leq \frac{c_0 R_{\max}}{c_1} \sum_{i: k_i > 1} \frac{\xi_i^{(t)}}{k_i - 1} \quad (55)$$

$$\leq \frac{c_0 R_{\max}}{c_1} \cdot \|\xi_t\|_1 \quad (56)$$

where the last inequality uses  $k_i - 1 \geq 1$  for overlapping positions. Summing over all subproblem pairs contributes at most a factor of  $\binom{K}{2} \leq K^2$ , giving:

$$\mathbb{E}[C_t] \leq C' \cdot K^2 \cdot R_{\max} \cdot \|\xi_t\|_1 \quad (57)$$

for constant  $C' = c_0/c_1$ .

Now, bringing everything together. From part (3), we have:

$$\mathbb{E}[C_t] \leq C' \cdot K^2 \cdot R_{\max} \cdot \|\xi_t\|_1$$

The final bound is obtained by summing over  $T$  iterations:

$$\sum_{t=1}^T \mathbb{E}[C_t] \leq C' \cdot R_{\max} \cdot K^2 \sum_{t=1}^T \|\xi_t\|_1 \quad (58)$$

$$\leq C' \cdot R_{\max} \cdot K^2 \cdot O(d \lambda_{\max} \sqrt{T}) \quad (59)$$

$$= O(K^2 Q R_{\max} \sqrt{T}) \quad (60)$$

where we absorb  $d$ ,  $\lambda_{\max}$ , and  $C'$  into the constant, and use  $d \leq K \cdot Q$  (number of overlapping positions).  $\square$

#### 12.6.4 Exp3 Regret

**Theorem 12.12 (Standard Exp3 Regret with Importance Sampling).** Consider a combinatorial optimization problem over domain of size  $n$ , decomposed into  $K$  subproblems, each of size  $d$  with overlap  $|\mathcal{O}|$ . Let  $w_{i,j}(t) \in \mathbb{R}_+$  denote the Exp3 weight for assigning value  $j \in [d]$  to position  $i \in [n]$  at iteration  $t$ . The weights for each position  $i$  are  $\mathbf{w}_i(t) = [w_{i,1}(t), w_{i,2}(t), \dots, w_{i,d}(t)] \in \mathbb{R}_+^d$ . At each round  $t$ , the algorithm samples a solution  $\mathbf{x}_t$  coordinate-wise, where each position  $i$  selects value  $j$  with probability  $p_{i,j}(t) = \frac{w_{i,j}(t)}{\sum_{j'} w_{i,j'}(t)}$ . The algorithm observes only the total reward  $\rho_t = f(\mathbf{x}_t)$  for the selected solution. Weights are updated using importance-weighted rewards:

$$w_{i,j}(t+1) = w_{i,j}(t) \cdot \exp \left( \eta \cdot \frac{\rho_t \cdot \mathbb{I}[x_t(i) = j]}{p_{i,j}(t)} \right)$$

with learning rate  $\eta$ . If the objective function is  $L$ -Lipschitz continuous (Assumption 2), then the total Exp3 regret over  $T$  iterations is bounded by:

$$R_{\text{Exp3}}(T) \leq \frac{n \log d}{\eta} + \frac{\eta T n B^2}{2} + L |\mathcal{O}| \sqrt{T} \quad (61)$$

where  $B$  is the maximum reward magnitude. Setting  $\eta = \sqrt{\frac{2 \log d}{TB^2}}$  yields:

$$R_{\text{Exp3}}(T) = O\left(\sqrt{ndT \log d} + L|\mathcal{O}| \sqrt{T}\right) = O\left(\sqrt{KdT \log d} + L|\mathcal{O}| \sqrt{T}\right) \quad (62)$$

where the last equality uses  $n = Kd$  for the decomposition structure.

*Proof.* Following similar steps as in UCB Regret bound, using instantaneous regret as in Definition 12.5, we decompose:

$$r_t = f(\mathbf{x}^*) - \underbrace{\sum_{k=1}^K g_k(\mathbf{x}_{I_k}^*)}_{\text{optimal gap} \leq 0} + \underbrace{\sum_{k=1}^K [g_k(\mathbf{x}_{I_k}^*) - g_k(\mathbf{x}_{I_{k,t}})]}_{\text{Exp3 subproblem regret}} + \underbrace{C_t}_{\text{coupling error}} \quad (63)$$

For the Exp3 subproblem regret, we apply standard Exp3 analysis to each position  $i$ . The importance-weighted estimator  $\hat{\rho}_{i,j}(t) = \frac{\rho_t \cdot \mathbb{I}[x_t(i)=j]}{p_{i,j}(t)}$  is unbiased:  $\mathbb{E}[\hat{\rho}_{i,j}(t) | \mathcal{F}_{t-1}] = \rho_t$ . By the standard Exp3 regret bound (Auer et al. (2002b), also see Cesa-Bianchi and Lugosi (2006)), for each position  $i$ :

$$\sum_{t=1}^T \left[ \max_j \mathbb{E}[\rho_t(j)] - \mathbb{E}[\rho_t(x_t(i))] \right] \leq \frac{\log d}{\eta} + \frac{\eta T B^2}{2}$$

Summing over all  $n$  positions gives the subproblem regret bound:

$$\sum_{k=1}^K \sum_{t=1}^T [g_k(\mathbf{x}_{I_k}^*) - g_k(\mathbf{x}_{I_{k,t}})] \leq \frac{n \log d}{\eta} + \frac{\eta T n B^2}{2}$$

For the coupling error, by Assumption 3 (Bounded Coupling), the coordination violations satisfy  $|C_t| \leq L|\mathcal{O}|$  per iteration. By concentration inequalities, the cumulative coupling error satisfies:

$$\sum_{t=1}^T C_t = O(L|\mathcal{O}| \sqrt{T})$$

Combining these bounds yields:

$$R_{\text{Exp3}}(T) \leq \frac{n \log d}{\eta} + \frac{\eta T n B^2}{2} + L|\mathcal{O}| \sqrt{T}$$

Setting  $\eta = \sqrt{\frac{2 \log d}{TB^2}}$  and using  $n = Kd$  completes the proof.  $\square$

---

**Algorithm 5** UPDATEEXPERTS: Clipped Importance-Weighted Multi-Expert Parameter Update

---

**Require:** Solution  $x^{(t)}$ , reward  $r^{(t)}$ , parameters  $(\hat{V}, N, W, \tilde{L})$ , **Hyperparameters:** clipping threshold  $p_{\min}$ , learning rate  $\eta$

**Output:** Updated  $(\hat{V}, N, W, \tilde{L})$

**Initialize:**  $\tilde{r}^{(t)} \leftarrow \text{NORMALIZE}(r^{(t)})$  ▷ Normalization to  $[-1, 1]$

**Initialize:**  $\ell^{(t)} \leftarrow 1 - \tilde{r}^{(t)}$  ▷ Instantaneous loss (regret surrogate)

1: **for**  $i = 1, \dots, n$  **do**

2:    $a \leftarrow x_i^{(t)}$  ▷ Action selected at position  $i$

3:    $N_{i,a} \leftarrow N_{i,a} + 1$  ▷ Increment visit count

4:    $\tilde{p}_a^{(t)} \leftarrow \max(W_{i,a} \| W_i \|_1, p_{\min})$  ▷ Clipped selection probability

5:    $\hat{r}^{(t)} \leftarrow \tilde{r}^{(t)} / \tilde{p}_a^{(t)}$  ▷ Importance-weighted reward

6:    $\hat{\ell}^{(t)} \leftarrow \ell^{(t)} / \tilde{p}_a^{(t)}$  ▷ Importance-weighted loss

7:    $\hat{V}_{i,a} \leftarrow \hat{V}_{i,a} + \frac{1}{N_{i,a}} (\hat{r}^{(t)} - \hat{V}_{i,a})$  ▷ Expert 1 (UCB): running average

8:    $W_{i,a} \leftarrow W_{i,a} \cdot \exp(\eta \hat{r}^{(t)} / n)$  ▷ Expert 2 (EXP3): multiplicative weights

9:    $\tilde{L}_{i,a} \leftarrow \tilde{L}_{i,a} + \hat{\ell}^{(t)}$  ▷ Expert 3 (FTRL): cumulative regret

10: **end for**

11:  $W_i \leftarrow W_i / \| W_i \|_1$  for all  $i$  ▷ Normalize weights

12: **return**  $(\hat{V}, N, W, \tilde{L})$

---

**Lemma 12.13 (Bias from Clipped Importance Sampling).** Consider the EXP3 algorithm for a combinatorial bandit problem with  $n$  positions and  $d$  values per position where at each round  $t$  the learner samples an action  $x_t = (x_t(1), \dots, x_t(n))$  from a factorized distribution  $\Pr(x_t \mid \mathcal{F}_{t-1}) = \prod_{i=1}^n p_{i,x_t(i)}(t)$ . Let  $p_{i,j}(t)$  denote the probability of selecting value  $j$  at position  $i$  at a time  $t$ , with weights updated via  $w_{i,j}(t+1) = w_{i,j}(t) \cdot \exp(\eta \cdot \hat{r}_{i,j}(t))$  as:

$$p_{i,j}(t) = \frac{w_{i,j}(t)}{\sum_{j'=1}^d w_{i,j'}(t)}$$

where  $\eta > 0$  is the learning rate. Thus, each coordinate  $x_t(i)$  conditioned on the past history  $\mathcal{F}_{t-1}$  is drawn independently using its own probability vector  $\mathbf{p}_i(t)$ . The EXP3-style update for each position  $i$  uses the clipped importance-weighted estimator:

$$\hat{r}_{i,j}(t) = \begin{cases} \frac{\rho_t}{\max(p_{i,j}(t), C)} & \text{if } x_t(i) = j \\ 0 & \text{otherwise} \end{cases}$$

where clipping threshold  $C > 0$  is a fixed constant (e.g.,  $C=0.01$ ),  $\rho_t \in [-B, B]$  is the observed scalar reward, and  $x_t(i)$  is the value selected at position  $i$  in round  $t$ . Let  $\tilde{r}_{i,j}(t)$  denote the corresponding un-clipped estimator

$$\tilde{r}_{i,j}(t) = \begin{cases} \frac{\rho_t}{p_{i,j}(t)} & \text{if } x_t(i) = j \\ 0 & \text{otherwise} \end{cases}$$

Then the **total bias introduced by clipping** over  $T$  rounds satisfies:

$$\sum_{t=1}^T \sum_{i=1}^n |\mathbb{E}[\hat{r}_{i,x_t(i)}(t)] - \rho_t| \leq \frac{2nB\log(d)}{\eta} \tag{64}$$

*Proof.* Since the algorithm factorizes over positions, it suffices to prove the bound for a fixed position  $i$ , and then multiply by  $n$ . So let's fix  $i$  and omit the index  $i$  for brevity in  $w_j(t)$ ,  $p_j(t)$ ,  $\hat{r}_j(t)$ . The un-clipped estimator

$\tilde{r}_j(t)$  as defined above is an unbiased estimator of  $\rho_t$  as,

$$\mathbb{E}[\tilde{r}_{x_t}(t) \mid \mathcal{F}_{t-1}] = \mathbb{E}[\rho_t \mid \mathcal{F}_{t-1}],$$

If one uses a clipped estimator  $\hat{r}_j(t)$ , we want to bound the bias over  $T$  rounds for this position  $i$ :

$$\text{Bias}_{i,j} = \sum_{t=1}^T |\mathbb{E}[\hat{r}_{x_t}(t)] - \mathbb{E}[\rho_t]| \quad (65)$$

i.e the bias that emerges due to the approximate importance sampling. The per-round bias for this position:

$$\text{Bias}_{i,j}(t) := E[\hat{r}_{x_t}(t)] - \mathbb{E}[\rho_t] = \mathbb{E}[\hat{r}_{x_t}(t)] - \mathbb{E}[\tilde{r}_{x_t}(t)]$$

Now, if  $p_j(t) \geq C$ , then  $\hat{r}_j(t) = \tilde{r}_j(t)$ . If  $p_j(t) < C$ , then

$$\hat{r}_j(t) = \frac{\rho_t}{C}, \quad \tilde{r}_j(t) = \frac{\rho_t}{p_j(t)},$$

and since  $p_j(t) < C$ , we have  $|\hat{r}_j(t)| \leq |\tilde{r}_j(t)|$  with the same sign, so  $\tilde{r}_j(t) - \hat{r}_j(t) \geq 0$  for all  $j, t$ . This introduces a negative bias or underestimation of the reward. As  $|\text{Bias}_{i,j}(t)| = -\text{Bias}_{i,j}(t)$ ,

$$\sum_{t=1}^T |\mathbb{E}[\hat{r}_{x_t}(t)] - \mathbb{E}[\rho_t]| = \sum_{t=1}^T \mathbb{E}[\tilde{r}_{x_t}(t) - \hat{r}_{x_t}(t)] = \sum_{j=1}^d p_j(t)(\tilde{r}_j(t) - \hat{r}_j(t)). \quad (66)$$

Therefore, for this position,

$$\sum_{t=1}^T \left| \mathbb{E}[\hat{r}_{x_t}(t)] - \mathbb{E}[\rho_t] \right| = \sum_{t=1}^T \sum_{j=1}^d p_j(t)(\tilde{r}_j(t) - \hat{r}_j(t)). \quad (67)$$

Now, define potential function  $W(t) = \sum_{j=1}^d w_j(t)$ ,  $p_j(t) = \frac{w_j(t)}{W(t)}$

Update iterates for un-clipped estimators  $\tilde{r}_j(t)$  and clipped estimators  $\hat{r}_j(t)$ ,

$$\begin{aligned} w_j^{un}(t+1) &= w_j^{un}(t) \exp(\eta \tilde{r}_j(t)) \\ w_j^{cl}(t+1) &= w_j^{cl}(t) \exp(\eta \hat{r}_j(t)) \end{aligned}$$

Note, wrt **coupling argument** – we compare both weight sequences under the *same* realization of sampled actions  $\{x_t\}_{t=1}^T$ . That is, we run the actual algorithm (which uses clipped estimators and clipped probabilities  $p_j^{cl}(t)$ ) and track two weight sequences,  $w_j^{cl}(t)$ : the actual weights, updated with clipped estimators,  $w_j^{un}(t)$ : hypothetical weights, updated as if we used unclipped estimators for the same action sequence. Since the action  $x_t$  is the same for both, we have:

- If  $x_t = j$ : both  $\tilde{r}_j(t)$  and  $\hat{r}_j(t)$  are nonzero, with  $\tilde{r}_j(t) \geq \hat{r}_j(t)$
- If  $x_t \neq j$ : both  $\tilde{r}_j(t) = \hat{r}_j(t) = 0$

Therefore,  $w_j^{un}(t) \geq w_j^{cl}(t)$  for all  $j, t$ , which implies  $\Phi^{un}(t) \geq \Phi^{cl}(t)$ .

Now, consider log potentials  $\Phi$  as,

$$\Phi^{un}(t) = \log \sum_{j=1}^d w_j^{un}(t), \quad \Phi^{cl}(t) = \log \sum_{j=1}^d w_j^{cl}(t)$$

Then,

$$W(t+1) = \sum_{j=1}^d w_j(t+1) = \sum_{j=1}^d w_j^{un}(t) \exp(\eta \tilde{r}_j(t)) = W(t) \sum_{j=1}^d p_j(t) \exp(\eta \tilde{r}_j(t))$$

$$\Phi(t+1) - \Phi(t) = \log\left(\frac{W(t) \sum_{j=1}^d p_j(t) \exp(\eta \tilde{r}_j(t))}{W(t)}\right) = \log\left(\sum_{j=1}^d p_j(t) \exp(\eta \tilde{r}_j(t))\right)$$

For now let's assume that rewards are scaled to  $[-1, 1]$  (will multiply with  $B$  finally). Then  $|\tilde{r}_j(t)|, |\hat{r}_j(t)| \leq 1/C$ . With this we can use a standard Hoeffding-type inequality for exponential weights (e.g., Theorem 3.1 [Bubeck and Cesa-Bianchi \(2012\)](#) & Lemma 2.2 in [Cesa-Bianchi and Lugosi \(2006\)](#)) that gives, for any bounded  $z_j(t)$ ,

$$\Phi(t+1) - \Phi(t) \leq \eta \sum_{j=1}^d p_j(t) z_j(t) + \frac{\eta^2}{2}$$

whenever  $|z_j(t)| \leq 1$ ; After scaling, this becomes the same inequality up to a constant absorbed into  $\eta^2$ . Applying this once with  $z_j(t) = \tilde{r}_j(t)$  and once with  $z_j(t) = \hat{r}_j(t)$  and subtracting, we obtain

$$\Phi^{un}(T+1) - \Phi^{cl}(T+1) \leq \eta \sum_{t=1}^T \sum_{j=1}^d p_j(t) (\tilde{r}_j(t) - \hat{r}_j(t)) + O(\eta^2 T).$$

As both schemes start from identical initial weights  $w_j^{un}(1) = w_j^{cl}(1) = 1$ , so  $\Phi^{un}(1) = \Phi^{cl}(1) = \log d$ . Recall that  $\tilde{r}_j(t) \geq \hat{r}_j(t) \quad \forall j, t$

Thus, the un-clipped weights grow at least as fast as the clipped ones  $w_j^{un}(t+1) \geq w_j^{cl}(t+1)$  and if both schemes start from equal initial weights then for all  $t$ , we have  $\max_j w_j^{cl}(t) \leq \max_j w_j^{un}(t)$  and because  $\sum_{j=1}^d w_j(t) \leq d \cdot \max_j w_j(t)$  we get,

$$\max_j w_j^{cl}(t) \leq W^{cl}(t) \quad \text{and} \quad W^{un}(t) \leq d \max_j w_j^{un}(t),$$

taking log both sides and as this applies for every time step including  $T+1$  we get,

$$\Phi^{un}(T+1) - \Phi^{cl}(T+1) \leq \log d$$

Choosing  $\eta = O(1/\sqrt{T})$  makes the  $O(\eta^2 T)$  term constant-sized so we get,

$$\eta \sum_{t=1}^T \sum_{j=1}^d p_j(t) (\tilde{r}_j(t) - \hat{r}_j(t)) \leq \log d$$

In the normalized  $[-1, 1]$  case,

$$\sum_{t=1}^T \sum_{j=1}^d p_j(t) (\tilde{r}_j(t) - \hat{r}_j(t)) \leq \frac{\log d}{\eta}$$

Note, if the actual rewards satisfy  $\rho_t \in [-B, B]$ , then both  $\tilde{r}_j(t)$  and  $\hat{r}_j(t)$  scale linearly in  $\rho_t$ , so the difference  $\tilde{r}_j(t) - \hat{r}_j(t)$  scales by at most a factor  $B$ . Taking a slightly looser constant to account for both tails, we obtain

$$\sum_{t=1}^T \sum_{j=1}^d p_j(t) (\tilde{r}_j(t) - \hat{r}_j(t)) \leq \frac{2B \log d}{\eta}$$

Combining this with (67) yields, for this position  $i$ ,

$$\sum_{t=1}^T \left| \mathbb{E}[\hat{r}_{x_t}(t)] - \mathbb{E}[\rho_t] \right| \leq \frac{2B \log d}{\eta}$$

Finally, summing over  $i = 1, \dots, n$  positions gives

$$\sum_{t=1}^T \sum_{i=1}^n \left| \mathbb{E}[\hat{r}_{i,x_t(i)}(t)] - \mathbb{E}[\rho_t] \right| \leq \frac{2nB \log d}{\eta}$$

□

**Theorem 12.14 (Exp3 Subproblem Regret with Clipped Importance Sampling).** Consider a combinatorial optimization problem over domain of size  $n$ , decomposed into  $K$  subproblems, each of size  $d$  with overlap  $|\mathcal{O}|$ . Let  $w_{i,j}(t) \in \mathbb{R}_+$  denote the Exp3 weight for assigning value  $j \in [d]$  to position  $i \in [n]$  at iteration  $t$ . At each round  $t$ , the ALGORITHM 5 samples a solution  $\mathbf{x}_t$  coordinate-wise, where each position  $i$  selects value  $j$  with probability  $p_{i,j}(t) = \frac{w_{i,j}(t)}{\sum_{j'} w_{i,j'}(t)}$ .

The ALGORITHM 5 observes only the total reward  $\rho_t = f(\mathbf{x}_t)$  for the selected solution. Weights are updated using **clipped** importance-weighted rewards:

$$w_{i,j}(t+1) = w_{i,j}(t) \cdot \exp(\eta \cdot \hat{r}_{i,j}(t))$$

where

$$\hat{r}_{i,j}(t) = \begin{cases} \frac{\rho_t}{\max(p_{i,j}(t), C)} & \text{if } x_t(i) = j \\ 0 & \text{otherwise} \end{cases}$$

with clipping threshold  $C > 0$  (e.g.,  $C = 0.01$ ), learning rate  $\eta$ , and  $\rho_t \in [-B, B]$ .

If the objective function is  $L$ -Lipschitz continuous (Assumption 2), then:

(a) **Per-subproblem bound:** For each subproblem  $k \in [K]$  of size  $d$ , the Exp3 regret over  $T$  iterations satisfies:

$$\sum_{t=1}^T [g_k(\mathbf{x}_{I_k}^*) - g_k(\mathbf{x}_{I_{k,t}})] \leq \frac{d \log d}{\eta} + \frac{\eta T dB^2}{2C^2} + \frac{2dB \log(d)}{\eta} \quad (68)$$

With optimal learning rate  $\eta = \sqrt{\frac{2C^2 \log d}{TB^2}}$ , this simplifies to:

$$\sum_{t=1}^T [g_k(\mathbf{x}_{I_k}^*) - g_k(\mathbf{x}_{I_{k,t}})] = O\left(\frac{dB \sqrt{T \log d}}{C}\right) \quad (69)$$

(b) **Total bound across subproblems:** Summing over all  $K$  subproblems and including coupling errors, the total Exp3 regret over  $T$  iterations is:

$$R_{\text{Exp3}}(T) \leq \frac{n \log d}{\eta} + \frac{\eta T n B^2}{2C^2} + \frac{2nB \log(d)}{\eta} + L|\mathcal{O}| \sqrt{T} \quad (70)$$

Setting  $\eta = \sqrt{\frac{2C^2 \log d}{TB^2}}$  yields:

$$R_{\text{Exp3}}(T) = O\left(\frac{nB \sqrt{T \log d}}{C} + L|\mathcal{O}| \sqrt{T}\right) = O\left(\frac{KdB \sqrt{T \log d}}{C} + L|\mathcal{O}| \sqrt{T}\right) \quad (71)$$

where the last equality uses  $n = Kd$  for the decomposition structure.

(c) **Total Average regret across subproblems:** Equivalently, in terms of the average subproblem regret  $R_{\text{subproblem}}^{\text{avg}}(T) := \frac{1}{K} \sum_{k=1}^K R_k(T)$ , we obtain,

$$R_{\text{subproblem}}^{\text{avg}}(T) = \frac{1}{K} R_{\text{Exp3}}(T) = O\left(\frac{dB\sqrt{T \log d}}{C} + \frac{L|\mathcal{O}|}{K}\sqrt{T}\right)$$

The leading EXP3 term scales as  $O(d\sqrt{T \log d})$  & is independent of number of subproblems  $K$ .

*Proof.* Following the same instantaneous regret decomposition as in Theorem 12.12:

$$r_t = f(\mathbf{x}^*) - \underbrace{\sum_{k=1}^K g_k(\mathbf{x}_{I_k}^*)}_{\text{optimal gap} \leq 0} + \underbrace{\sum_{k=1}^K [g_k(\mathbf{x}_{I_k}^*) - g_k(\mathbf{x}_{I_{k,t}})]}_{\text{Exp3 subproblem regret}} + \underbrace{C_t}_{\text{coupling error}} \quad (72)$$

We prove part (a) first, then aggregate to obtain part (b).

Let's consider a single subproblem  $k$  of size  $d$ . Note, the key difference from standard Exp3 is that the importance-weighted estimator is now **biased** due to clipping. Therefore, let's decompose the subproblem regret into two components - 1) Standard Exp3 exploration-exploitation tradeoff and 2) Bias from clipping. If the unbiased estimator is defined as:

$$\tilde{r}_{i,j}(t) = \begin{cases} \frac{\rho_t}{p_{i,j}(t)} & \text{if } x_t(i) = j \\ 0 & \text{otherwise} \end{cases}$$

By standard Exp3 analysis (Auer et al., 2002), if we were using  $\tilde{r}_{i,j}(t)$ , the regret for each position  $i \in I_k$  would be:

$$\sum_{t=1}^T \left[ \max_j \mathbb{E}[\rho_t(j)] - \mathbb{E}[\rho_t(x_t(i))] \right] \leq \frac{\log d}{\eta} + \frac{\eta T B^2}{2C^2}$$

where the variance term increases by  $1/C^2$  because:

$$\text{Var}[\hat{r}_{i,j}(t)] \leq \frac{B^2}{C^2}$$

Summing over all  $d$  positions in subproblem  $k$ :

$$\sum_{t=1}^T [g_k(\mathbf{x}_{I_k}^*) - g_k(\mathbf{x}_{I_{k,t}})] \leq \frac{d \log d}{\eta} + \frac{\eta T d B^2}{2C^2}$$

However, we are actually using  $\hat{r}_{i,j}(t)$  instead of  $\tilde{r}_{i,j}(t)$ , which introduces bias. By Lemma 12.13, for the  $d$  positions in subproblem  $k$ , the total bias over all rounds is:

$$\sum_{t=1}^T \sum_{i \in I_k} |\mathbb{E}[\hat{r}_{i,x_t(i)}(t)] - \rho_t| \leq \frac{2dB \log(d)}{\eta}$$

*Combining both components:*

The total regret for subproblem  $k$  is:

$$\sum_{t=1}^T [g_k(\mathbf{x}_{I_k}^*) - g_k(\mathbf{x}_{I_{k,t}})] \leq \frac{d \log d}{\eta} + \frac{\eta T d B^2}{2C^2} + \frac{2dB \log(d)}{\eta} = \frac{(1+2B)d \log d}{\eta} + \frac{\eta T d B^2}{2C^2} \quad (73)$$

Setting  $\eta = \sqrt{\frac{2C^2(1+2B)\log d}{TB^2}}$  gives:

$$\sum_{t=1}^T [g_k(\mathbf{x}_{I_k}^*) - g_k(\mathbf{x}_{I_{k,t}})] = \frac{(1+2B)d\log d}{\sqrt{\frac{2C^2(1+2B)\log d}{TB^2}}} + \frac{\sqrt{\frac{2C^2(1+2B)\log d}{TB^2}} \cdot TdB^2}{2C^2} \quad (74)$$

$$= d\sqrt{\frac{(1+2B)TB^2\log d}{2C^2}} + d\sqrt{\frac{(1+2B)TB^2\log d}{2C^2}} \quad (75)$$

$$= 2d\sqrt{\frac{(1+2B)TB^2\log d}{2C^2}} = O\left(\frac{dB\sqrt{T\log d}}{C}\right) \quad (76)$$

This proves part (a).

### Part (b): Total bound across subproblems.

Summing the per-subproblem bounds over all  $K$  subproblems:

$$\sum_{k=1}^K \sum_{t=1}^T [g_k(\mathbf{x}_{I_k}^*) - g_k(\mathbf{x}_{I_{k,t}})] \leq K \cdot \left( \frac{(1+2B)d\log d}{\eta} + \frac{\eta TdB^2}{2C^2} \right) = \frac{(1+2B)n\log d}{\eta} + \frac{\eta TnB^2}{2C^2}$$

where we used  $n = Kd$  (total number of positions equals  $K$  subproblems times  $d$  positions per subproblem).

Adding the coupling error term: By Assumption 3 (Bounded Coupling), the coordination violations satisfy  $|C_t| \leq L|\mathcal{O}|$  per iteration. By concentration inequalities:

$$\sum_{t=1}^T C_t = O(L|\mathcal{O}|\sqrt{T}) \quad (\text{where } |\mathcal{O}| \leq KQ)$$

Combining all terms:

$$R_{\text{Exp3}}(T) \leq \underbrace{\frac{n\log d}{\eta} + \frac{\eta TnB^2}{2C^2} + \frac{2nB\log(d)}{\eta}}_{\text{subproblem regret}} + \underbrace{L|\mathcal{O}|\sqrt{T}}_{\text{coupling error}} \quad (77)$$

Setting  $\eta = \sqrt{\frac{2C^2\log d}{TB^2}}$  to balance the first two terms:

$$R_{\text{Exp3}}(T) = \frac{(1+2B)n\log d}{\sqrt{\frac{2C^2\log d}{TB^2}}} + \frac{\sqrt{\frac{2C^2\log d}{TB^2}} \cdot TnB^2}{2C^2} + L|\mathcal{O}|\sqrt{T} \quad (78)$$

$$= n\sqrt{\frac{TB^2(1+2B)\log d}{2C^2}} + n\sqrt{\frac{TB^2\log d}{2C^2}} + L|\mathcal{O}|\sqrt{T} \quad (79)$$

$$= (\sqrt{(1+2B)} + 1) \cdot n\sqrt{\frac{TB^2\log d}{2C^2}} + L|\mathcal{O}|\sqrt{T} \quad (80)$$

$$= O\left(\frac{nB\sqrt{T\log d}}{C} + L|\mathcal{O}|\sqrt{T}\right) \quad (81)$$

Using  $n = Kd$ :

$$R_{\text{Exp3}}(T) = O\left(\frac{KdB\sqrt{T\log d}}{C} + L|\mathcal{O}|\sqrt{T}\right)$$

This completes the proof of part (b).  $\square$

### Remarks

- Impact of clipping constant  $C$ :** The regret scales as  $1/C$ , so smaller clipping thresholds lead to higher regret. However, in practice,  $C = 0.01$  provides a good balance between variance reduction and bias.
- Comparison to unclipped Exp3:** If  $C = O(1/\sqrt{T})$  (diminishing clipping), the bias term becomes  $O(n\sqrt{T \log d})$ , matching the standard Exp3 bound. With fixed  $C$ , we pay an extra factor of  $1/C$  but gain significant practical stability.
- Bias vs. variance tradeoff:** The clipping introduces bias of order  $O(nB \log d/\eta)$  but reduces variance from  $O(B^2/p_{\min}^2)$  to  $O(B^2/C^2)$  where  $C$  is constant. The bias is absorbed into the leading  $O(\sqrt{T})$  term.
- Key insight:** The bias diminishes **not because the threshold decreases**, but because Exp3 concentrates probability mass on good actions over time, making clipping events increasingly rare.

### 12.6.5 FTRL Regret

Follow-the-Regularized-Leader is a classic online learning algorithm using convex losses so naturally, this does not align with our setting. Combinatorial (ex: TSP) or categorical multiobjective problems especially in our case use non-convex losses.

So we use a clever trick - linearization via Probablilty simplex! Since we have a bandit (multi-arm) setting we lift it to continuous space by we choose a position/action by sampling from a distribution of as  $p_i \in \Delta_d = \{p \in \mathbb{R}^n : p_j \geq 0, \sum_j p_j = 1\}$  where  $p_{i,j} = Pr(\text{assign city } j \text{ to position } i)$ . Mathematically,

Discrete problem :  $x_i \in \{0, 1, \dots, n-1\}$  choose 1 city for position i

Lifted continuous problem :  $p_i \in \Delta_d = \{p \in \mathbb{R}^n : p_j \geq 0, \sum_j p_j = 1\}$

where  $p_{i,j} = Pr(\text{assign city } j \text{ to position } i)$ .

Therefore, based on which position/action is selected we assign it a reward and this results in expected loss to be convex!

Mathematically, if we define any position-wise loss for any subproblem k based on  $(i, j)$  as  $\ell_t^{(k)}(i, j)$

$$\ell_t^{(k)}(i, j) = \text{marginal loss contribution when position } i \text{ has city } j$$

Where marginal loss means the contribution of placing city  $j$  at position  $i$  is the change in total tour cost caused by that assignment. Then,

$$\ell_t^{(k)}(i, j) = 1 - r_t^{(k)}(i, j)$$

where  $r_t(i, j) = \frac{\text{Reward(solution with city } j \text{ at position } i) - R_{\min}}{R_{\max} - R_{\min}}$

This captures the cost contribution of assignment  $(i, j)$  to the full tour.

So now how do we connect any subproblem's objective to position-wise losses? It is quite straightforward, since we define  $\ell_t^{(k)}(i, j)$  as the normalized marginal loss, we're measuring the incremental effect of each assignment, not assuming positions are independent. Therefore, by our definition,

$$g_k(x_{I_k}^*) - g_k(x_{I_{k,t}}) = \sum_{i \in I_k} [r_t^{(k)}(i, x_{I_k}^*) - r_t^{(k)}(i, x_{I_{k,t}})] = \sum_{i \in I_k} [\ell_t^{(k)}(i, x_{i,t}) - \ell_t^{(k)}(i, x_i^*)]$$

Additionally, note since we define  $\ell_t^{(k)}(i, j) = 1 - r_t^{(k)}(i, j)$  - reward, then expected loss under distribution  $p_i$  is linear in  $p_i$ ,

$$\text{If any loss function } \ell_t^{(k)}(i, j) := (1 - r_t^{(k)}(i, j))$$

$$\mathbb{E}_{j \sim p_i} [\ell_t^{(k)}(i, j)] = \sum_{j=0}^{d-1} p_{i,j} \cdot \ell_t^{(k)}(i, j) = \langle p_i, \ell_t^{(k)}(i) \rangle$$

Now, at each round  $t$ , FTRL selects a distribution  $p_i \in \Delta_d$  for position  $i$ , then the (FTRL) regret for position  $i$  is defined against the best fixed distribution  $p_i^* \in \Delta_d$ :

$$R_i^{\text{FTRL}}(T) = \sum_{t=1}^T \mathbb{E}_{j \sim p_i} [\ell_t^{(k)}(i, j)] - \min_{p_i^* \in \Delta_d} \sum_{t=1}^T \langle p_i^*, \ell_t^{(k)}(i) \rangle \quad (82)$$

Crucially, since the loss is linear in  $p_i$ , the minimum over the simplex  $\Delta_d$  is achieved at a vertex, i.e., a deterministic action:

$$\min_{p_i^* \in \Delta_d} \sum_{t=1}^T \langle p_i^*, \ell_t^{(k)}(i) \rangle = \min_{j \in [d]} \sum_{t=1}^T \ell_t^{(k)}(i, j) \quad (83)$$

This connects the continuous formulation back to competing against the best fixed discrete action in hindsight. Hence, we can **apply convex optimization results** to obtain regret bounds for the original discrete problem!

**Theorem 12.15 (FTRL Regret under Decomposed Subproblems).** *Consider a combinatorial optimization problem over domain of size  $n$ , decomposed into  $K$  subproblems, each of size  $d$  with overlap  $|\mathcal{O}|$ . Let  $w_{i,j}(t) \in \mathbb{R}_+$  denote the FTRL weight for assigning value  $j \in [d]$  to position  $i \in [n]$  at iteration  $t$ . Using the probability simplex lifting, FTRL maintains cumulative losses and selects actions via:*

$$p_i^{(t)} = \arg \min_{p_i \in \Delta_d} \left\{ \sum_{s=1}^{t-1} \langle p_i, \ell_s(i) \rangle + \frac{1}{\eta} \text{Reg}(p_i) \right\}$$

where  $\text{Reg}(p) = -\sum_{j=1}^d p_j \log p_j$  is the negative entropy regularizer and  $\eta = \sqrt{\frac{2 \log d}{T}}$ .

If the objective function is  $L$ -Lipschitz continuous (Assumption 2), then:

(a) **Per-subproblem bound:** For each subproblem  $k \in [K]$  of size  $d$ , with  $T_k$  active rounds:

$$\sum_{t \in \mathcal{T}_k} [g_k(\mathbf{x}_{I_k}^*) - g_k(\mathbf{x}_{I_{k,t}})] \leq d \sqrt{2 T_k \log d} \quad (84)$$

(b) **Total bound across subproblems:** The total FTRL regret satisfies:

$$R_{\text{FTRL}}(T) \leq d \sqrt{2 K T \log d} + L |\mathcal{O}| \sqrt{T} \quad (85)$$

where  $|\mathcal{O}| = O(KQ)$ .

(c) **Average subproblem regret:**

$$R_{\text{subproblem}}^{\text{avg,FTRL}}(T) = \frac{1}{K} R_{\text{FTRL}}(T) = O \left( d \sqrt{\frac{T \log d}{K}} + \frac{L |\mathcal{O}|}{K} \sqrt{T} \right) \quad (86)$$

The leading FTRL term is  $O(d \sqrt{T \log d})$  and is independent of the number of subproblems  $K$  (up to the  $\sqrt{1/K}$  improvement factor).

*Proof.* FTRL maintains cumulative losses and selects:

$$x_{k,t} = \arg \min_{x_k} \left\{ \sum_{s=1}^{t-1} l_s(x_k) + \frac{1}{\eta} \text{Reg}(x_k) \right\}$$

where  $l_t(i,j) = 1 - r_t$  when position  $i$  has value  $j$ ,  $r_t(x_k) \in [0, 1]$  is normalized reward and regularizer  $\text{Reg}(x) = -\sum_j x_j \log x_j$  (negative entropy) with learning rate  $\eta = \sqrt{\frac{2 \log d}{T}}$ .

Following similar analysis as in previous sections, using instantaneous regret definition:

$$r_t = f(x^*) - \underbrace{\sum_{k=1}^K g_k(x_{I_k}^*)}_{\text{optimal regret } \leq 0} + \underbrace{\sum_{k=1}^K [g_k(x_{I_k}^*) - g_k(x_{I_{k,t}})]}_{\text{FTRL subproblem regret}} + \underbrace{C_t}_{\text{coupling error}} \quad (87)$$

We bound FTRL subproblem regret as in Lemma 12.8.3. Summing over all  $K$  subproblems:

$$\sum_{k=1}^K \sum_{t=1}^{T_k} [g_k(x_{I_k}^*) - g_k(x_{I_{k,t}})] \leq \sum_{k=1}^K d \sqrt{2 T_k \log d}$$

Using Cauchy-Schwarz inequality:

$$\sum_{k=1}^K \sqrt{T_k} \leq \sqrt{K \sum_{k=1}^K T_k} = \sqrt{KT}$$

Therefore:

$$\sum_{k=1}^K d \sqrt{2 T_k \log d} \leq d \sqrt{2 \log d} \cdot \sqrt{KT} = d \sqrt{2KT \log d}$$

Adding the coupling error term  $L|\mathcal{O}| \sqrt{T}$  (where  $|\mathcal{O}| = O(KQ)$ ) gives part (b). Part (c) follows by dividing by  $K$ .  $\square$

## 12.7 Decomposition Validity

In this section, we prove that our metric-based decomposition satisfies the assumptions required for bounded regret. We establish that the decomposition is well-defined, the induced local modifications have bounded effect on the objective, and the overlap structure admits controlled coupling error.

### 12.7.1 Decomposition Construction

**Definition 12.16 (Metric-Based Decomposition).** Let  $(\mathcal{X}, \delta)$  be the solution space equipped with a problem-specific metric  $\delta$ , and let  $D : [n] \times [n] \rightarrow \mathbb{R}_+$  be a problem-specific distance on indices (e.g., geographic distance for TSP, similarity for clustering). We construct subproblems via two complementary strategies:

1. **(Sliding Window)** For initialization and maintaining coverage, define subproblems  $k \in \{1, \dots, K\}$  as:

$$S_k = \{(k-1)(s-o_t) + 1, \dots, \min(n, (k-1)(s-o_t) + s)\}$$

where  $s$  is the subproblem size and  $o_t$  is the overlap at iteration  $t$ . Under diminishing overlap:

$$o_t = \lfloor o_0 \cdot t^{-\alpha} \rfloor$$

for initial overlap  $o_0$  and decay rate  $\alpha > 0$ .

2. (**k**-Nearest Neighbors) For adaptive refinement, given a center  $c \in [n]$ , define:

$$S_c = \arg \min_{S \subseteq [n], |S|=s} \sum_{j \in S} D_{c,j}$$

That is,  $S_c$  contains the  $s$  indices closest to center  $c$  under distance  $D$ .

The full decomposition  $\{S_1, \dots, S_K\}$  combines sliding windows (for coverage) with k-NN subproblems (for problem-structure exploitation).

When overlap is time-varying, we write  $S_k^{(t)}$  to denote the subproblem defined at iteration  $t$ ; when the time index is omitted, the statements apply uniformly over all iterations.

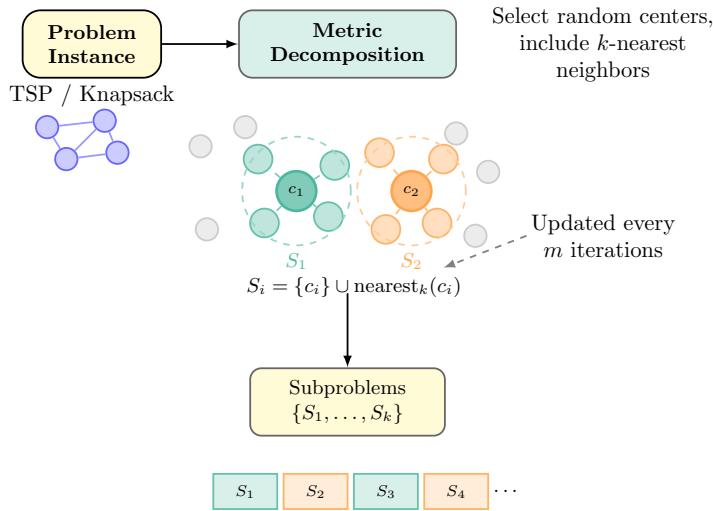


Figure 20: Diagram generated from TikZ code in a separate file.

*Remark 12.17 (Domain Agnosticity).* The decomposition operates on *indices*  $[n]$ , not on solution values. This makes it domain-agnostic: the same construction applies to permutations (TSP), binary vectors (Knapsack), and other combinatorial domains. The metric  $\delta$  on  $\mathcal{X}$  and the distance  $D$  on  $[n]$  are problem-specific parameters.

### 12.7.2 Local Stability

We first establish that modifications restricted to a subproblem induce bounded changes in the objective.

**Lemma 12.18 (Subproblem Diameter Bound).** Let  $S_k \subseteq [n]$  be a subproblem with  $|S_k| = s$ . For any  $x, x' \in \mathcal{X}$  that differ only on coordinates indexed by  $S_k$  (i.e.,  $x_i = x'_i$  for all  $i \notin S_k$ ):

$$\delta(x, x') \leq D(S_k)$$

where  $D(S_k)$  is the diameter of  $S_k$  under metric  $d$ , defined as:

$$D(S_k) := \max_{x, x' \in \mathcal{X}: x_i = x'_i \forall i \notin S_k} \delta(x, x')$$

The diameter depends on the metric structure:

- For Hamming distance:  $D(S_k) \leq |S_k| = s$
- For Kendall tau distance:  $D(S_k) \leq \binom{s}{2} = O(s^2)$
- For weighted metrics:  $D(S_k) \leq \sum_{i \in S_k} w_i$

*Proof.* Since  $x$  and  $x'$  agree on all coordinates outside  $S_k$ , any contribution to  $\delta(x, x')$  must arise from differences within  $S_k$ . For **Hamming distance**,  $d_H(x, x') = |\{i : x_i \neq x'_i\}| \leq |S_k|$  since disagreements can only occur on  $S_k$ . For **Kendall tau distance** on permutations,  $d_\tau(x, x')$  counts inversions. If  $x'$  differs from  $x$  only on positions in  $S_k$ , then all inversions that differ between  $x$  and  $x'$  must involve at least one index in  $S_k$ . The total number of affected inversions is therefore at most  $\binom{|S_k|}{2}$ . For **weighted metrics** of the form  $d_w(x, x') = \sum_i w_i \cdot \mathbf{1}[x_i \neq x'_i]$ , only coordinates in  $S_k$  contribute, giving  $d_w(x, x') \leq \sum_{i \in S_k} w_i$ .  $\square$

**Lemma 12.19 (Local Stability under Metric Decomposition).** *Let  $S_k$  be a subproblem constructed via Definition 12.16. For any  $x, x' \in \mathcal{X}$  differing only on coordinates in  $S_k$ , under Assumption 12.2.2 (Lipschitz continuity):*

$$|f_j(x') - f_j(x)| \leq L \cdot D(S_k)$$

for all objectives  $j \in \{1, \dots, m\}$ .

*Proof.* By Lemma 12.18,  $\delta(x, x') \leq D(S_k)$ . Applying Assumption 12.2.2:

$$|f_j(x') - f_j(x)| \leq L \cdot \delta(x, x') \leq L \cdot D(S_k)$$

$\square$

### 12.7.3 Overlap Structure and Coupling

We now characterize the overlap induced by the decomposition and bound its effect on coupling error.

**Definition 12.20 (Overlap Set and Multiplicity).** Given a decomposition  $\{S_1, \dots, S_K\}$ :

- The *multiplicity* of index  $i$  is  $\rho_i := |\{k : i \in S_k\}|$
- The *maximum multiplicity* is  $\rho := \max_i \rho_i$
- The *overlap set* is  $\mathcal{O} := \{i \in [n] : \rho_i \geq 2\}$
- The *pairwise overlap* between  $S_m$  and  $S_l$  is  $O_{ml} := S_m \cap S_l$

**Lemma 12.21 (Overlap Bounds).** *Let  $\{S_1, \dots, S_K\}$  be a decomposition where each subproblem has size at most  $s$  and each index appears in at most  $\rho$  subproblems. Then:*

(a) **Per-subproblem overlap:**  $|S_k \cap \mathcal{O}| \leq s$

(b) **Pairwise overlap:** For sliding window decomposition with overlap parameter  $o$ :

$$|O_{k,k+1}| = o \quad (\text{consecutive subproblems})$$

(c) **Total overlap counting:**

$$\sum_{k=1}^K |S_k \cap \mathcal{O}| \leq (\rho - 1) \cdot n$$

(d) **Overlap set size:**

$$|\mathcal{O}| \leq \frac{(\rho - 1) \cdot n}{\rho}$$

*Proof.* (a) Since  $|S_k| \leq s$ , trivially  $|S_k \cap \mathcal{O}| \leq |S_k| \leq s$ . (b) For sliding window decomposition with step size  $(s - o)$ , consecutive subproblems  $S_k$  and  $S_{k+1}$  share exactly  $o$  indices by construction. (c) Each index  $i$  with multiplicity  $\rho_i$  is counted in  $\rho_i$  subproblems and contributes to overlap in  $(\rho_i - 1)$  of them. Summing over all indices:

$$\sum_{k=1}^K |S_k \cap \mathcal{O}| = \sum_{i \in \mathcal{O}} (\rho_i - 1) \leq \sum_{i=1}^n (\rho_i - 1) \leq (\rho - 1) \cdot n$$

(d) If  $|\mathcal{O}| = m$ , then the total multiplicity is at least  $2m$  (each overlap index appears  $\geq 2$  times). Since total multiplicity is  $\sum_i \rho_i \leq \rho \cdot n$ , we have  $2m \leq \rho \cdot n$ , giving  $m \leq \rho \cdot n / 2$ . The tighter bound follows from the constraint that non-overlap indices have multiplicity exactly 1. The bound follows by counting total multiplicity:  $\sum_i \rho_i \leq \rho n$ , while each  $i \in \mathcal{O}$  contributes at least  $(\rho_i - 1) \geq 1$  excess appearances beyond the baseline.  $\square$

**Lemma 12.22 (Overlap-Induced Coupling Bound).** *Under Assumption 12.2.3, for any modification on subproblem  $S_k$ :*

$$|f_j(x') - f_j(x)| \leq L \cdot D(S_k) + C \cdot |S_k \cap \mathcal{O}|$$

*Proof.* This follows directly from Assumption 12.2.3 with  $I_k = S_k$ :

$$|f_j(x') - f_j(x)| \leq L \cdot \delta(x, x') + C \cdot |I_k \cap \mathcal{O}|$$

By Lemma 12.18,  $\delta(x, x') \leq D(S_k)$ , yielding the result.  $\square$

#### 12.7.4 Validity of Decomposition

We now prove the main result: our decomposition satisfies the bounded coupling assumption.

**Proposition 12.23 (Decomposition Satisfies Bounded Coupling).** *The metric-based decomposition (Definition 12.16) with subproblem size  $s$  and maximum overlap multiplicity  $\rho$  satisfies Assumption 12.2.3. Specifically, for any local modification on subproblem  $S_k$ :*

$$|f_j(x') - f_j(x)| \leq L \cdot D(s) + C \cdot (\rho - 1) \cdot s$$

where  $D(s)$  is the maximum diameter of a size- $s$  subproblem under metric  $d$ .

*Proof.* Combine Lemma 12.19 (local stability) and Lemma 12.21(a) (per-subproblem overlap bound):

$$\begin{aligned} |f_j(x') - f_j(x)| &\leq L \cdot D(S_k) + C \cdot |S_k \cap \mathcal{O}| && \text{(Lemma 12.22)} \\ &\leq L \cdot D(s) + C \cdot s && (|S_k| \leq s, |S_k \cap \mathcal{O}| \leq s) \end{aligned}$$

For the tighter bound involving  $\rho$ , note that in sliding window decomposition, each position in  $S_k$  belongs to at most  $\rho$  subproblems, so at most  $(\rho - 1) \cdot s$  coordinates in  $S_k$  are in the overlap set  $\mathcal{O}$ . The bound  $C \cdot (\rho - 1) \cdot s$  is tighter when overlap multiplicity is explicitly controlled; the bound  $C \cdot s$  is the uniform worst case (when  $\rho$  is not tracked).  $\square$

**Corollary 12.24 (Bounded Coupling Error per Iteration).** *The coupling error  $C_t$  defined in Definition 12.5 satisfies:*

$$|C_t| \leq C \cdot (\rho_t - 1) \cdot s$$

where  $\rho_t := \max_i |\{k : i \in S_k^{(t)}\}|$  is the maximum overlap multiplicity at iteration  $t$ .

### 12.7.5 Diminishing Overlap

A key feature of our algorithm is diminishing overlap, which reduces coordination overhead as the algorithm converges.

**Lemma 12.25 (Diminishing Overlap Schedule).** *Under the diminishing overlap schedule  $o_t = \lfloor o_0 \cdot t^{-\alpha} \rfloor$  with  $\alpha > 0$ :*

- (a) *The overlap converges:  $\lim_{t \rightarrow \infty} o_t = 0$*
- (b) *The overlap multiplicity converges:  $\lim_{t \rightarrow \infty} \rho_t = 1$*
- (c) *The overlap set vanishes:  $\lim_{t \rightarrow \infty} |\mathcal{O}_t| = 0$*

*Proof.* (a) Since  $t^{-\alpha} \rightarrow 0$  as  $t \rightarrow \infty$  for  $\alpha > 0$ , we have  $o_0 \cdot t^{-\alpha} \rightarrow 0$ , hence  $o_t = \lfloor o_0 \cdot t^{-\alpha} \rfloor \rightarrow 0$ .

(b) With  $o_t = 0$ , the sliding window step size becomes  $s - o_t = s$ , meaning subproblems are disjoint. Each index then belongs to exactly one subproblem, so  $\rho_t = 1$ .

(c) When  $\rho_t = 1$ , no index appears in multiple subproblems, so  $\mathcal{O}_t = \emptyset$ .  $\square$

**Theorem 12.26 (Vanishing Coupling Error).** *Under diminishing overlap with decay rate  $\alpha > 0$ , the coupling error satisfies:*

$$|C_t| \leq C \cdot s \cdot o_0 \cdot t^{-\alpha}$$

In particular:

- (a) *The per-iteration coupling error vanishes:  $C_t \rightarrow 0$  as  $t \rightarrow \infty$*
- (b) *The cumulative coupling error is bounded:*

$$\sum_{t=1}^T C_t \leq \begin{cases} O(T^{1-\alpha}) & \text{if } \alpha < 1 \\ O(\log T) & \text{if } \alpha = 1 \\ O(1) & \text{if } \alpha > 1 \end{cases}$$

*Proof.* (a) By Corollary 12.24,  $|C_t| \leq C \cdot (\rho_t - 1) \cdot s$ . With sliding window overlap  $o_t$ , the multiplicity satisfies  $\rho_t - 1 \leq \lceil o_t/(s - o_t) \rceil \leq O(o_t)$  for  $o_t < s/2$ . Substituting  $o_t = O(t^{-\alpha})$ :

$$|C_t| \leq C \cdot s \cdot O(t^{-\alpha}) = O(t^{-\alpha}) \rightarrow 0$$

(b) Summing the bound:

$$\sum_{t=1}^T C_t \leq C' \sum_{t=1}^T t^{-\alpha}$$

The sum  $\sum_{t=1}^T t^{-\alpha}$  is:

- $O(T^{1-\alpha})$  for  $\alpha < 1$  (integral bound)
- $O(\log T)$  for  $\alpha = 1$  (harmonic series)
- $O(1)$  for  $\alpha > 1$  (convergent series)

$\square$

**Remark 12.27 (Optimal Decay Rate).** The choice  $\alpha = 0.5$  (as in our implementation) gives  $\sum_t C_t = O(\sqrt{T})$ , which matches the  $O(\sqrt{T})$  scaling of the subproblem regret terms. This balances exploration (via overlap) in early iterations with exploitation (via reduced coordination cost) in later iterations.

### 12.7.6 Domain-Agnostic Local Operators

Our algorithm uses domain-agnostic local operators that respect the metric structure.

**Definition 12.28 (Unit Perturbation Operator).** A local operator  $T_S : \mathcal{X} \rightarrow \mathcal{X}$  restricted to indices  $S \subseteq [n]$  is a *unit perturbation* if:

1.  $T_S(x)$  differs from  $x$  only on coordinates in  $S$
2.  $\delta(x, T_S(x)) \leq 1$

### Domain-Specific Unit Perturbations

- *Permutations (Kendall tau)*: A single adjacent swap  $(x_i, x_{i+1}) \mapsto (x_{i+1}, x_i)$  gives  $d_\tau(x, T(x)) = 1$ .
- *Binary vectors (Hamming)*: A single bit flip  $x_i \mapsto 1 - x_i$  gives  $d_H(x, T(x)) = 1$ .
- *Categorical*: Changing one coordinate to an adjacent category.

**Lemma 12.29 (Unit Perturbation Bound).** Let  $T_S : \mathcal{X} \rightarrow \mathcal{X}$  be a unit perturbation operator restricted to indices  $S$ . Then:

$$|f_j(T_S(x)) - f_j(x)| \leq L + C \cdot |S \cap \mathcal{O}|$$

*Proof.* Apply Assumption 12.2.3 with  $\delta(x, T_S(x)) \leq 1$ :

$$|f_j(T_S(x)) - f_j(x)| \leq L \cdot \delta(x, T_S(x)) + C \cdot |S \cap \mathcal{O}| \leq L + C \cdot |S \cap \mathcal{O}|$$

□

*Remark 12.30 (Adaptive Step Sizes).* Lemma 12.29 suggests using larger steps (multiple unit perturbations) when far from the optimum, and smaller steps when close. This motivates adaptive step size selection based on estimated distance to local optima, as implemented in our algorithm.

### 12.7.7 Summary

We have established the following guarantees for our metric-based decomposition:

1. **Local Stability** (Lemma 12.19): Modifications within a subproblem induce objective changes bounded by  $L \cdot D(S_k)$ .
2. **Bounded Coupling** (Proposition 12.23): The decomposition satisfies Assumption 12.2.3 with coupling error  $O((\rho - 1) \cdot s)$ .
3. **Vanishing Coupling** (Theorem 12.26): Under diminishing overlap, cumulative coupling error is  $O(\sqrt{T})$  for decay rate  $\alpha = 0.5$ .
4. **Domain Agnosticity**: The construction applies to any combinatorial domain with a well-defined metric.

These results justify the regret decomposition in Definition 12.5 and the bounds derived in subsequent sections.

## 12.8 Proof Appendix

### 12.8.1 UCB Subproblem Regret Proof

**Lemma 12.31 (UCB Subproblem Regret (Position–Value UCB)).** *For any subproblem  $k$  with index set  $I_k$  and activation set  $\mathcal{T}_k$ , with  $T_k := |\mathcal{T}_k|$ . Assume the normalized scalar reward satisfies  $r_t \in [0, 1]$  and Assumption 12.2.5 (A.5') holds, i.e., there exist  $\{\mu_{i,j}\}_{i \in I_k, j \in \mathcal{A}_i} \subset [0, 1]$  such that  $g_k(x^{(k)}) = \sum_{i \in I_k} \mu_{i,x_i}$  and  $\mathbb{E}[r_t | \mathcal{F}_{t-1}, x_{i,t} = j] = \mu_{i,j} + b_{i,t}$  with  $b_{i,t}$  independent of  $j$ . Suppose subproblem  $k$  is handled by a position–value UCB expert that maintains estimates  $\hat{\mu}_{i,j}$  and counts  $N_{i,j}$  for each  $(i, j)$ , and at each  $t \in \mathcal{T}_k$  updates all positions  $i \in I_k$  for the realized value  $x_{i,t}$  using the same reward  $r_t$ . Let  $A_{\max} := \max_{i \in I_k} |\mathcal{A}_i|$ . Then there exists some constant  $c > 0$  such that*

$$\mathbb{E} [R_k^{\text{UCB}}(\mathcal{T}_k)] := \mathbb{E} \left[ \sum_{t \in \mathcal{T}_k} (g_k(x_k^*) - g_k(x_t^{(k)})) \right] \leq c' |I_k| \sqrt{T_k \log T}$$

where  $x_k^* := (x^*)^{(k)} = x_{I_k}^*$  and  $|I_k|$  is the number of positions/coordinates in subproblem  $k$  and where  $c' = c\sqrt{A_{\max}}$  and  $A_{\max} = O(1)$ , i.e.,  $A_{\max}$  is problem-dependent and treated as a constant.

*Proof sketch.* For a fixed subproblem  $k$  and a round  $t \in \mathcal{T}_k$ . By Assumption 12.2.5 (A.5') the surrogate is locally additive,  $g_k(x^{(k)}) = \sum_{i \in I_k} \mu_{i,x_i}$ , hence, we can write

$$g_k(x_k^*) - g_k(x_t^{(k)}) = \sum_{i \in I_k} (\mu_{i,x_i^*} - \mu_{i,x_{i,t}}) = \sum_{i \in I_k} \Delta_{i,x_{i,t}}, \quad (88)$$

where  $\mu_{i,j}$  is the mean reward of choosing value  $j$  at position  $i$ , and  $\Delta_{i,j} := \mu_{i,x_i^*} - \mu_{i,j}$  is the sub-optimality gap for value  $j$  at position  $i$ . Let  $N_{i,j}(t)$  be the number of times value  $j$  has been selected at position  $i$  up to time  $t$  (restricted to activations of subproblem  $k$ ). Under Assumption 12.2.5, when  $x_{i,t} = j$  we observe  $r_t = \mu_{i,j} + b_{i,t} + \eta_t$  where  $b_{i,t}$  does not depend on  $j$  and  $\eta_t$  is conditionally 1-sub-Gaussian (or bounded).

Thus, for fixed  $(i, j)$ , define the average bias along the samples of arm  $(i, j)$ :

$$\bar{b}_{i,j}(t) := \frac{1}{N_{i,j}(t)} \sum_{s \leq t: x_{i,s}=j} b_{i,s}.$$

Since  $r_s \in [0, 1]$  and  $\eta_s := r_s - \mathbb{E}[r_s | \mathcal{F}_{s-1}, x_{i,s}]$  is a bounded martingale difference, a standard Azuma/Hoeffding argument yields with probability at least  $1 - 2/t^2$ ,

$$\left| \hat{\mu}_{i,j}(t) - \mu_{i,j} - \bar{b}_{i,j}(t) \right| \leq \sqrt{\frac{2 \log t}{N_{i,j}(t)}}. \quad (89)$$

By Assumption 12.2.5 (A.5'), for fixed  $i$  the bias term is independent of  $j$ , so  $\bar{b}_{i,j}(t) = \bar{b}_{i,j'}(t)$  for all  $j, j' \in \mathcal{A}_i$  (hence it cancels in within-position comparisons). Now, the position–value UCB the index for  $(i, j)$  is of the form,

$$\hat{u}_{i,j}(t) = \hat{\mu}_{i,j}(t) + c \sqrt{\frac{2 \log t}{N_{i,j}(t)}},$$

for some constant  $c > 0$ . Suppose at time  $t$  the algorithm selects a suboptimal value  $j \neq x_i^*$  for position  $i$ . Then by the UCB selection rule, its index must be no smaller than the index of the optimal value  $x_i^*$ :

$$\hat{u}_{i,j}(t) \geq \hat{u}_{i,x_i^*}(t). \quad (90)$$

On the event where (89) holds for both  $(i, j)$  and  $(i, x_i^*)$ , combining (90) with the concentration bounds yields

$$\hat{\mu}_{i,j}(t) - \hat{\mu}_{i,x_i^*}(t) \geq c\sqrt{\frac{2\log t}{N_{i,j}(t)}} - c\sqrt{\frac{2\log t}{N_{i,x_i^*}(t)}}.$$

Rewriting the gap  $\Delta_{i,j}$  and using triangle inequality, we obtain:

$$\begin{aligned}\Delta_{i,j} &= \mu_{i,x_i^*} - \mu_{i,j} \\ &= (\mu_{i,x_i^*} - \hat{\mu}_{i,x_i^*}) + (\hat{\mu}_{i,j} - \mu_{i,j}) - (\hat{\mu}_{i,j} - \hat{\mu}_{i,x_i^*}) \\ &\leq \sqrt{\frac{2\log t}{N_{i,j}(t)}} + \sqrt{\frac{2\log t}{N_{i,x_i^*}(t)}} - (\hat{\mu}_{i,j} - \hat{\mu}_{i,x_i^*}) \\ &\leq 2\sqrt{\frac{2\log t}{N_{i,j}(t)}},\end{aligned}$$

where the last inequality uses (90) together with (89) (applied to both  $(i, j)$  and  $(i, x_i^*)$ ) and the fact that the bias terms cancel across values at the same position.

Thus, whenever a suboptimal value  $j \neq x_i^*$  is selected at position  $i$  at time  $t$ , we must have,

$$\Delta_{i,j} \leq 2\sqrt{\frac{2\log t}{N_{i,j}(t)}} \Rightarrow N_{i,j}(t) \leq \frac{8\log t}{\Delta_{i,j}^2} \leq \frac{8\log T}{\Delta_{i,j}^2}.$$

Therefore, the number of times a suboptimal value  $j$  can be selected at position  $i$  is at most  $N_{i,j}(T_k) \leq \min\{T_k, \frac{8\log T}{\Delta_{i,j}^2}\}$ . Plugging this into the regret expression and using (88),

$$\begin{aligned}R_k^{\text{UCB}}(T_k) &= \sum_{t \in \mathcal{T}_k} \sum_{i \in I_k} \Delta_{i,x_{i,t}} = \sum_{i \in I_k} \sum_{j \neq x_i^*} \Delta_{i,j} N_{i,j}(T_k) \\ &\leq \sum_{i \in I_k} \sum_{j \neq x_i^*} \Delta_{i,j} \cdot \min\left\{T_k, \frac{8\log T}{\Delta_{i,j}^2}\right\}.\end{aligned}$$

Rather than continuing with a gap-dependent summation, we invoke the standard gap-free regret bound for UCB. Applying the standard gap-free regret bound for UCB1 [Auer et al. \(2002a\)](#) to the  $A_i$ -armed position- $i$  bandit over  $T_k$  rounds yields,

$$\mathbb{E}[R_i(T_k)] \leq C\sqrt{A_i T_k \log T}.$$

and since  $A_i \leq A_{\max}$ , summing over  $i \in I_k$  yields

$$\mathbb{E}[R_k^{\text{UCB}}(T_k)] \leq \sum_{i \in I_k} C\sqrt{A_i T_k \log T} \leq C|I_k|\sqrt{A_{\max} T_k \log T}.$$

for a universal constant  $C > 0$ . □

### 12.8.2 Exp3 Subproblem Regret Proof

**Lemma 12.32 (Unbiased Exp3 Subproblem Regret).** *Consider a subproblem  $k$  with  $d$  positions, where each position  $i \in I_k$  has an effective choice of size  $d$  (due to decomposition constraints). Let  $T_k$  denote the total iterations spent optimizing subproblem  $k$  and at each iteration  $t$ , the algorithm observes only the total reward  $\rho_t$  for the selected solution  $\mathbf{x}_t$ . Therefore, for each position  $i \in I_k$ , we use importance-weighted reward estimates as:*

$$\hat{\rho}_{i,j}(t) = \frac{\rho_t \cdot \mathbb{I}[x_t(i) = j]}{p_{i,j}(t)}$$

where  $p_{i,j}(t)$  is the probability of selecting value  $j$  for position  $i$ . Standard Exp3 analysis with learning rate  $\eta$  gives:

$$\sum_{t=1}^{T_k} \mathbb{E}[\rho_{i,x_{i^*}} - \rho_{i,a_{i,t}}] \leq \frac{\log d}{\eta} + \frac{\eta T_k B^2}{2}$$

where  $x_i^*$  is the optimal value for position  $i$ ,  $a_{i,t}$  is the value selected at iteration  $t$ ,  $B$  is the maximum reward magnitude, and the expectation is over Exp3's randomized selection.

**(a) Per-subproblem bound:** Summing over all  $d$  positions in subproblem  $k$ :

$$\sum_{t \in T_k} \mathbb{E}[g_k(\mathbf{x}_{I_k}^*) - g_k(\mathbf{x}_{I_{k,t}})] \leq d \left( \frac{\log d}{\eta} + \frac{\eta T_k B^2}{2} \right) \quad (91)$$

**(b) Total bound across subproblems:** The expected regret of the global Exp3 strategy over  $T$  iterations is:

$$\sum_{k=1}^K \left[ \sum_{t \in T_k} \mathbb{E}[g_k(\mathbf{x}_{I_k}^*) - g_k(\mathbf{x}_{I_{k,t}})] \right] \leq d \sqrt{2KT \log d \cdot B^2} \quad (92)$$

*Proof.* For a subproblem  $k$  with  $d$  positions, and assuming each position has an effective choice of size  $d$  due to decomposition, the total regret per subproblem is the sum of regrets for each position  $i \in I_k$ :

$$\sum_{t \in T_k} \mathbb{E}[g_k(\mathbf{x}_{I_k}^*) - g_k(\mathbf{x}_{I_{k,t}})] \leq \sum_{i \in I_k} \left( \frac{\log d}{\eta} + \frac{\eta T_k B^2}{2} \right) = d \left( \frac{\log d}{\eta} + \frac{\eta T_k B^2}{2} \right)$$

With decomposition, each position only has  $d$  relevant values (those compatible with other positions in the subproblem), so we can write:

$$\begin{aligned} \text{Total Exp3 Regret} &= \sum_{k=1}^K \left[ \sum_{t \in T_k} \mathbb{E}[g_k(\mathbf{x}_{I_k}^*) - g_k(\mathbf{x}_{I_{k,t}})] \right] \\ &\leq \sum_{k=1}^K d \left( \frac{\log d}{\eta} + \frac{\eta T_k B^2}{2} \right) \\ &= \sum_{k=1}^K d \left( \frac{\log d}{\eta} \right) + \sum_{k=1}^K d \frac{\eta T_k B^2}{2} \\ &= dK \frac{\log d}{\eta} + d \frac{\eta T B^2}{2} \end{aligned}$$

where in the last line we used  $\sum_{k=1}^K T_k = T$ .

To get the best possible regret bound, we need to choose the optimal value of  $\eta$  that minimizes the upper bound:

$$\begin{aligned} \frac{d}{d\eta} \left[ dK \frac{\log d}{\eta} + d \frac{\eta T B^2}{2} \right] &= 0 \\ -\frac{dK \log d}{\eta^2} + d \frac{T B^2}{2} &= 0 \\ \implies \eta &= \sqrt{\frac{2K \log d}{T B^2}} \end{aligned}$$

Substituting the optimal  $\eta$  back into the inequality:

$$\begin{aligned}
\sum_{k=1}^K \left[ \sum_{t \in T_k} \mathbb{E}[g_k(\mathbf{x}_{I_k}^*) - g_k(\mathbf{x}_{I_{k,t}})] \right] &\leq dK \frac{\log d}{\sqrt{\frac{2K \log d}{TB^2}}} + d \sqrt{\frac{2K \log d}{TB^2} \cdot \frac{TB^2}{2}} \\
&= dK \log d \sqrt{\frac{TB^2}{2K \log d}} + d \sqrt{\frac{2K \log d \cdot TB^2}{4}} \\
&= d \sqrt{KT \log d \cdot B^2} (\sqrt{2}) \\
&\leq d \sqrt{2KT \log d \cdot B^2}
\end{aligned}$$

This completes the proof.  $\square$

### 12.8.3 Follow The Regularized Leader (FTRL)

A learning algorithm that makes decisions based on cumulative losses from all past rounds and uses regularization to prevent overfitting to early observations.

**Lemma 4 (FTRL subproblem regret):** For  $K$  subproblems each of size  $d$ , with overlap set  $O$ , the FTRL regret is:

$$\sum_{k=1}^K \sum_{t=1}^{T_k} [g_k(x_{I_k}^*) - g_k(x_{I_{k,t}})] \leq d \sqrt{KT \log d} \quad (93)$$

**Proof:** FTRL update rule,

$$p_i^{(t)} = \arg \min_{p_i \in \Delta_n} \left\{ \sum_{s=1}^{t-1} \langle p_i, l_s(i) \rangle + \frac{1}{\eta} \text{Reg}(p_i) \right\} \quad (94)$$

Using FTRL standard regret bound with regularizer  $\text{Reg}$  and losses  $l_t$  [ref] states that,

$$\sum_{t=1}^T \langle p^t, l_t \rangle - \sum_{t=1}^T \langle p^*, l_t \rangle \leq \frac{\text{Reg}(p^*) - \text{Reg}(p^{(1)})}{\eta} + \frac{\eta}{2} \sum_{t=1}^T \|g_t\|_*^2$$

Now, for Regularizer  $\text{Reg}$ , since we lifted our discrete space to probability simplex, we use negative entropy defined as,

$$\text{Reg}(p^{(t)}) := - \sum_{j=1}^n p_j^{(t)} \log(p_j^{(t)})$$

where  $\text{Reg}(p^{(1)}) = 0$  when  $p$  is uniform (all weights distributed equally) and  $\max_{p \in \Delta_n} \text{Reg}(p^{(t)}) = \log n$  when  $p^{(t)}$  is delta function. Therefore, we can upper bound the regularizer term in 94 as,

$$\text{Reg}(p^*) - \text{Reg}(p^{(1)}) \leq \log n - 0$$

Now, for bounding the gradient term,  $\|g_t\|_*^2$  which is a dual norm associated with the regularizer, since we use negative entropy regularization and  $g_t = l_t$  (wrt to our loss definition) the associated dual norm is  $l_\infty$  i.e -

$$\sum_{t=1}^T \langle p^t, l_t \rangle - \sum_{t=1}^T \langle p^*, l_t \rangle \leq \frac{\log d}{\eta} + \frac{\eta}{2} \sum_{t=1}^T \|l_t\|_\infty^2$$

Since  $l_t(j) \in [0, 1] \forall j$ ,

$$\|l_t\|_\infty = \max_j |l_t(j)| \leq 1$$

$$\implies \sum_{t=1}^T \langle p^t, l_t \rangle - \sum_{t=1}^T \langle p^*, l_t \rangle \leq \frac{\log d}{\eta} + \frac{T\eta}{2}$$

optimizing for  $\eta$  since RHS is quadratic in  $\eta$ ,

$$\eta = \sqrt{\frac{2\log d}{T}} \text{ substituting it back,}$$

$$\sum_{t=1}^T \langle p^t, l_t \rangle - \sum_{t=1}^T \langle p^*, l_t \rangle \leq \sqrt{2T\log d}$$

This establishes the **regret for a single problem**.

Now,  $\text{Regret}_T = \sum_{t=1}^T \langle p^t, l_t \rangle - \sum_{t=1}^T \langle p^*, l_t \rangle$  and for any subproblem  $k$ , noting the dimension is  $d$  (which relates to  $n$  in the proof) and the time is  $T$ ,

$$\text{Regret}_k = \sum_{t=1}^{T_k} [g_k(x_{I_k}^*) - g_k(x_{I_{k,t}})] \leq d\sqrt{2T_k \log d}$$

Therefore, now for all  $K$  subproblems,

$$\sum_{k=1}^K \text{Regret}_k \leq \sum_{k=1}^K d\sqrt{2T_k \log n}$$

For subproblem  $k$  with  $d$  positions:

$$\begin{aligned} \sum_{k=1}^K \sum_{t=1}^{T_k} [g_k(x_{I_k}^*) - g_k(x_{I_{k,t}})] &= \sum_{k=1}^K \sum_{t=1}^{T_k} \sum_{i \in I_k} [l_t(i, x_{i,t}) - l_t(i, x_i^*)] = \sum_{k=1}^K \sum_{i \in I_k} \sum_{t=1}^{T_k} [\langle p_i^t, l_t(i) \rangle - l_t(i, x_i^*)] \\ &\leq \sum_{k=1}^K \sum_{i \in I_k} \sqrt{2T_k \log n} = \sum_{k=1}^K d\sqrt{2T_k \log d} = d\sqrt{2 \log d} \sum_{k=1}^K \sqrt{T_k} \end{aligned}$$

Using the Cauchy-Schwarz inequality,

$$\begin{aligned} \sum_{k=1}^K \sqrt{T_k} &\leq \sqrt{\sum_{k=1}^K (\sqrt{T_k})^2} \sqrt{\sum_{k=1}^K (1)^2} \leq \sqrt{KT} \\ \implies \sum_{k=1}^K \sum_{t=1}^{T_k} [g_k(x_{I_k}^*) - g_k(x_{I_{k,t}})] &\leq d\sqrt{2KT \log d} \end{aligned}$$

□

#### 12.8.4 Coupling Error: Lagrangian coordination

We employ Lagrangian coordination to handle the overlapping positions that arise from our decomposition of search space. Lagrangian relaxation provides a principled mathematical framework for decomposing optimization problems with coupling constraints.

From our decomposition, each position  $i$  may appear in multiple subproblems. Let  $S_i$  denote the set of subproblems containing position  $i$ , with  $k_i = |S_i|$  being the overlap count. For positions with  $k_i > 1$ , we require coordination. Rather than enforcing hard equality constraints (all subproblems must assign identical values), we use a **soft violation measure** that quantifies coordination risk. For position  $i$  at iteration  $t$ :

$$\xi_i^{(t)} = (k_i - 1) \cdot \text{Var}(\hat{\mu}_i^{(t)}) \cdot \left(1 - \frac{N_{i,v_i^{(t)}}}{\sum_v N_{i,v}}\right) \quad (95)$$

This continuous measure captures: (1) overlap degree ( $k_i - 1$ ), (2) value estimate uncertainty  $\text{Var}(\hat{\mu}_i)$ , and (3) assignment confidence (visit ratio). As learning progresses,  $\xi_i^{(t)} \rightarrow 0$ , naturally relaxing coordination penalties. Now, let  $O = \{i : k_i > 1\}$  denote the set of overlapping positions. Then, Lagrangian for our coordination problem is:

$$L(x, \lambda) = f(x) + \sum_{i \in O} \lambda_i \xi_i(x)$$

where  $f(x)$  is the original objective (total reward) and  $\lambda_i \geq 0$  are dual variables penalizing soft violations. This follows a primal-dual relationship where:

$$\text{Primal Problem: } \min_x f(x) \quad \text{s.t.} \quad \xi_i(x) = 0 \quad \forall i \in O \quad (96)$$

$$\text{Dual Problem: } \max_{\lambda \geq 0} g(\lambda) \quad \text{where} \quad g(\lambda) = \min_x L(x, \lambda) \quad (97)$$

Now, for our combinatorial setting, the primal objective  $f(x)$  is non-convex and non-smooth. However, the dual function  $g(\lambda) = \min_x [f(x) + \lambda^T \xi(x)]$  is **always concave** (even when the primal is non-convex!), as it is the pointwise minimum of affine functions in  $\lambda$  [Boyd et al. \(2003\)](#).

Therefore, we solve the Lagrangian dual problem by maximizing the concave function  $g(\lambda)$ . Equivalently, we minimize the convex function  $-g(\lambda)$ . The subgradient of  $-g(\lambda)$  at  $\lambda$  is  $-\xi(x_t)$  where  $x_t = \arg \min_x L(x, \lambda_t)$ .

**Accelerated Dual Optimization via Mirror Descent.** Standard subgradient ascent for maximizing  $g(\lambda)$  achieves  $O(\sqrt{T})$  cumulative dual gap. Since our dual function  $g(\lambda)$  is concave (making  $-g(\lambda)$  convex), we can apply **online convex optimization (OCO)** methods and we use **mirror descent** with entropic regularization operating in log-space:  $\lambda_{t+1} = \exp(\log(\lambda_t) + \alpha_t \xi_t)$ . This offers three advantages: (1) the exponential update **automatically enforces**  $\lambda \geq 0$  without projection, (2) multiplicative updates are **scale-adaptive**, and (3) mirror descent is optimal for bounded domains [Bubeck \(2015\)](#). We apply **Nesterov momentum**  $\lambda_{t+1} \leftarrow \tilde{\lambda}_t + \theta_t (\tilde{\lambda}_t - \lambda_t)$  with  $\theta_t = 2/(t+1)$ , which improves the problem-dependent constants and accelerates early-iteration convergence while maintaining the optimal  $O(\sqrt{T})$  rate

Using mirror descent with Nesterov acceleration to optimize the dual variables  $\lambda$ , we obtain the following bound on coupling error.

**Lemma 12.33 (Dual Gap Bound for Mirror Descent).** *Let  $g : \mathcal{R}^d \rightarrow \mathcal{R}$  be a concave function with optimum at  $\lambda^* \in \mathcal{R}^d$  satisfying  $\lambda^* \in [0, \lambda_{\max}]^d$ . For all  $t \in \{1, \dots, T\}$ , let  $\xi_t \in \mathbb{R}^d$  be a subgradient of  $g$  at  $\lambda_t$  with  $\|\xi_t\|_\infty \leq L$ . Assume  $\lambda_1 \in [0, \lambda_{\max}]^d$ . Under mirror descent with entropic regularization and update rule:*

$$\lambda_{t+1} = \lambda_{\max} [\exp(\log(\lambda_t) + \alpha_t \xi_t)] \quad (98)$$

with step size  $\alpha_t = \frac{\alpha_0}{\sqrt{t}}$ , the cumulative dual gap is bounded by:

$$\sum_{t=1}^T (g(\lambda^*) - g(\lambda_t)) \leq O(d\lambda_{max}L\sqrt{T}) \quad (99)$$

*Proof.* The mirror descent update for the Lagrangian dual  $\lambda_t$  operates in the dual space defined by the entropic regularizer  $\Phi(\lambda) = \sum_i \lambda_i \log \lambda_i$ . Starting from the general mirror descent iteration, we specialize to the entropic regularizer  $\Phi(\lambda) = \sum_i \lambda_i \log \lambda_i$  to obtain our exponential update rule and then bound convergence using Bregman divergence analysis.

Now, the standard mirror descent update for minimizing any function  $f$  is:

$$\lambda_{t+1} = \arg \min_{\lambda} \left\{ \langle \nabla f(\lambda_t), \lambda \rangle + \frac{1}{\alpha_t} D_{\Phi}(\lambda || \lambda_t) \right\} \quad (100)$$

where  $D_{\Phi}(\lambda || \mu) = \Phi(\lambda) - \Phi(\mu) - \langle \nabla \Phi(\mu), \lambda - \mu \rangle$  is the Bregman divergence.

Since we are maximizing  $g(\lambda)$ , we minimize  $f(\lambda) = -g(\lambda)$ , so  $\nabla f(\lambda_t) = -\xi_t$  where  $\xi_t$  is a subgradient of  $g$ , therefore,

$$\lambda_{t+1} = \arg \min_{\lambda} \left\{ -\langle \xi_t, \lambda \rangle + \frac{1}{\alpha_t} D_{\Phi}(\lambda || \lambda_t) \right\} \quad (101)$$

Expanding the Bregman divergence,

$$\begin{aligned} \lambda_{t+1} &= \arg \min_{\lambda} \left\{ -\langle \xi_t, \lambda \rangle + \frac{1}{\alpha_t} [\Phi(\lambda) - \Phi(\lambda_t) - \langle \nabla \Phi(\lambda_t), \lambda - \lambda_t \rangle] \right\} \\ \lambda_{t+1} &= \arg \min_{\lambda} \left\{ -\langle \xi_t, \lambda \rangle + \frac{1}{\alpha_t} \Phi(\lambda) - \frac{1}{\alpha_t} \Phi(\lambda_t) - \frac{1}{\alpha_t} \langle \nabla \Phi(\lambda_t), \lambda \rangle + \frac{1}{\alpha_t} \langle \nabla \Phi(\lambda_t), \lambda_t \rangle \right\} \end{aligned}$$

The terms  $-\frac{1}{\alpha_t} \Phi(\lambda_t)$  and  $+\frac{1}{\alpha_t} \langle \nabla \Phi(\lambda_t), \lambda_t \rangle$  are constants with respect to  $\lambda$  (they only depend on  $\lambda_t$ , which is fixed at iteration  $t$ ). Since adding a constant to an objective does not change its minimizer, we drop these terms and rearrange to get,

$$\lambda_{t+1} = \arg \min_{\lambda} \left\{ \frac{1}{\alpha_t} \Phi(\lambda) - \left\langle \xi_t + \frac{1}{\alpha_t} \nabla \Phi(\lambda_t), \lambda \right\rangle \right\}$$

By first-order optimality condition (dual space update), taking the gradient with respect to  $\lambda$  and setting it to zero,

$$\begin{aligned} \frac{\partial}{\partial \lambda} \left[ \frac{1}{\alpha_t} \Phi(\lambda) - \left\langle \xi_t + \frac{1}{\alpha_t} \nabla \Phi(\lambda_t), \lambda \right\rangle \right] \Big|_{\lambda=\lambda_{t+1}} &= 0 \\ \frac{1}{\alpha_t} \nabla \Phi(\lambda_{t+1}) - \left( \xi_t + \frac{1}{\alpha_t} \nabla \Phi(\lambda_t) \right) &= 0 \end{aligned}$$

Multiplying by  $\alpha_t$  and rearranging:

$$\nabla \Phi(\lambda_{t+1}) = \nabla \Phi(\lambda_t) + \alpha_t \xi_t \quad (102)$$

This is the **dual space update rule**. The update happens in the "gradient space" of  $\Phi$ , not directly in  $\lambda$ -space. We add the subgradient  $\alpha_t \xi_t$  to  $\nabla \Phi(\lambda_t)$  to get  $\nabla \Phi(\lambda_{t+1})$ .

Note, the dual space update  $\nabla \Phi(\lambda_{t+1}) = \nabla \Phi(\lambda_t) + \alpha_t \xi_t$  holds for *any* choice of regularizer  $\Phi$  [ref for standard

gradient descent]. We choose entropic regularization  $\Phi(\lambda) = \sum_i \lambda_i \log \lambda_i$  because it naturally handles the constraint  $\lambda \geq 0$  (dual variables must be non-negative). For this choice, the gradient is:

$$\frac{\partial \Phi}{\partial \lambda_i} = \frac{\partial}{\partial \lambda_i} [\lambda_i \log \lambda_i] = \log \lambda_i + \lambda_i \cdot \frac{1}{\lambda_i} = \log \lambda_i + 1 \quad (103)$$

Therefore (component-wise):

$$\nabla \Phi(\lambda) = \log \lambda + \mathbf{1} \quad (104)$$

where  $\mathbf{1} = (1, 1, \dots, 1)^\top$  is the vector of all ones. Substituting this into the dual space update we can derive the exponential update as,

$$\begin{aligned} \nabla \Phi(\lambda_{t+1}) &= \nabla \Phi(\lambda_t) + \alpha_t \xi_t \\ \log \lambda_{t+1} + \mathbf{1} &= \log \lambda_t + \mathbf{1} + \alpha_t \xi_t \\ \log \lambda_{t+1} &= \log \lambda_t + \alpha_t \xi_t \\ \lambda_{t+1} &= \exp(\log \lambda_t + \alpha_t \xi_t) \end{aligned}$$

This gives the multiplicative update (component-wise):

$$\lambda_{i,t+1} = \lambda_{i,t} \cdot \exp(\alpha_t \xi_{i,t}) \quad (105)$$

followed by projection onto  $[0, \lambda_{max}]^d$ .

For convergence analysis, instead of bounding the optimality gap or primal-dual gap, we bound the **cumulative dual gap**, defined as  $\sum_{t=1}^T [g(\lambda^*) - g(\lambda_t)]$ , which measures how close our current dual solution is to the optimal dual solution over all  $T$  iterations. Bounding the sum of these gaps over time guarantees that, on average, the Lagrangian coordination (LC) mechanism is making good progress toward optimality.

For that purpose, in standard sub-gradient analysis, one bounds progress using Euclidean distance  $\|\lambda_t - \lambda^*\|_2^2$ . For mirror descent, we instead use the Bregman divergence induced by the regularizer  $\Phi$ . For our entropic regularizer  $\Phi(\lambda) = \sum_i \lambda_i \log \lambda_i$ , this becomes:

$$D_\Phi(\lambda || \mu) = \sum_i \lambda_i \log \left( \frac{\lambda_i}{\mu_i} \right) - \sum_i (\lambda_i - \mu_i) \quad (106)$$

which is the **KL-divergence** (up to sign conventions). The Bregman divergence plays the same role as squared Euclidean distance, but is adapted to the geometry of entropic regularization.

Using the fundamental property of concave functions (first-order condition for concavity) [Boyd and Vandenberghe \(2004\)](#), for any two points  $\lambda^*, \lambda_t$ :

$$g(\lambda^*) \leq g(\lambda_t) + \langle \nabla g(\lambda_t), \lambda^* - \lambda_t \rangle \quad (107)$$

Substituting sub-gradient  $g(\lambda^*)$  and rearranging,

$$g(\lambda^*) - g(\lambda_t) \leq \langle \xi_t, \lambda^* - \lambda_t \rangle \quad (108)$$

Therefore, the dual gap at iteration  $t$  is related to the distance between the dual variables! Formally, the dual gap is bounded by the inner product of the sub-gradient with the distance to the optimum. This connects the dual gap to the geometry of the dual space, which we will exploit using Bregman divergences.

Using Three-Point property of mirror descent [Beck and Teboulle \(2003\)](#); [Bubeck \(2015\)](#) that relates the Bregman divergence before and after an update  $\lambda_{t+1} = \arg \min_{\lambda} \{\alpha_t \langle \xi_t, \lambda \rangle + D_{\Phi}(\lambda || \lambda_t)\}$ :

$$D_{\Phi}(\lambda^* || \lambda_t) - D_{\Phi}(\lambda^* || \lambda_{t+1}) \geq \alpha_t \langle \xi_t, \lambda^* - \lambda_{t+1} \rangle \quad (109)$$

Using the sub-gradient inequality (108) ,

$$\begin{aligned} \alpha_t(g(\lambda^*) - g(\lambda_t)) &\leq \alpha_t \langle \xi_t, \lambda^* - \lambda_t \rangle \\ \langle \xi_t, \lambda^* - \lambda_t \rangle &= \langle \xi_t, \lambda^* - \lambda_t \rangle + \xi_t \lambda_{t+1} - \xi_t \lambda_{t+1} = \langle \xi_t, \lambda^* - \lambda_{t+1} \rangle + \langle \xi_t, \lambda_{t+1} - \lambda_t \rangle \\ &= \alpha_t \langle \xi_t, \lambda^* - \lambda_{t+1} \rangle + \alpha_t \langle \xi_t, \lambda_{t+1} - \lambda_t \rangle \\ &\leq D_{\Phi}(\lambda^* || \lambda_t) - D_{\Phi}(\lambda^* || \lambda_{t+1}) + \alpha_t \langle \xi_t, \lambda_{t+1} - \lambda_t \rangle \end{aligned}$$

where the second inequality uses (109). Now to bound the Residual term  $\alpha_t \langle \xi_t, \lambda_{t+1} - \lambda_t \rangle$ , we use the fact that in log-space (see the mirror descent update):

$$\begin{aligned} \log \lambda_{t+1} - \log \lambda_t &= \alpha_t \xi_t \\ \lambda_{i,t+1} &= \lambda_{i,t} \exp(\alpha_t \xi_{i,t}) \end{aligned}$$

Therefore:

$$\lambda_{i,t+1} - \lambda_{i,t} = \lambda_{i,t} [\exp(\alpha_t \xi_{i,t}) - 1]$$

Using the inequality  $e^x - 1 \leq xe^{|x|}$  for all  $x$ ,

$$\begin{aligned} |\lambda_{i,t+1} - \lambda_{i,t}| &= \lambda_{i,t} |\exp(\alpha_t \xi_{i,t}) - 1| \\ &\leq \lambda_{i,t} \cdot \alpha_t |\xi_{i,t}| \cdot e^{\alpha_t |\xi_{i,t}|} \\ &\leq \lambda_{max} \cdot \alpha_t L \cdot e^{\alpha_t L} \end{aligned}$$

where as stated before  $\lambda_{i,t} \leq \lambda_{max}$  and  $|\xi_{i,t}| \leq \|\xi_t\|_{\infty} \leq L$ .

Using this, we can now simplify the residual term as,

$$\begin{aligned} |\langle \xi_t, \lambda_t - \lambda_{t+1} \rangle| &\leq \sum_i |\xi_{i,t}| \cdot |\lambda_{i,t+1} - \lambda_{i,t}| \\ &\leq \sum_i L \cdot \lambda_{max} \alpha_t L e^{\alpha_t L} \\ &= d \lambda_{max} L^2 \alpha_t e^{\alpha_t L} \end{aligned}$$

Now, for small step sizes  $\alpha_t L \ll 1$ , we have  $e^{\alpha_t L} \approx 1 + \alpha_t L$ , giving:

$$\alpha_t \langle \xi_t, \lambda_{t+1} - \lambda_t \rangle \leq d \lambda_{max} L^2 \alpha_t^2 + d \lambda_{max} L^3 \alpha_t^3$$

For small step sizes we can ignore higher order terms and it can be absorbed in  $O(\cdot)$ ,

$$\alpha_t \langle \xi_t, \lambda_{t+1} - \lambda_t \rangle \leq \alpha_t^2 \|\xi_t\|_{\infty}^2 \cdot d \lambda_{max} \quad (110)$$

$$\implies \alpha_t(g(\lambda^*) - g(\lambda_t)) \leq D_{\Phi}(\lambda^* || \lambda_t) - D_{\Phi}(\lambda^* || \lambda_{t+1}) + \alpha_t^2 L^2 d \lambda_{max} \quad (111)$$

Summing both sides for  $t = 1, \dots, T$ , we get a Telescoping sum,

$$\begin{aligned}
\sum_{t=1}^T \alpha_t (g(\lambda^*) - g(\lambda_t)) &\leq \sum_{t=1}^T [D_\Phi(\lambda^* || \lambda_t) - D_\Phi(\lambda^* || \lambda_{t+1})] + \sum_{t=1}^T \alpha_t^2 L^2 d\lambda_{max} \\
&= D_\Phi(\lambda^* || \lambda_1) - D_\Phi(\lambda^* || \lambda_{T+1}) + \sum_{t=1}^T \alpha_t^2 L^2 d\lambda_{max} \\
&\leq D_\Phi(\lambda^* || \lambda_1) + L^2 d\lambda_{max} \sum_{t=1}^T \alpha_t^2 \\
&\leq d\lambda_{max} \log(\lambda_{max}) + L^2 d\lambda_{max} \sum_{t=1}^T \alpha_t^2
\end{aligned}$$

Since  $D_\Phi(\lambda^* || \lambda_{T+1}) \geq 0$  and  $D_\Phi(\lambda^* || \lambda_1) \leq d\lambda_{max} \log(\lambda_{max})$  for bounded domain.

Let's choose constant step size  $\alpha_t = \alpha$  for simplicity,

$$\sum_{t=1}^T (g(\lambda^*) - g(\lambda_t)) \leq \frac{d\lambda_{max} \log(\lambda_{max})}{\alpha} + \alpha L^2 d\lambda_{max} T \tag{112}$$

Optimizing over  $\alpha$  by setting  $\frac{d}{d\alpha} = 0$  we get  $\alpha^* = \sqrt{\frac{\log(\lambda_{max})}{L^2 T}}$ . Substituting this above,

$$\sum_{t=1}^T (g(\lambda^*) - g(\lambda_t)) \leq 2d\lambda_{max} L \sqrt{T \log(\lambda_{max})} = O(d\lambda_{max} L \sqrt{T}) \tag{113}$$

Additionally, note that while we use acceleration it has no effect on the asymptotic bound derivation above so we used the standard mirror descent approach to derive the worst-case bound.  $\square$

### 12.8.5 Coupling Error: Disagreement requires uncertainty

**Lemma 12.34** (Disagreement requires uncertainty). *For position  $i$  contained in  $k_i > 1$  subproblems, let  $\sigma_i^2(t) := \text{Var}(\hat{\mu}_i^{(t)})$  denote the variance of value estimates. Then:*

$$\mathbb{E}[\mathbb{I}[x_{m,i}^{(t)} \neq x_{l,i}^{(t)}]] \leq c_0 \cdot \sigma_i^2(t)$$

for a constant  $c_0 > 0$  depending on the selection mechanism.

*Proof.* Disagreement between subproblems  $m$  and  $l$  on position  $i$  requires that they select different values. Since all subproblems share global parameters (value estimates  $\hat{\mu}_{i,j}$ , weights  $w_{i,j}$ ), they compute identical selection scores for each position-value pair  $(i, j)$ .

Disagreement arises due to stochasticity in the selection mechanism (e.g., randomized tie-breaking, Hedge/EXP3 sampling, or noise in reward observations). Under any score-based randomized selection rule, the probability that two subproblems select different values is controlled by the spread of the score distribution, which is captured by the variance  $\sigma_i^2(t)$ . In particular, if  $\sigma_i^2(t) = 0$ , all value estimates are identical and all subproblems select the same (highest-estimated) value with probability one. When  $\sigma_i^2(t) > 0$ , the probability of selecting different values is at most proportional to  $\sigma_i^2(t)$ . Similar arguments apply to EXP3 and FTRL-style methods. A formal derivation for softmax- or Hedge-style sampling follows from standard bounds relating selection entropy or total variation distance to score variance. For UCB-based methods, subproblems select  $(\arg \max_j \text{UCB}_{i,j}(t))$ , which

is deterministic given shared parameters. Disagreement can therefore occur only when estimates are sufficiently close that stochastic noise in the reward observations flips the ordering; by standard concentration arguments, this event occurs with probability  $O(\sigma_i^2(t))$ . For EXP3/FTRL-style methods, subproblems compute identical sampling distributions  $p_i^{(t)}$  from shared weights. If a single global sample is used, no disagreement occurs; however, when subproblems sample independently or when stochasticity arises from delayed or noisy updates, disagreement can occur with probability controlled by the entropy (or variance) of  $p_i^{(t)}$ , which is again captured by  $\sigma_i^2(t)$ . In all cases, disagreement vanishes as  $\sigma_i^2(t) \rightarrow 0$ , which is sufficient for the regret analysis.  $\square$