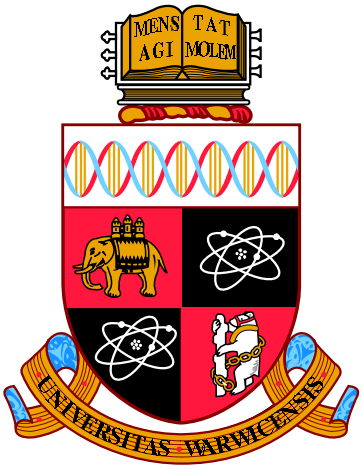

Efficient Information Collection in Stochastic Optimisation

by

Xin Fei

Thesis

Submitted to the University of Warwick

in partial fulfilment of the requirements

for admission to the degree of

Doctor of Philosophy

Warwick Business School

May 2019

Contents

List of Tables

List of Figures

Acknowledgments

This thesis would not have been possible without the guidance and the help of many people. First and foremost, I would like to express my deepest appreciation to my supervisors, Dr. Nalân Gülpınar and Prof. Jürgen Branke, for their support and encouragement. It has been my honour to be one of their Ph.D. students. They granted me the freedom to choose research directions and support me to explore the unknown domains. Moreover, they have taught me, both consciously and unconsciously, how to become a scientific researcher. I appreciate all their contributions of time, effort and ideas to make my Ph.D. experience productive.

I would like to extend my sincere thanks to Prof. Bo Chen, Prof. Leroy White, Prof. Steve Alpern and Dr. Arne K Strauss for their valuable advice during the Ph.D. upgrade, annual review and completion review. I also have to thank the members of my thesis defence committee, Dr. Xuan Vinh Doan, Professor Kevin Glazebrook, and Dr. Vladimir Deineko for their helpful comments and suggestions.

My grateful thanks also go to my colleagues and friends: Bingqing Lan, Folarin Oyebolu, Gongtai Wang, Mustafa Demirbilek, Wen Zhang, Ruini Qu for their unstinting help, support and companionship.

Last but not the least, I would also like to say a heartfelt thank you to my Mum and Dad for always believing in me and encouraging me to follow my dreams.

Declarations

I declare that any material contained in this thesis has not been submitted for a degree to any other University. I further declare that one paper titled "Efficient Solution Selection for Two-stage Stochastic Programs", drawn from Chapter 2 of this thesis, was co-authored with Nalân Gülpınar and Jürgen Branke. It has been accepted in *European Journal of Operational Research*. Furthermore, the paper titled "New Sampling Strategies when Searching for Robust Solutions", drawn from Chapter 3 of this thesis, was co-authored with Jürgen Branke and Nalân Gülpınar. It has been accepted in *IEEE Transactions on Evolutionary Computation*. Finally, the paper titled "An Adaptive Rollout Algorithm for Clinical Trial Scheduling and Resource Allocation in Drug Development Under Uncertainty", drawn from Chapter 4 of this thesis, was co-authored with Nalân Gülpınar and Jürgen Branke.

Xin Fei

September 2018

Abstract

This thesis focuses on a class of information collection problems in stochastic optimisation. Algorithms in this area often need to measure the performances of several potential solutions, and use the collected information in their search for high-performance solutions, but only have a limited budget for measuring. A simple approach that allocates simulation time equally over all potential solutions may waste time in collecting additional data for the alternatives that can be quickly identified as non-promising. Instead, algorithms should amend their measurement strategy to iteratively examine the statistical evidences collected thus far and focus computational efforts on the most promising alternatives. This thesis develops new efficient methods of collecting information to be used in stochastic optimisation problems.

First, we investigate an efficient measurement strategy used for the solution selection procedure of two-stage linear stochastic programs. In the solution selection procedure, finite computational resources must be allocated among numerous potential solutions to estimate their performances and identify the best solution. We propose a two-stage sampling approach that exploits a Wasserstein-based screening rule and an optimal computing budget allocation technique to improve the efficiency of obtaining a high-quality solution. Numerical results show our method provides good trade-offs between computational effort and solution performance.

Then, we address the information collection problems that are encountered in the search for robust solutions. Specifically, we use an evolutionary strategy to solve a class of simulation optimisation problems with computationally expensive blackbox functions. We implement an archive sample approximation method to

reduce the required number of evaluations. The main challenge in the application of this method is determining the locations of additional samples drawn in each generation to enrich the information in the archive and minimise the approximation error. We propose novel sampling strategies by using the Wasserstein metric to estimate the possible benefit of a potential sample location on the approximation error. An empirical comparison with several previously proposed archive-based sample approximation methods demonstrates the superiority of our approaches.

In the final part of this thesis, we propose an adaptive sampling strategy for the rollout algorithm to solve the clinical trial scheduling and resource allocation problem under uncertainty. The proposed sampling strategy method exploits the variance reduction technique of common random numbers and the empirical Bernstein inequality in a statistical racing procedure, which can balance the exploration and exploitation of the rollout algorithm. Moreover, we present an augmented approach that utilises a heuristic-based grouping rule to enhance the simulation efficiency by breaking down the overall action selection problem into a selection problem involving small groups. The numerical results show that the proposed method provides competitive results within a reasonable amount of computational time.

Chapter 1

Introduction and Background

The challenge of collecting relevant information to support decision-making arises naturally in various real-world applications. An example is news article recommendation. News websites often recommend a small set of news articles to visitors to improve user satisfaction and increase website traffic. The background monitoring system needs to identify the most popular content as quickly as possible by collecting visitors' preferences, which are determined based on clicks or search terms. In this context, a contextual bandit algorithm could be utilised to improve the efficiency of information collection [Li et al., 2010]. Artificial intelligence game playing is another example in which efficient information collection is necessary. Google DeepMind's AlphaGo defeated Ke Jie, one of the best Go players, during the 2017 Future of Go Summit. Go is characterised by uncertain opponent actions and numerous possible moves at each step. The challenge for AlphaGo was to determine the best move by exploring the possible future outcomes of each move within a limited time. To address this challenge, AlphaGo employed a sampling algorithm called upper confidence bounds applied to trees to guide the measurement process [Esteva et al., 2017]. Information collection problems also exist in the dose-finding design. Pharmaceutical companies recruit participants to collect clinical data for testing the safety and efficacy of experimental drugs in various dosages. However, different participants might react differently to the experimental drugs. Companies need an efficient clinical trial design to identify the optimal dosage that satisfies pre-specified clinical goals [Thall, 2012].

1.1 Review of Efficient Information Collection

The diversity of real-world problems pose numerous challenges in information collection that inspire new thoughts and research directions. Many research communities have been formed to advance the state-of-the-art in measurement strategies and resolve practical information collection problems through theoretical and methodological innovation. Powell and Ryzhov [2012] performed a comprehensive review of efficient information collection approaches and their applications. In the following, we review several studies that are more closely related to the works of this thesis.

Statistical ranking and selection (R&S) is one of research streams that have devoted extensive efforts to address such information collection problems. This research field is concerned with problems where a finite set of competing alternatives, such as business rules, engineering designs or scheduling policies, is to be compared within a certain amount of time. The goal of this process is to assign evaluation time as efficiently as possible across alternatives to select the best alternative when evaluation time runs out. Three main approaches are often cited in the literature, namely, indifference-zone, optimal computing budget allocation and expected value-of-information. The indifference-zone method was firstly proposed by Bechhofer [1954] and could find the best whenever there exists a zone between the best and second best alternatives. The earliest version is not an anytime algorithm, which means that a specified simulation rule (i.e. simulation replications for each alternative) is followed to finish all requested simulations to ensure that a probability guarantee is achieved. Kim and Nelson [2001] and Jeff Hong [2006] later extended the idea of indifference zone to be a sequential procedure that can stop whenever the simulation budget is exhausted, but it no longer has such probability of correct selection guarantee. Recently, Fan et al. [2016] imposed an assumption of unequal means and proposed a new indifference-zone formulation without the need to specify the indifference zone. Optimal computing budget allocation is a Bayesian decision-theoretic approach that iteratively maximise the probability of correct selection. Chen [1995] first proposed this appraoch and applied the Bonferroni inequality to create a lower bound approximation for the probability of correct selection, so that the algorithm could evaluate the potential improvement by using the predictive posterior distributions. Some recent works in this framework can be found, for example, in Chen et al. [2000]; Fu et al. [2007] and Peng et al. [2016]. Moreover, some variants of optimal computing budget allocation take the expected opportunity cost of wrong selection as the simulation objective (e.g. see He et al. [2007] and Gao and Chen [2015]). Another class of Bayesian approach is so-called

expected value-of-information that myopically collects the sampling information to improve the decision-maker's knowledge state. Chick and Inoue [2001] first proposed a two-stage sampling procedure in the principle of expected value-of-information. Chick et al. [2010] developed two sequential approaches that iteratively assign a small amount of samples to several or one designs per sampling stage. Frazier et al. [2009] later implemented this idea and proposed a so-called knowledge gradient method to handle the the R&S problem with correlated normal beliefs. Scott et al. [2011] extended the idea of knowledge gradient to account for continuous decision variables. For a comprehensive review on R&S methods, readers are referred to Branke et al. [2007].

The studies on multi-armed bandit (MAB) problems constitute another important research community. In the MAB problem, a slot machine with multiple arms is given, each of which gives a stochastic reward when pulled. The distributions for the stochastic rewards collected from various arms are unknown. The objective is to play the arms one-by-one in an optimal sequence to maximise the expected total reward in the bandit process. A good balance between exploitation (pulling the best arm given current information) and exploration (collecting more information to find a better arm) is needed. The MAB problem was first studied during the Second World War and is challenging to be solved optimally. Gittins [1979] proposed an index policy that computes a so-called "Gittins index" for each arm and chooses the arm with the largest index value at each period of the bandit process. The Gittins index policy has been proved optimally for infinite-horizon stationary MAB problems with time-discounted rewards. Lai and Robbins [1985] further showed that the Gittins index policy can be asymptotically optimal in the finite-horizon bandit process. On the other hand, the Gittins index policy is no longer guaranteed to be optimal for the restless bandit problem where selected and unselected arms respectively will produce various rewards, and their states continue to evolve regardless of the player's actions. Papadimitriou and Tsitsiklis [1999] showed that the computational complexity of restless bandits is $PSPACE$ hard. Whittle [1988] first studied the restless bandits with time-average rewards. He applied the Lagrange relaxation to reduce the complexity of solving the restless bandits model and thus received a Whittle index policy. The success of Whittle's method relies on a structural property called "indexability", which might not be hold for all restless bandits. In the literature, the indexability property has been established for some special classes of restless bandits, such as stochastic scheduling problems [Glazebrook et al., 2006], bidirectional restless bandits [Glazebrook et al., 2013] and reinitialising restless bandits [Villar, 2016]. For more details on the Gittins and Whittle index policies, we

refer readers to the book by Gittins et al. [2011]. Another class of index policies that have received considerable attention in recent years is the upper confidence bound policy. The index of each arm is calculated as the sum of empirical average reward and a "bonus" term. Similar with Gittins-like index policies, the upper confidence bound policies choose the arm with the highest index value at each period. For the stationary MAB problems, the basic setting of the bonus term is defined by Hoeffding confidence bounds [Auer et al., 2002]. In the literature, some variants of bonus terms can be found, e.g. sample variance based confidence bounds [Audibert et al., 2007], or relative entropy-based confidence bounds [Cappé et al., 2013]. Moreover, some studies were presented to deal with the case of restless bandits, see Garivier and Moulines [2011] and Bouneffouf and Féraud [2016]. The research on the use of efficient measurement strategies for optimisation problems is closely related to our works. For instance, in the traffic network optimisation problems, Ryzhov and Powell [2011] assumed the time taken to pass through an individual edge is stochastic and applied the knowledge gradient policy to collect information on the edges to find the shortest path. Ryzhov and Powell [2012] studied a general linear programming problem with stochastic input parameters. One needs to identify the location of the optimal solution by collecting relevant information on input parameters. The challenge is that inaccurate estimations of input parameters might mislead the simplex algorithm about vertex locations. Defourny et al. [2015] assumed that the parameters of the objective function in a robust optimisation model are unknown, and proposed an efficient measurement strategy to refine the optimisation model. Moreover, efficient information collection methods are widely used in the simulation optimisation problem. Jones et al. [1998] considered a simulation optimisation problem with expensive black-box functions, and proposed an efficient global optimisation method to sequentially update a Kriging meta-model of the black-box objective function. The basic idea of this method is similar with that of the knowledge gradient policy, which evaluates the point that will lead to the largest expected improvement with respect to the current best solution at each simulation stage. Ahmed and Alkhamis [2002] combined the use of indifference zone with simulated annealing to improve the algorithmic efficiency. Shi and Ólafsson [2000] proposed a nested partitions method that systematically partitions the decision space and collects information to identify the most promising region. For finite-horizon Markov decision processes, Chang et al. [2005] proposed a multi-stage adaptive sampling algorithm that utilises the upper confidence bound policy to balance the exploration and exploitation when searching for optimal state-action functions. Chang et al. [2007] presented a pursuit learning automata approach that

assigns equal probabilities of obtaining evaluations to all candidate actions at the initial simulation stage and then sequentially increases the evaluation probabilities for promising candidate actions.

1.2 Structure of Thesis and Contributions

This thesis deals with some aspects related to efficient information collection in stochastic optimisation. Our focus is on sampling approaches that estimate the performance of potential solutions and guide algorithms towards the most promising solutions. The thesis structure and contributions are summarised as follows.

Chapter 2 studies the information collection problem that arises from the post-processing procedure of two-stage stochastic programs. The sampling-based stochastic programming model in the complex applications tends to be computationally challenging. Thus, a reasonable sample size needs to be identified, and a group of approximate models can then be constructed using such a number of samples. These approximate models can produce a set of potential solutions. We consider the problem of distributing a finite computational budget among numerous potential solutions for identifying the best solution. We propose a two-stage measurement strategy: Firstly, we utilise a Wasserstein-based screening rule to remove a number of inferior solutions from the simulation. The key to the success of this screening rule is the quantitative stability of Wasserstein metric in stochastic programs. We can infer an upper bound performance of potential solutions by using the Wasserstein distances between samples in the approximate models and those in the actual model. Secondly, we apply optimal computing budget allocation to determine the number of evaluations to be used for each solution. We conduct several numerical experiments to examine the performance of our approach. The numerical results indicate that the proposed approach outperforms existing approaches under relatively limited run times.

In Chapter 3, we propose new sampling strategies in evolutionary algorithms for finding the best mean solution to stochastic black-box optimisation problems. In this chapter, we focus on input uncertainty, such as in manufacturing, where the actual manufactured product may differ from the specified design but should still function well. Estimating the mean performance of a potential solution in such black-box setting is challenging, especially if the function is expensive to evaluate. One measurement might take a few weeks or possibly even a month. In this context, we implement an archive sample approximation method to reduce the required number of evaluations when estimating the mean performance. The main challenge

lies in determining the best locations of additional samples to complement existing simulation results and reduce the estimation error of mean performance. We use Wasserstein distance to estimate the possible benefit of a potential sample location on the estimation error, and propose new sampling strategies based on this metric to guide evolutionary algorithms towards the promising region of decision space. An empirical comparison with several previously proposed archive-based sample approximation methods demonstrates the superiority of our approaches.

In Chapter 4, we focus on drug development and propose an adaptive sampling strategy to support the rollout search process. Specifically, drug development is a process of testing the safety and efficacy of experimental drugs in a series of clinical trials, entrenched with a high uncertainty that an experimental drug will eventually get market approval. The profitability of drug development relies on effective strategic decision-making that comprises clinical trial scheduling and resource allocation across multiple drug projects. In this work, the drug development problem is formulated as a discrete-time finite Markov decision process, and an adaptive rollout algorithm is introduced for the curse of dimensionality arisen in the stochastic dynamic program model. The proposed algorithm includes two innovations. First, a base policy is used to obtain optimistic estimates of future outcomes after taking a particular action at a state of the system. This policy utilises a rolling horizon optimisation model to identify a specific action to be taken at each state of the system and optimistically estimates the expected profit after a particular action is implemented. We prove that the optimistic policy follows a sequential consistency property and the performance of the rollout algorithm is at least as good as the optimistic policy. Second, an adaptive sampling approach is proposed to efficiently identify the best action, which exploits the variance reduction technique of common random numbers and the empirical Bernstein inequality in a statistical racing procedure. Besides, we present an augmented adaptive sampling approach that utilises a heuristic-based grouping rule to enhance the simulation efficiency by breaking down the overall action selection problem into a selection problem involving small groups. The proposed algorithms can provide competitive results within a reasonable computational time.

Chapter 5 summarises the contributions of this thesis and describes future research directions.

Chapter 2

Efficient Solution Selection for Two-stage Stochastic Programs

2.1 Introduction

Real-life optimisation problems often involve uncertainties and require solutions that can handle such uncertainties in the modelling process. Techniques such as the two-stage linear stochastic programming with recourse (SPR) incorporate random data within the model formulation and determine a solution that satisfies the constraints and leads to the best expected objective function value for all possible scenarios. The development of SPR can be traced back to research conducted in the 1950s and 1960s, e.g., Beale [1955], Dantzig [1955] and Wets [1966]. The successful applications of SPR can be found in various sectors such as portfolio management [Dupačová, 1999; Miller and Ruszczyński, 2011], energy planning [Beraldi et al., 2008; Zhou et al., 2013; Feng and Ryan, 2014], supply chain management [Joensson et al., 1993; Santoso et al., 2005; Dillon et al., 2017], and transportation planning [Cheung and Chen, 1998; Barbarosoğlu and Arda, 2004; Liu et al., 2009].

SPR problems can become computationally intractable in numerous applications because each possible sample generated from random data is associated with one or several decision variables and constraints within the model formulation. If the sample space is considerably large or continuous, then determining an optimal solution within a reasonable timeframe will be impossible for such a model. Studies have proposed the utilisation of sample average approximation (SAA) to identify approximate solutions to large-scale SPR problems, e.g., see Gürkan et al. [1994] and Shapiro and Homem-de Mello [1998]. SAA leverages computational challenges in such a way that a subset of samples generated from random data is used to con-

struct approximate models, which provide potential solutions to the original SPR model. Monte-Carlo sampling as well as several variance reduction techniques such as quasi-Monte-Carlo sampling [Leövey and Römisch, 2015; Heitsch et al., 2016], importance sampling [Parpas et al., 2015] and Latin hyper-cube sampling [Linderoth et al., 2006] can be utilised to generate such samples. Moreover, some authors suggested the generation of samples that should satisfy a specified criterion, such as probability distances [Pflug, 2001; Dupačová et al., 2003] or moment discrepancies [Høyland et al., 2003; Gülpınar et al., 2004].

Once samples are generated, various optimisation algorithms can be used to solve the resulting SAA model. One approach is to utilise the simplex algorithm, which is conveniently implemented by modern optimisation solvers. Alternatively, some studies exploited the problem structure and proposed decomposition-based optimisation algorithms, for instance, see Dantzig and Wolfe [1960] and Van Slyke and Wets [1969]. Subsequently, numerous authors introduced advanced procedures such as the multi-cut approach [Birge and Louveaux, 1988], the trust region method [Linderoth and Wright, 2003], the regularised decomposition [Ruszczyński and Świetanowski, 1997] and the level bundle method [Wolf et al., 2014; van Ackooij et al., 2017] to improve the efficiency of utilising the decomposition principle. For a comprehensive review on decomposition approaches, the readers are referred to Rahmaniani et al. [2017].

The identification of high-quality solutions has been widely studied because of their importance for performance-sensitive SAA users. The approximate solutions can asymptotically converge to the optimal one as the number of samples gets sufficiently large, for details, see Shapiro and Homem-de Mello [1998] and Homem-de Mello and Bayraksan [2014]. Shapiro et al. [2002] showed that, given an arbitrary number of samples, each solution has a certain probability of being the optimal one, and the value is related to problem-specific factors and the number of samples. If the best solution is selected from several potential ones on the basis of their performances, then the probability of determining the optimal solution is significantly increased. Therefore, large quantities of samples and potential solutions are both important in searching for high-quality SAA solutions. However, these requirements may be difficult to satisfy simultaneously within a given period because they compete with each other on time allocations. As suggested by Lee et al. [2006], a practical remedy is to determine a proper balance between these objectives by using computing time allocation algorithms. The authors also showed that the algorithmic efficiency significantly influences performance of the final solution. The application of highly efficient optimisation approaches, such as the aforementioned

decomposition-based method, is beneficial because the SAA user can implement a large sample size to strengthen the approximation of random data and obtain improved solutions within the same timeframe. Moreover, an effective solution selection method is also important because it can promptly determine the best option among a large group of potential solutions. However, only a limited number of studies is concerned with solution selection for the SPR problems.

Defourny et al. [2013] applied a brute-force approach that runs extensive simulations for each potential solution to identify a good policy in multi-stage linear stochastic programming. Instead of individually evaluating solution performance, Kleywegt et al. [2002] used a ranking and selection approach called indifference zone to determine a good solution in two-stage stochastic discrete optimisation. The indifference zone approach, which assigns simulation replications for each potential solution on the basis of performance statistics and guarantees the overall procedure at least a certain probability of selecting the best solution, was proposed by Nelson et al. [2001]. However, this approach is not an anytime algorithm, which means that a specified simulation rule (i.e. simulation replications for each potential solution) is followed to finish all requested simulations so that a probability guarantee is achieved. Also, this method is highly conservative and usually takes many more samples than necessary [Branke et al., 2007].

In the present study, we propose a solution selection method for the large-scale two-stage linear SPR problems that can deal with numerous potential solutions and return fairly efficient solutions within a finite computational budget. The contributions of this study are threefold.

- First, a Wasserstein-based screening ($\mathcal{WS}$) approach is proposed to identify potentially promising solutions. We demonstrate that the worst-case performances of SAA solutions in the respective Wasserstein distance regions can be ranked by using the Wasserstein distance between the sampling measure used in the SAA model and the original probability measure. Solutions with small distance values have good worst-case performances and thus be classified as the most promising evaluated in the simulation.

- Second, an optimal computing budget allocation technique ($\mathcal{OCBA}$) [He et al., 2007] is used to determine how many simulation replications to use for each potential solution. The technique is an anytime algorithm, which asymptotically minimises the penalty of selecting an incorrect solution, so that the probability of achieving a good potential solution is greatly increased. We then introduce a new two-stage selection process called $\mathcal{WS}$-$\mathcal{OCBA}$, which integrates $\mathcal{OCBA}$

with $\mathcal{WS}$ to improve the simulation efficiency.

- Third, we conduct several numerical experiments to analyse performance of $\mathcal{WS}$ and the $\mathcal{WS}$-$\mathcal{OCBA}$ approaches. Results show that $\mathcal{WS}$ achieves a satisfactory trade-off between the number of potential solutions in the promising group and the performance loss. The findings also indicate that $\mathcal{WS}$-$\mathcal{OCBA}$ outperforms the existing approaches under relatively limited run times.

The remainder of this chapter is structured as follows. In Section 2.2 we provide a brief overview of two-stage stochastic programming and introduce the solution performance estimation procedure. Section 2.3 introduces the underlying principle of $\mathcal{WS}$-$\mathcal{OCBA}$. In Section 2.4, we explain the underlying principle of using Wasserstein distance in the solution screening. Section 2.5 discusses the estimation process of Wasserstein distance. Section 2.6 describes the proposed solution selection algorithm. In Section 2.7, we study the efficacy of our proposed strategies. Section 2.8 concludes this chapter by summarising our findings.

2.2 Sample Average Approximation for Two-stage Linear SPR

2.2.1 Two-stage Linear SPR Formulation

Let $\xi \in \mathbb{R}^{\mu}$ be a random vector with finite second moments. Specifically, random vector ξ is defined on the probability space $(\Xi, \mathcal{B}(\Xi), \mathbb{P})$, where Ξ is the sample space, $\mathcal{B}(\Xi)$ is the Borel sigma algebra with respect to Ξ, and $\mathbb{P} : \mathcal{B}(\Xi) \to [0, 1]$ is the probability measure. Without loss of generality, a two-stage linear SPR problem with fixed recourse can be formulated as

$$\min_{x \in \mathcal{X}} f(x) = \min_{x \in \mathcal{X}} c'x + \int_{\Xi} g(x, \xi)\, \mathbb{P}(d\xi) \tag{2.1}$$

where $c \in \mathbb{R}^{\kappa}$ is a vector of constant parameters and $\mathcal{X} \subset \mathbb{R}^{\kappa}$ represents a non-empty convex feasible set. In addition, let $g(x, \xi)$ denote the optimal value of the second-stage decision problem, formulated as follows:

$$g(x, \xi) = \min_{y \in \mathbb{R}^{\iota}} \{q'y \mid Wy = H(\xi) - T(\xi)x,\ y \geq 0\} \tag{2.2}$$

where $q \in \mathbb{R}^{\iota}$ and $W \in \mathbb{R}^{\mu \times \iota}$ are a fixed vector and a fixed matrix, respectively. Moreover, $T(\xi) \in \mathbb{R}^{\mu \times \kappa}$ and $H(\xi) \in \mathbb{R}^{\mu}$ affinely depend on random vector ξ in this study.

We also make the following assumptions throughout this study:

A(1) **Relatively complete recourse**: For each tuple $(\hat{x}, \xi)$, the corresponding second-stage decision problem (2.2) is feasible.

A(2) Dual feasibility: The set $\{\pi \mid \pi' W \leq q\}$ is not empty.

Assumption A(1) ensures the feasibility of the primal second-stage decision problem. Assumption A(2) implies dual feasibility in the second-stage decision problem. Assumptions A(1) and A(2) represent necessary conditions for the stability result of the Wasserstein metric (for detailed information, see Dupačová et al. [2003]) which will be used in our approach.

For the two-stage linear SPR problems, properties of the probability space majorly influence the computational burden. The problems, in the case of random data with continuous sample space, are rarely solvable because the resulting model formulation consists of an infinite number of second-stage decision variables and constraints. Moreover, the SPR problems might still suffer from computational intractability even when the probability distribution of random data is discrete. For instance, consider random data with 10 components, each of which follows a uniform distribution and can take 200 possible values. If we select one possible value for each component according to its distribution and then combine them as one sample, then the number of distinctive scenarios reaches 200^{10}. Since the computational complexity increases exponentially with the number of samples taken into account, the optimal solution is difficult to obtain [Shapiro and Homem-de Mello, 1998]. The SAA approach can be applied for identifying near-optimal solutions to the SPR problem.

2.2.2 The Near-optimal Solution and its Performance Estimator

Suppose that a group of samples $\hat{\Xi} = \{\hat{\xi}_m : m = 1, \ldots, M\}$ with respective probability values $\{\mathbb{Q}(\hat{\xi}_m) : m = 1, \ldots, M\}$ is generated from the random data, thereby we can obtain the following approximate model,

$$\begin{aligned} \min_{x,\ y(\hat{\xi}_m)} \quad & c'x + \sum_{m=1}^{M} \mathbb{Q}(\hat{\xi}_m) q' y(\hat{\xi}_m) \\ s.t. \quad & x \in \mathcal{X}, \\ & W y(\hat{\xi}_m) = H(\hat{\xi}_m) - T(\hat{\xi}_m) x, \quad m = 1, \ldots, M, \\ & y(\hat{\xi}_m) \geq 0, \quad m = 1, \ldots, M. \end{aligned} \tag{2.3}$$

The resulting solution is typically not the optimal solution for the original SPR model, so it is important to evaluate its performance in the original model.

Let $\hat{x}$ and $f(\hat{x})$ denote a potential solution and its performance in the original model, respectively. Mak et al. [1999] suggested using Monte-Carlo estimation to infer the value $f(\hat{x})$. Assume that we have K *i.i.d* batches of samples with size N_E and equal probabilities; that is, $\tilde{\Xi}_k = \{\tilde{\xi}_n^k : n = 1, \ldots, N_E\}$, for $k = 1, \ldots, K$. We can estimate a true solution performance for each batch k of samples by computing the optimal value $\hat{f}_{N_E}^k(\hat{x})$:

$$\begin{aligned} \hat{f}_{N_E}^k(\hat{x}) = \frac{1}{N_E} \sum_{n=1}^{N_E} \quad & \min_{y(\tilde{\xi}_n^k)} \left[c'\hat{x} + q'y(\tilde{\xi}_n^k) \right] \\ & s.t. \;\; Wy(\tilde{\xi}_n^k) = H(\tilde{\xi}_n^k) - T(\tilde{\xi}_n^k)\hat{x} \\ & \qquad y(\tilde{\xi}_n^k) \geq 0. \end{aligned} \tag{2.4}$$

By averaging over all optimal values $\hat{f}_{N_E}^k(\hat{x})$, for $k = 1, \ldots, K$, we obtain an estimator of $f(\hat{x})$ as

$$\mathcal{J}(\hat{x}) = \frac{1}{K} \sum_{k=1}^{K} \hat{f}_{N_E}^k(\hat{x}). \tag{2.5}$$

Let $\sigma^2(\hat{f}_{N_E}(\hat{x}))$ denote the unknown population variance of the optimal values of SAA models with the first-stage decision $\hat{x}$ and sample size N_E. Mak et al. [1999] demonstrated that $\mathcal{J}(\hat{x})$ is an unbiased estimator and follows the Central Limit Theorem:

$$\sqrt{K}\Big[\mathcal{J}(\hat{x}) - f(\hat{x})\Big] = \sqrt{K}\left[\frac{1}{K}\sum_{k=1}^{K} \hat{f}_{N_E}^k(\hat{x}) - f(\hat{x})\right] \rightarrow \mathcal{N}\Big(0, \sigma^2(\hat{f}_{N_E}(\hat{x}))\Big), \tag{2.6}$$

when $K \rightarrow \infty$. Note that $\mathcal{N}\Big(0, \sigma^2(\hat{f}_{N_E}(\hat{x}))\Big)$ is a Gaussian distribution with variance $\sigma^2(\hat{f}_{N_E}(\hat{x}))$ and zero mean. Then, the population variance $\sigma^2(\hat{f}_{N_E}(\hat{x}))$ can be estimated by using the following estimator as

$$\mathcal{V}(\hat{x}) = \sum_{k=1}^{K} \frac{\left[\hat{f}_{N_E}^k(\hat{x}) - \mathcal{J}(\hat{x})\right]^2}{K-1}. \tag{2.7}$$

In the SAA framework, the performance of an approximate solution depends on the number of samples used to represent the random data; therefore, a sufficient number of samples should be included in the model formulation. Moreover, rather than focusing on only one solution, considering multiple potential solutions can be

also beneficial for finding the optimal solution. In the following subsection, we will introduce the solution selection problem.

2.2.3 Solution Selection Under a Fixed Computing Budget

Suppose that a group of potential solutions is given and a simulation is required to select the best one as the final solution under the fixed computational budget. One of the challenges we might encounter is that the "best" solution based on Monte-Carlo estimation may not be really the best solution if insufficient information is available for analysing the solution performance. In this study, we model the solution selection process as a computing time allocation problem. Let $\{\hat{x}^\lambda : \lambda = 1, \ldots, \Lambda\}$ denote a set of potential solutions. The performance of each solution $\hat{x}^\lambda$ is evaluated by K^λ batch samples with size N_E. The CPU time to compute each batch sample in (2.4) is denoted by $t(\hat{x}^\lambda, N_E)$. Let the batch number K^λ for $\lambda = 1, \ldots, \Lambda$ represent unknown decision variables. Moreover, we define $\hat{x}^s$ as the solution with best sample mean. Given the total simulation budget T^{total}, the computing time allocation problem can be formulated as follows:

$$\begin{aligned} \min_{K^1, K^2, \ldots, K^\Lambda} \quad & f(\hat{x}^s) \\ s.t. \quad & \sum_{\lambda=1}^{\Lambda} t(\hat{x}^\lambda, N_E) K^\lambda \leq T^{total} \\ & \hat{x}^s = \arg\min \left\{ \frac{1}{K^\lambda} \sum_{k=1}^{K^\lambda} \hat{f}^k_{N_E}(\hat{x}) : \lambda = 1, \ldots, \Lambda \right\} \\ & K^\lambda \in \mathbb{N}, \quad \lambda = 1, \ldots, \Lambda. \end{aligned} \tag{2.8}$$

It is challenging to solve the solution selection problem in the sense that its objective function represents true performance of potential solutions.

2.3 Optimal Computing Budget Allocation

The $\mathcal{OCBA}$ technique is a class of heuristic methods for solving the solution selection problem. We use an $\mathcal{OCBA}$ variant in which the correct selection evidence is defined as expected opportunity cost [He et al., 2007]. Let us denote $\hat{x}^b = \arg\max\{f(\hat{x}^\lambda) : \lambda = 1, \ldots, \Lambda\}$ as the true best solution. Then, the expected opportunity cost $\mathbb{E}(\mathrm{OC})$

quantifies the penalty due to wrong selection and is defined as follows:

$$\mathbb{E}(\mathrm{OC}) = \mathbb{E}\Big[f(\hat{x}^s) - f(\hat{x}^b)\Big] = \sum_{\lambda=1,\ \lambda\neq s}^{\Lambda} Prob(\hat{x}^\lambda = \hat{x}^b)\Big[f(\hat{x}^s) - f(\hat{x}^\lambda)\Big], \tag{2.9}$$

where $Prob(\hat{x}^\lambda = \hat{x}^b)$ denotes the probability of solution $\hat{x}^\lambda$ being the true best solution $\hat{x}^b$.

As $Prob(\hat{x}^\lambda = \hat{x}^b)$ is unknown in practice, He et al. [2007] proposed to use an upper bound approximation of $\mathbb{E}(\mathrm{OC})$ that can be estimated during the simulation procedure.

Theorem 2.1. Let $\phi(\cdot)$ and $\Phi(\cdot)$ be the probability density function (PDF) and cumulative distribution function of standard normal distribution, respectively. Moreover, K^s is the number of evaluated samples for the solution $\hat{x}^s$ and K^λ is the number of evaluated samples for solution $\hat{x}^\lambda$. The upper bound approximation of expected opportunity cost (abbreviated as AEOC) is presented as follows:

$$\mathbb{E}(\mathrm{OC}) \leq AEOC = \sum_{\lambda=1,\lambda\neq s}^{\Lambda} \Big\{\mathcal{V}_{s,\lambda} \times \phi(z_{s,\lambda}) + \delta_{s,\lambda} \times \Phi(-z_{s,\lambda})\Big\} \tag{2.10}$$

where $\mathcal{V}_{s,\lambda} = \frac{\mathcal{V}(\hat{x}^s)}{K^s} + \frac{\mathcal{V}(\hat{x}^\lambda)}{K^\lambda}$, $\delta_{s,\lambda} = \mathcal{J}(\hat{x}^s) - \mathcal{J}(\hat{x}^\lambda)$ and $z_{s,\lambda} = \frac{-\delta_{s,\lambda}}{\sqrt{\mathcal{V}_{s,\lambda}}}$. Note that $\mathcal{V}(\hat{x}^\lambda)$ and $\mathcal{V}(\hat{x}^s)$ are the estimation variances of solution $\hat{x}^\lambda$ and $\hat{x}^s$, which can be computed by Equation (2.7).

Proof. He et al. [2007] provided an upper bound approximation for the probability $Prob(\hat{x}^\lambda = \hat{x}^b)$ as follows,

$$Prob(\hat{x}^\lambda = \hat{x}^b) \leq Prob\Big(f(\hat{x}^\lambda) < f(\hat{x}^s)\Big). \tag{2.11}$$

Hence, we can obtain

$$\begin{aligned}\mathbb{E}(\mathrm{OC}) &\leq \sum_{\lambda=1,\lambda\neq s}^{\Lambda} Prob\Big(f(\hat{x}^\lambda) < f(\hat{x}^s)\Big)\Big[f(\hat{x}^s) - f(\hat{x}^\lambda)\Big] \\ &= \sum_{\lambda=1,\lambda\neq s}^{\Lambda} \int_0^{+\infty} t\ \eta_{s,\lambda}(t)\ dt = \mathrm{AEOC},\end{aligned} \tag{2.12}$$

where $\eta_{s,\lambda}$ denotes the PDF of random value $\mathcal{N}\Big(f(\hat{x}^s) - f(\hat{x}^\lambda), \mathcal{V}_{s,\lambda}\Big)$. As shown in

He et al. [2007], we can apply the integration by parts method to compute (2.12) as follows,

$$\int_0^{+\infty} t\ \eta_{s,\lambda}(t)\ dt = \mathcal{V}_{s,\lambda}\phi(z_{s,\lambda}) + \delta_{s,\lambda}\Phi(-z_{s,\lambda}). \tag{2.13}$$

□

Given a finite computing budget, we aim to allocate the simulation budget to sequentially minimise the upper bound of expected opportunity cost. If we evaluate one additional sample for the solution $\hat{x}^\lambda$, then the upper bound of expected opportunity cost will change to

$$\widehat{\mathrm{AEOC}}_\lambda = \sum_{\lambda'=1,\lambda'\neq s}^{\Lambda} \int_0^{\infty} x\ \eta_{s,\lambda',\lambda}\ dx,\ \lambda = 1,\ldots,\Lambda, \tag{2.14}$$

where $\eta_{s,\lambda',\lambda}$ is the PDF of the normally distributed random variable and defined as

$$\begin{cases} \mathcal{N}\left(f(\hat{x}^s) - f(\hat{x}^\lambda), \frac{\mathcal{V}(\hat{x}^s)}{K^s+1} + \frac{\mathcal{V}(\hat{x}^{\lambda'})}{K^{\lambda'}}\right), & \text{if } \hat{x}^\lambda = \hat{x}^s \\ \mathcal{N}\left(f(\hat{x}^s) - f(\hat{x}^\lambda), \frac{\mathcal{V}(\hat{x}^s)}{K^s} + \frac{\mathcal{V}(\hat{x}^{\lambda'})}{K^{\lambda'}+1}\right), & \text{if } \hat{x}^\lambda = \hat{x}^{\lambda'} \\ \mathcal{N}\left(f(\hat{x}^s) - f(\hat{x}^\lambda), \frac{\mathcal{V}(\hat{x}^s)}{K^s} + \frac{\mathcal{V}(\hat{x}^{\lambda'})}{K^{\lambda'}}\right), & \text{if } \hat{x}^\lambda \neq \hat{x}^s \text{ and } \hat{x}^\lambda \neq \hat{x}^{\lambda'}. \end{cases}$$

The above integration can be computed by using (2.10). Then, the possible reduction of AEOC can be computed as,

$$\mathcal{Y}_\lambda = \mathrm{AEOC} - \widehat{\mathrm{AEOC}}_\lambda \geq 0. \tag{2.15}$$

Notice that the value $\mathcal{Y}_\lambda$ for $\lambda = 1,\ldots,\Lambda$ is always non-negative because the $\mathcal{OCBA}$ procedure aims to greedily reduce $\mathbb{E}(\mathrm{OC})$ by sampling the performances of SAA solutions. In other words, $\mathcal{Y}_\lambda$ represents the estimated improvement on $\mathbb{E}(\mathrm{OC})$ when evaluating the performance of SAA solution $\hat{x}^\lambda$. If the improvement for a SAA solution is negative, then we will stop the sampling procedure. Next, the sample would be assigned to the solution that leads to the maximum reduction of AEOC, i.e.,

$$\lambda^* = \arg\max\{\mathcal{Y}_\lambda :\ \lambda = 1,\ldots,\Lambda\}. \tag{2.16}$$

The overall procedure is described in Algorithm 2.1. The $\mathcal{OCBA}$ technique can reduce the overall simulation budget necessary to identify the best solution. However, the $\mathcal{OCBA}$ procedure might fail to identify the best solution given abundant

potential solutions. For instance, if the limited computing budget runs out during the initial estimation of Algorithm 2, then some solutions will not be evaluated. If the best solution is one of those ignored, then it is not possible to correctly identify the best solution. Moreover, the solution selection method is challenging when the computing budget is insufficient in the sequential decision process because the initial estimation has taken a large proportion of the computing budget. Then insufficient estimation might mislead our choice of the final solution.

Algorithm 2.1: The $\mathcal{OCBA}$ Procedure

input : potential solutions: $\{\hat{x}^\lambda : \lambda = 1, \ldots, \Lambda\}$;
size of sample N_E in the performance estimation;
number of samples K_0 evaluated in the initial estimation.
output: best solution based on the simulation results.

```
while simulation budget is available do
    Step 1: Initial Estimation;
    for λ = 1, ..., Λ do
        evaluate the performance of solution x̂^λ using K_0 samples;
        compute the performance statistics of solution x̂^λ using (2.6) and (2.7);
        select the current best solution x̂^s;
    Step 2: Sequential Decision Process;
    for λ = 1, ..., Λ do
        compute the expected opportunity cost reduction Y_λ as in (2.10) and (2.14);
    find λ* = argmax{Y_λ, λ = 1, ..., Λ};
    simulate one additional sample for solution x̂^{λ*} and update its statistics;
select the current best solution x̂^s.
```

2.4 Wasserstein-based Solution Screening

When the number of potential solutions is large and the computing budget is insufficient to perform an initial simulation for every solution in the $\mathcal{OCBA}$ procedure, it is natural to consider performing the simulation for a certain portion of potential solutions. An important question is *how to select a set of promising solutions for the extensive simulation.* In this paper, we propose to select those solutions through the Wasserstein distance due to its low computational cost that allows it to be run for every potential solution. Specifically, the potential solutions are sorted according to their Wasserstein distances, and only the high-rank solutions are selected for

extensive simulation. A similar paradigm called "ordinal transformation" is used in a simulation optimisation study [Xu et al., 2016], wherein the authors considered the simulation output on a user-defined low-fidelity model as the low-cost measure for each potential solution. The potential solutions were clustered according to low-fidelity simulation results and the extensive simulation was applied to select the best solution cluster.

The Wasserstein distance metric, a kind of statistical metric for quantifying the dissimilarity between two probability measures, is the key component of $\mathcal{WS}$ method. The definition of Wasserstein metric can be found in Appendix A. This metric has been widely applied in stochastic programming. One application is scenario reduction wherein the Wasserstein distance is used as the quality indicator of samples in the approximate model. Given a fixed number of samples used in the approximate model, some authors proposed heuristics to select the "best" samples with minimum distance. For example, Dupačová et al. [2003] presented two myopic scenario reduction heuristics for two-stage SPR, namely, forward selection and backward reduction. Furthermore, Heitsch and Römisch [2009] extended these heuristics for multi-stage SPR. In another application, the Wasserstein distance was used to define an ambiguity set for stochastic programs with distributional uncertainty [Mohajerin Esfahani and Kuhn, 2017]. In addition, the Wasserstein distance was applied to reduce the optimality gap estimator bias, and this application benefits testing the optimality of a given solution [Stockbridge and Bayraksan, 2013]. In this study, we use the Wasserstein distance metric to roughly rank the performance of potential solutions.

Before stating the main result, we first introduce the stability result of stochastic programming.

Theorem 2.2. Assume that the solution sets of original model and SAA model are singleton. Let x^* and $\hat{x}$ denote the unique solutions of the original model and SAA model, respectively. Moreover, let $\mathbb{P}$ be the original probability measure, and $\mathbb{Q}$ represent the sampling measure of a SAA model. The difference between measures $\mathbb{P}$ and $\mathbb{Q}$ is quantified by the Wasserstein distance $W(\mathbb{P}, \mathbb{Q})$. Under assumptions A(1) and A(2), there exit constants $\tau > 0$ and $\varepsilon > 0$ such that

$$\|x^* - \hat{x}\|_2 \leq \frac{2\tau}{\varepsilon} W(\mathbb{P}, \mathbb{Q}).$$

Proof. For details, see the proof of Theorem 1 introduced by Dupačová et al. [2003]. □

Theorem 2.2 utilises two constants τ and ε to establish the Lipschitz-like property of SAA solutions with respect to changes in the probability measure. Next, we will show the relationship between the solution performance and the corresponding Wasserstein distance.

Theorem 2.3. Under assumptions A(1) and A(2), the performance difference between solutions x^* and $\hat{x}$ satisfies the following inequality:

$$f(\hat{x}) \leq f(x^*) + \langle |\mathcal{L}_{\hat{x}}|, \hat{\tau} W(\mathbb{P}, \mathbb{Q}) I \rangle, \tag{2.17}$$

where $\mathcal{L}_{\hat{x}} = \mathbb{E}[\pi_{\hat{x}}(\xi) T(\xi)] + c$ with $\pi_{\hat{x}}(\xi) = \operatorname{argmax}\{\pi'[H(\xi) - T(\xi)\hat{x}] \ : \ \pi' W \leq q\}$, $\hat{\tau} = \frac{2\tau}{\varepsilon}$ is a positive coefficient, and $I \in \mathbb{R}^{\kappa}$ is the all one vector.

Proof. Consider the two-stage SPR problem given in (2.1). Under assumptions A(1) and A(2), the Proposition 2.3 introduced by Shapiro et al. [2009] proved that, if function $g(\hat{x}, \xi)$ for $\hat{x} \in \mathcal{X}, \xi \in \Xi$ is finite, then the objective function $f(\cdot)$ will be sub-differentiable at the point $\hat{x}$. Let $\mathcal{L}_{\hat{x}}$ denote one of sub-gradients for solution $\hat{x}$. The following sub-gradient inequality holds,

$$f(x^*) \geq f(\hat{x}) + \langle \mathcal{L}_{\hat{x}}, x^* - \hat{x} \rangle. \tag{2.18}$$

Shapiro et al. [2009] showed that the sub-gradient $\mathcal{L}_{\hat{x}}$ of a two-stage SPR problem can be calculated as

$$\mathcal{L}_{\hat{x}} = \mathbb{E}[\pi'_{\hat{x}}(\xi) T(\xi)] + c,$$

where $\pi_{\hat{x}}(\xi)$ represent a vector of dual decisions associated with constraints of the second-stage problem (2.2). It is computed as follows,

$$\pi_{\hat{x}}(\xi) = \arg\max_{\pi} \{\pi'[H(\xi) - T(\xi)\hat{x}] \ : \ \pi' W \leq q\}.$$

By substituting the sub-gradient and re-arranging both sides of inequality (2.18), we obtain

$$f(\hat{x}) - f(x^*) \leq -\langle \mathbb{E}[\pi'_{\hat{x}}(\xi) T(\xi)] + c, x^* - \hat{x} \rangle. \tag{2.19}$$

Since $\hat{x}$ is a potential solution, $f(\hat{x})$ cannot be smaller than $f(x^*)$ for the minimisa-

tion problem. Therefore, we can obtain

$$|f(\hat{x}) - f(x^*)| \leq \langle |\mathbb{E}[\pi'_{\hat{x}}(\xi)T(\xi)] + c|, |x^* - \hat{x}| \rangle. \tag{2.20}$$

Using Theorem 2.2, the distance between solutions x^* and $\hat{x}$ is upper bounded by the Wasserstein distance $W(\mathbb{P}, \mathbb{Q})$ as follows,

$$\|x^* - \hat{x}\|_2 = \sqrt{\sum_{\kappa'=1}^{\kappa} (x^*_{\kappa'} - \hat{x}_{\kappa'})^2} \leq \frac{2\tau}{\varepsilon} W(\mathbb{P}, \mathbb{Q}) \leq \hat{\tau} W(\mathbb{P}, \mathbb{Q}) \tag{2.21}$$

where $\hat{\tau} = \frac{2\tau}{\varepsilon}$ is a positive coefficient. For each element of solutions x^* and $\hat{x}$, we can write

$$\hat{\tau} W(\mathbb{P}, \mathbb{Q}) \geq \sqrt{\sum_{\kappa'=1}^{\kappa} (x^*_{\kappa'} - \hat{x}_{\kappa'})^2} \geq \sqrt{(x^*_{\kappa'} - \hat{x}_{\kappa'})^2} = |x^*_{\kappa'} - \hat{x}_{\kappa'}|, \ \kappa' = 1, \ldots, \kappa. \tag{2.22}$$

Therefore, $|x^* - \hat{x}|$ can be approximated by using the Wasserstein distance, $W(\mathbb{P}, \mathbb{Q})$, as follows,

$$|x^* - \hat{x}| \leq \hat{\tau} W(\mathbb{P}, \mathbb{Q}) I. \tag{2.23}$$

By combining inequality (2.23) with (2.19), we obtain (2.17). □

Theorem 2.3 implies that performance of potential solution $\hat{x}$ is bounded by the sub-gradient $\mathcal{L}_{\hat{x}}$ and the Wasserstein distance $W(\mathbb{P}, \mathbb{Q})$. The potential solution $\hat{x}$ becomes the optimal solution x^* for the actual problem if either probability measures $\mathbb{Q}$ and $\mathbb{P}$ are identical or the sub-gradient at $\hat{x}$ becomes zero.

As mentioned before, the Wasserstein distance $W(\mathbb{P}, \mathbb{Q})$ can be efficiently estimated using Monte-Carlo estimation; however, the calculation of sub-gradients $\mathcal{L}_{\hat{x}}$ for two-stage liner SPRs is computationally expensive. Next, we introduce the worst-case solution performance in the Wasserstein-bounded region. We will use this performance measure in the solution screening procedure.

Definition 2.1: Worst-case Solution Performance in the Wasserstein Region. Let $\Gamma\big(\hat{x}(\mathbb{Q})\big)$ represent a set possessing sub-gradients of all feasible solutions within the bounded region $\{x : |x - x^*| \leq \hat{\tau} W(\mathbb{P}, \mathbb{Q}) I\}$. The worst-case performance $\mathcal{G}^w(\hat{x})$ of a given solution $\hat{x}$ within its corresponding Wasserstein-bounded region can

be determined as follows:

$$\mathcal{G}^w(\hat{x}) = \max_{\mathcal{L}' \in \Gamma\left(\hat{x}(\mathbb{Q})\right)} \left\{ f(x^*) + \left\langle \left| \mathcal{L}' \right|, \hat{\tau} W(\mathbb{P}, \mathbb{Q}) I \right\rangle \right\}. \tag{2.24}$$

Notice that $\mathcal{G}^w(\hat{x})$ is defined as a maximisation problem with respect to the sub-gradient value $\mathcal{L}'$. The next theorem states the applicability of the Wasserstein distance in sequencing the worst-case performance of potential solutions.

Theorem 2.4. Consider a set of potential solutions $\{x^\lambda : \lambda = 1, \ldots, \Lambda\}$ with respective Wasserstein distances $\{W(\mathbb{P}, \mathbb{Q}^\lambda) : \lambda = 1, \ldots, \Lambda\}$. Let $[\lambda]$ denote the λ-th potential solution in the increasing sequence of Wasserstein distances as

$$W(\mathbb{P}, \mathbb{Q}^{[1]}) \leq W(\mathbb{P}, \mathbb{Q}^{[2]}) \leq \cdots \leq W(\mathbb{P}, \mathbb{Q}^{[\Lambda]}). \tag{2.25}$$

Then, the worst-case solution performances of these solutions satisfy,

$$\mathcal{G}^w(\hat{x}^{[1]}) \leq \mathcal{G}^w(\hat{x}^{[2]}) \leq \cdots \leq \mathcal{G}^w(\hat{x}^{[\Lambda]}). \tag{2.26}$$

Proof. Assume that the Wasserstein distances satisfy the sequence as in (2.25). Since the inequality (2.23) can be written for all potential solutions $\hat{x}^\lambda$, we can construct the same relationship as in sequence of

$$\begin{aligned} &\left\{ x : |x - x^*| \leq \hat{\tau} W(\mathbb{P}, \mathbb{Q}^{[1]}) I \right\} \subseteq \left\{ x : |x - x^*| \leq \hat{\tau} W(\mathbb{P}, \mathbb{Q}^{[2]}) I \right\} \\ &\subseteq \cdots \subseteq \left\{ x : |x - x^*| \leq \hat{\tau} W(\mathbb{P}, \mathbb{Q}^{[\Lambda]}) I \right\}. \end{aligned} \tag{2.27}$$

Hence, the feasibility set of all sub-gradients at approximate potential solutions holds the sequence of

$$\Gamma\left(\hat{x}^{[1]}(\mathbb{Q})\right) \subseteq \Gamma\left(\hat{x}^{[2]}(\mathbb{Q})\right) \subseteq \cdots \subseteq \Gamma\left(\hat{x}^{[\Lambda]}(\mathbb{Q})\right). \tag{2.28}$$

From (2.25) and (2.28), one can say that the solution with a smaller Wasserstein distance leads to a smaller feasibility set and a small coefficient vector in the optimisation problem (2.24). Hence, the same sequence order also holds for the worst-case performances at potential solutions as stated in (2.26). □

Theorem 2.4 indicates that the sequence of Wasserstein distances for SAA solutions encapsulates the trend of the worst-case solution performances. We should note that the rank of actual solution performances in general does not follow the sequence of Wasserstein distances. Therefore, when the solution screening is applied according to the sequence of Wasserstein distances, a performance loss that is caused by eliminating the best solution might arise. We describe the performance loss as follows.

Definition 2.2: Performance Loss. Let $\{\hat{x}^{\lambda'} : \lambda' = 1, \ldots, \Lambda^{\mathcal{P}}\}$ denote a set of promising solutions obtained from a specific screening procedure. If the computing budget is restricted on those promising solutions, then the performance loss (PL) due to screening out the best solution can be computed as

$$PL = \min\{f(\hat{x}^{\lambda'}) : \lambda' = 1, \ldots, \Lambda^{\mathcal{P}}\} - \min\{f(\hat{x}^{\lambda}) : \lambda = 1, \ldots, \Lambda\}. \tag{2.29}$$

If $PL = 0$, then the promising group contains the best solution. Otherwise, PL is always greater than zero. The value PL reflects the quality of the promising group; thus, having a screening procedure that has a performance guarantee is desirable. Next, we prove that the proposed screening approach provides an upper bound for the performance loss.

Theorem 2.5. Assume that for a set of potential solutions $\{\hat{x}^{\lambda} : \lambda = 1, \ldots, \Lambda\}$, the corresponding set of Wasserstein distances $\{W(\mathbb{P}, \mathbb{Q}^{\lambda}) : \lambda = 1, \ldots, \Lambda\}$ possesses an increasing sequence of distance values. In other words, the following inequalities hold:

$$W(\mathbb{P}, \mathbb{Q}^{[1]}) \leq W(\mathbb{P}, \mathbb{Q}^{[2]}) \leq \cdots \leq W(\mathbb{P}, \mathbb{Q}^{[\Lambda]}). \tag{2.30}$$

Then, the Wasserstein-based screening provides the following upper bound for the performance loss:

$$PL \leq \mathcal{G}^{w}(\hat{x}^{[1]}) - \min\{f(\hat{x}^{\lambda}) : \lambda = 1, \ldots, \Lambda\}. \tag{2.31}$$

Proof. From (2.17) and (2.24), we can write the following inequality for each promising solution

$$f(\hat{x}^{\lambda'}) \leq \mathcal{G}^{w}(\hat{x}^{\lambda'}), \quad \forall \lambda' = 1, \ldots, \Lambda^{\mathcal{P}}. \tag{2.32}$$

Hence, from the definition of performance loss, it follows:

$$\min\{f(\hat{x}^{\lambda'}) : \lambda' = 1, \ldots, \Lambda^{\mathcal{P}}\} \leq \min\{\mathcal{G}^w(\hat{x}^{\lambda'}) : \lambda' = 1, \ldots, \Lambda^{\mathcal{P}}\}. \tag{2.33}$$

This yields

$$\begin{aligned} PL &= \min\{f(\hat{x}^{\lambda'}) : \lambda' = 1, \ldots, \Lambda^{\mathcal{P}}\} - \min\{f(\hat{x}^{\lambda}) : \lambda = 1, \ldots, \Lambda\} \\ &\leq \min\{\mathcal{G}^w(\hat{x}^{\lambda'}) : \lambda' = 1, \ldots, \Lambda^{\mathcal{P}}\} - \min\{f(\hat{x}^{\lambda}) : \lambda = 1, \ldots, \Lambda\}. \end{aligned} \tag{2.34}$$

Clearly, selecting a promising subgroup of solutions out of the top $\Lambda^{\mathcal{P}}$ of the lowest Wasserstein distances (using Theorem 2.3) provides

$$PL \leq \mathcal{G}^w(\hat{x}^{[1]}) - \min\{f(\hat{x}^{\lambda}) : \lambda = 1, \ldots, \Lambda\}. \tag{2.35}$$

So by combining with (2.34), we find that (2.31) holds. □

Theorem 2.5 implies that the Wasserstein-based screening provides a fixed upper bound for the performance loss even without performing any simulation. The tightness of the bound depends on the stability result of the Wasserstein distance.

2.5 Wasserstein Distance Estimation

In this section, we introduce a method of estimating Wasserstein distance in the two-stage linear SPR problems. Let N denote the number of samples for original probability measure, and M be the number of samples in the sampling measure. According to the definition of Wasserstein distance in Appendix A, the transportation problem involves $N \times M$ decision variables and $N + M + N \times M$ linear constraints. If the value of N is very large or even infinite, then the corresponding computational complexity will be intractable. In the context, we can replace the probability measure $\mathbb{P}$ with a group of sampling measures, and take the average Wasserstein distance between random measures and probability measure $\mathbb{Q}$ as the estimator of $W(\mathbb{P}, \mathbb{Q})$. The following Lemma states that the bias of such an estimator is bounded.

Lemma 2.1. Let P^i for $i = 1, \ldots, \mathcal{I}$ denote the sampling measure induced by N_W realisations generated from the probability measure $\mathbb{P}$. If we use the estimator

$$\hat{W}(\mathbb{P}, \mathbb{Q}) = \frac{1}{\mathcal{I}} \sum_{i=1}^{\mathcal{I}} W(P^i, \mathbb{Q})$$

to infer the Wasserstein distance value $W(\mathbb{P}, \mathbb{Q})$, then its bias satisfies the following inequality,

$$\left|W(\mathbb{P}, \mathbb{Q}) - \hat{W}(\mathbb{P}, \mathbb{Q})\right| \leq \frac{1}{\mathcal{I}} \sum_{i=1}^{\mathcal{I}} W(P^i, \mathbb{P}). \tag{2.36}$$

Proof. Since the Wasserstein distance is a metric, it satisfies the reverse triangle inequality:

$$|W(\mathbb{P}, \mathbb{Q}) - W(P^i, \mathbb{Q})| \leq W(P^i, \mathbb{P}). \tag{2.37}$$

Next, we can compute the sum of inequalities (2.37) over all P^i for $i \in \mathcal{I}$ as follows;

$$-\sum_{i=1}^{\mathcal{I}} W(P^i, \mathbb{P}) \leq \mathcal{I} \times W(\mathbb{P}, \mathbb{Q}) - \sum_{i=1}^{\mathcal{I}} W(P^i, \mathbb{Q}) \leq \sum_{i=1}^{\mathcal{I}} W(P^i, \mathbb{P}). \tag{2.38}$$

By dividing both sides of the above inequality by $\mathcal{I}$, we obtain inequality (2.36). □

Lemma 2.1 states that the absolute value of estimation bias is bounded by the average Wasserstein distance. We can increase the sample size of the sampling measure to minimise the fluctuation of bias. The main benefit of using Lemma 2.1 is to reduce the computational burden due to the large sample number. We can now solve a group of relatively small optimisation problems to infer the actual Wasserstein distance $W(\mathbb{P}, \mathbb{Q})$. Each "small" optimisation problem only has $N_W \times M$ decision variables and $N_W + M + N_W \times M$ constraints. The overall Wasserstein distance estimation procedure is described in Algorithm 2.2.

Algorithm 2.2: Wasserstein Distance Estimation Procedure

input : number of sampling measures $\mathcal{I}$;
number of samples in each sampling measure N_W.
output: estimated Wasserstein distance value $\hat{W}(\mathbb{P}, \mathbb{Q})$.

1 generate $\mathcal{I}$ groups of i.i.d N_W realisations from probability measure $\mathbb{P}$;
2 **for** $i = 1, \ldots, \mathcal{I}$ **do**
3 | compute the Wasserstein distance $W(P^i, \mathbb{Q})$ using (A.1);
4 **end**
5 $\hat{W}(\mathbb{P}, \mathbb{Q}) \leftarrow \frac{1}{\mathcal{I}} \sum_{i=1}^{\mathcal{I}} W(P^i, \mathbb{Q})$.

2.6 The Proposed Two-stage Selection Approach

We propose a two-stage method that combines the Wasserstein-based screening method with the optimal computing budget allocation. Algorithm 2.3 describes integration of the solution screening approach into $\mathcal{OCBA}$. The Wasserstein-based solution screening method selects potential solutions that are then more closely examined using $\mathcal{OCBA}$. The solution screening procedure takes a proportion of the computing budget. Thus, it is necessary to properly allot the computing budget among different procedures; namely, solution screening, initial estimation, and sequential allocation. We now highlight some rules for a practical application of Algorithm 2.3.

Algorithm 2.3: The $\mathcal{WS}$-$\mathcal{OCBA}$ Procedure

input : candidate solutions: $\{\hat{x}^\lambda : \lambda = 1, \cdots, \Lambda\}$;
size of sample N_E in the performance estimation;
number of samples $\mathcal{I}$ with size N_W in Wasserstein distance;
number of promising solutions: $\Lambda^{\mathcal{P}}$;
number of samples K_0 evaluated in the initial estimation.

output: best solution based on the simulation results.

1 **for** $\lambda = 1, \cdots, \Lambda$ **do**
2 run Algorithm 2.2 for solution x^λ to estimate the corresponding Wasserstein distance;
3 rank and select the top $\Lambda^{\mathcal{P}}$ candidate solutions according to their Wasserstein distances;
4 **while** *computing budget is available* **do**
5 implement the evaluation procedure as in Algorithm 2.1.

As shown in Algorithm 2.3, the Wasserstein-based solution screening selects candidates that are then more closely examined using the $\mathcal{OCBA}$ algorithm. The solution screening procedure takes a proportion of computing budget. Thus, it is necessary to properly allot the computing budget for the procedures, namely, solution screening, initial estimation, and sequential decision. We now highlight the rules for the practical application of Algorithm 2.3.

- A proper computing budget allocation is important for $\mathcal{WS}$ and $\mathcal{OCBA}$. The solution screening process provides the sequence of potential solutions based on their worst-case performances in the corresponding Wasserstein-bounded region whereas the actual performance is exploited in $\mathcal{OCBA}$. Therefore, we should allocate the computing budget to $\mathcal{OCBA}$ such that there are sufficient simulation replications to guarantee the accuracy of the selection procedure.

Table 2.1: Complexity of benchmark problems

Problem Instances	D/C		Dim	NS	Optimal Values
	$1st$ stage	$2nd$ stage			
LandS	4/2	12/7	3	1×10^6	225.620±0.020
Retail	7/0	70/22	7	1×10^{11}	154.410±0.770
20term	64/3	764/124	40	1.1×10^{12}	254298.572±38.743
SSN	89/1	706/175	86	1.1×10^{70}	9.840±0.100

- A large sample size is preferred for the Wasserstein distance estimation because it helps to reduce potential estimation bias; however, the computational cost of the Wasserstein distance estimation should be controlled to a certain level, which aims to secure sufficient computing budget in $\mathcal{OCBA}$. The rule of thumb is that, for a single solution, the computational cost of the Wasserstein distance estimation should be smaller than that of the initial estimation in Algorithm 2.1, because the Wasserstein distance estimation is designed to be a fast simulator. Otherwise, it might be wise to use the initial simulation results for eliminating the inferior actions.

- The number of potential solutions to be run for simulation (i.e., promising solutions) is determined by the computing budget to be allocated for $\mathcal{OCBA}$. We should guarantee to have sufficient computing budget for Step 2 of Algorithm 2.1. Otherwise, the insufficient simulation may lead to selection errors.

2.7 Numerical Experiments

2.7.1 Overview of Test Instances

In this section, we describe four benchmark problem instances to be used for the numerical experiments. Table 2.1 presents complexity of these instances in terms of number of decision variables (D), constraints (C), dimension of uncertainties (Dim), number of scenarios (NS) as well as the optimal values (mean ± standard error). The corresponding problem descriptions are briefly summarised as below:

- **LandS** is an electricity planning problem [Louveaux and Smeers, 1988]. The first-stage decisions are concerned with an allocation of four power terminals, and the second-stage decisions are related to allocating the power supply to various residential areas.

- **Retail** is taken from Herer et al. [2006], which is a supply chain optimisation

problem involving multiple retailers and one supplier. The objective is to design the optimal replenishment policy for each retailer.

- **20term**, adopted from Linderoth et al. [2006], is a large-scale vehicle management problem. The first-stage decisions find the vehicle locations at the beginning of the plan, and the second-stage decisions optimise the fleet transportation plan on the basis of the initial vehicle locations.

- **SSN** [Sen et al., 1994] aims to design an optimal telecommunication network that can minimise the number of lost demands. In view of randomness of the communication demand in the network, the bandwidth between destination and origin nodes must be sufficiently large to satisfy the customer demand. Otherwise, that demand will be lost.

All test instances are minimisation problems, which are written in SMPS format and publicly available online [1]. This study implements an SMPS parser that is based on the Julia programming language and the COIN-OR linear program solver. All numerical experiments are computed on a machine with i7-6700K CPU and 32GB memory.

2.7.2 Wasserstein Distance-based Screening with Various Numbers of Promising Solutions

We adopt the following benchmark strategies to compare with $\mathcal{WS}$:

- Random screening ($\mathcal{RS}$): a fixed number of solutions is randomly selected from the potential solutions.

- Moment discrepancy ($\mathcal{MD}$): a fixed number of solutions is selected according to the total discrepancy of the first four statistical moments (mean, variance, skewness and Kurtosis) between measures $\mathbb{P}$ and $\mathbb{Q}$.

The experimental settings are as follows. We implement SAA with 200 samples to generate 200 approximate solutions for each test instance. Table 2.2 shows the statistical description of those approximate solutions. Figure 2.1 graphically illustrates performances of the potential solutions in terms of the mean, range and confidence intervals. We apply three strategies to select various numbers (from 1 to 200) of solutions and compute the corresponding performance loss (as discussed in Section 2.4). The performance of each solution is evaluated with 30 groups of

[1] `http://pages.cs.wisc.edu/~swright/stochastic/sampling/` and `http://plato.asu.edu/sd/instances/`

Table 2.2: Statistical description of potential solutions.

Problem Instances	Mean	Standard Deviation	Maximum	Minimum
LandS	225.659	0.029	225.823	225.639
Retail	156.034	2.225	172.673	154.393
20term	254318.492	20.011	254441.922	254305.193
SSN	10.990	0.375	12.496	10.385

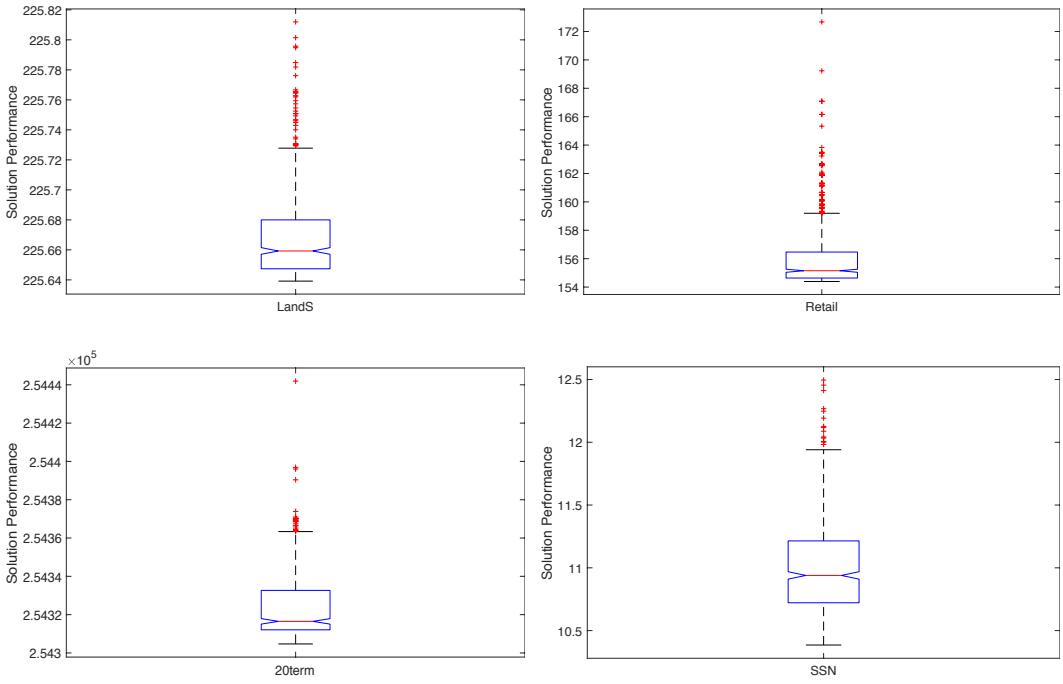

Figure 2.1: Box plots providing the mean (horizontal line in box), range (vertical lines extending from box), and 95% confidence intervals (vertical extent of box) for the performances of candidate solutions.

500,000 samples. For $\mathcal{WS}$, the solution's Wasserstein distance is estimated using four samples with a size of 1,000.

The results in Figure 2.2 confirm the superiority of $\mathcal{WS}$ in all test instances. The performance of $\mathcal{RS}$ and $\mathcal{MD}$ are worse than that of $\mathcal{WS}$ when the number of promising solutions is less than 30 in test instances 20term, Retail and SSN. While the $\mathcal{MD}$ approach outperforms $\mathcal{RS}$ in all benchmark problems except LandS. The average performance losses obtained by three strategies coincide as the number of promising solutions approaches 200. Moreover, the convergence of $\mathcal{WS}$ in LandS and Retail is more rapid than that in SSN and 20term, indicating that the stability of the quantitative result varies with the problem structure. The results also imply that the required number of promising solutions for guaranteeing a relatively small

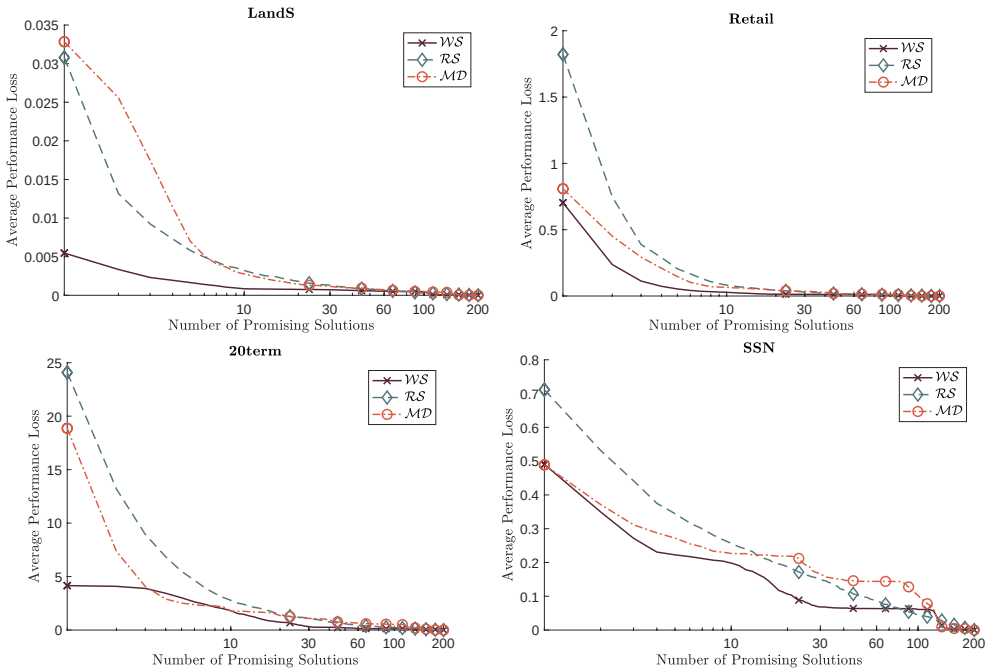

Figure 2.2: Comparison of average performance losses obtained by $\mathcal{RS}$ and $\mathcal{WS}$ using various numbers of promising solutions.

performance loss changes with the problem structure. For example, more solutions need to be included in the simulation procedure for 20term and SSN than LandS and Retail.

2.7.3 Average Performance Comparison with Benchmark Algorithms

We also conduct numerical experiments to study the performance of the solution selection algorithm in terms of the CPU time. The performance of $\mathcal{WS}$-$\mathcal{OCBA}$ is compared with that of the following approaches.

- $\mathcal{OCBA}$ is the standard $\mathcal{OCBA}$ algorithm as described in Algorithm 2.1 .
- $\mathcal{EAS}$ uses equal allocation and selection algorithm, which sequentially allocates the same number of simulations to each candidate solution and selects the current best solution based on the simulation results.
- $\mathcal{WS}$-$\mathcal{EAS}$ employs $\mathcal{EAS}$ with the proposed Wasserstein-based solution screening procedure.

For each test instance, we utilise SAA with 200 samples to generate 200 candidate solutions and then apply all algorithms to identify the best candidate.

This procedure is repeated 100 times to examine the average performance of each algorithm. The detailed experimental settings are listed in Table 2.3. Figure 2.3 presents the convergence comparisons over 100 runs in terms of CPU time and average solution performance. The computational costs of solution screening for the $\mathcal{WS}$-based algorithms are included in the comparison results.

Table 2.3: Experimental settings.

Problem Instances	Algorithms	Solution Screening				N_E	C/E(s)	Initial Estimation	
		$\Lambda^{\mathcal{P}}$	$\mathcal{I}$	N_W	TC(s)			K	TC(s)
LandS	$\mathcal{EAS}$	–	–	–	–	50,000	1.5	–	–
	$\mathcal{OCBA}$							5	1,460
	$\mathcal{WS}$-$\mathcal{EAS}$	10	4	1,000	91			–	–
	$\mathcal{WS}$-$\mathcal{OCBA}$							5	1,460
Retail	$\mathcal{EAS}$	–	–	–	–	50,000	2.4	–	–
	$\mathcal{OCBA}$							5	2,400
	$\mathcal{WS}$-$\mathcal{EAS}$	10	4	1,000	105			–	–
	$\mathcal{WS}$-$\mathcal{OCBA}$							5	2,400
20term	$\mathcal{EAS}$	–	–	–	–	20,000	20.4	–	–
	$\mathcal{OCBA}$							5	20,400
	$\mathcal{WS}$-$\mathcal{EAS}$	10	4	1,000	128			–	–
	$\mathcal{OCBA}$							5	20,400
SSN	$\mathcal{EAS}$	–	–	–	–	20,000	24.8	–	–
	$\mathcal{OCBA}$							5	24,800
	$\mathcal{WS}$-$\mathcal{OCBA}$	10	4	1,000	154			–	–
	$\mathcal{WS}$-$\mathcal{OCBA}$							5	24,800

$\Lambda^{\mathcal{P}}$: number of promising solution; $\mathcal{I}$: number of samples for Wasserstein estimation;
N_W : sample size for Wasserstein estimation; TC: total CPU time;
N_E: sample size for performance evaluation; C/E: CPU time of each evaluation
K: number of samples used in the initial estimation.

As shown in Figure 2.3, the $\mathcal{EAS}$ algorithm provides the worst results for all test instances among all algorithms because the equal allocation strategy might spend unnecessary evaluations on candidate solutions with a large mean. We also observe that $\mathcal{EAS}$ and $\mathcal{OCBA}$ exhibit similar convergence patterns on LandS (before 1,500s) and Retail (before 2,500s) whereas the overall patterns of $\mathcal{EAS}$ and $\mathcal{OCBA}$ for 20term and SSN are almost identical. The reason behind this phenomenon could be because $\mathcal{OCBA}$ has a sufficient computing budget to finish the initial estimation on LandS and Retail, but it is not the case for test instances 20term and SSN (see Table 2.3). When $\mathcal{OCBA}$ stays in the stage of initial estimation, its evaluation behaviour (as described in Algorithm 2.1) is similar to $\mathcal{EAS}$.

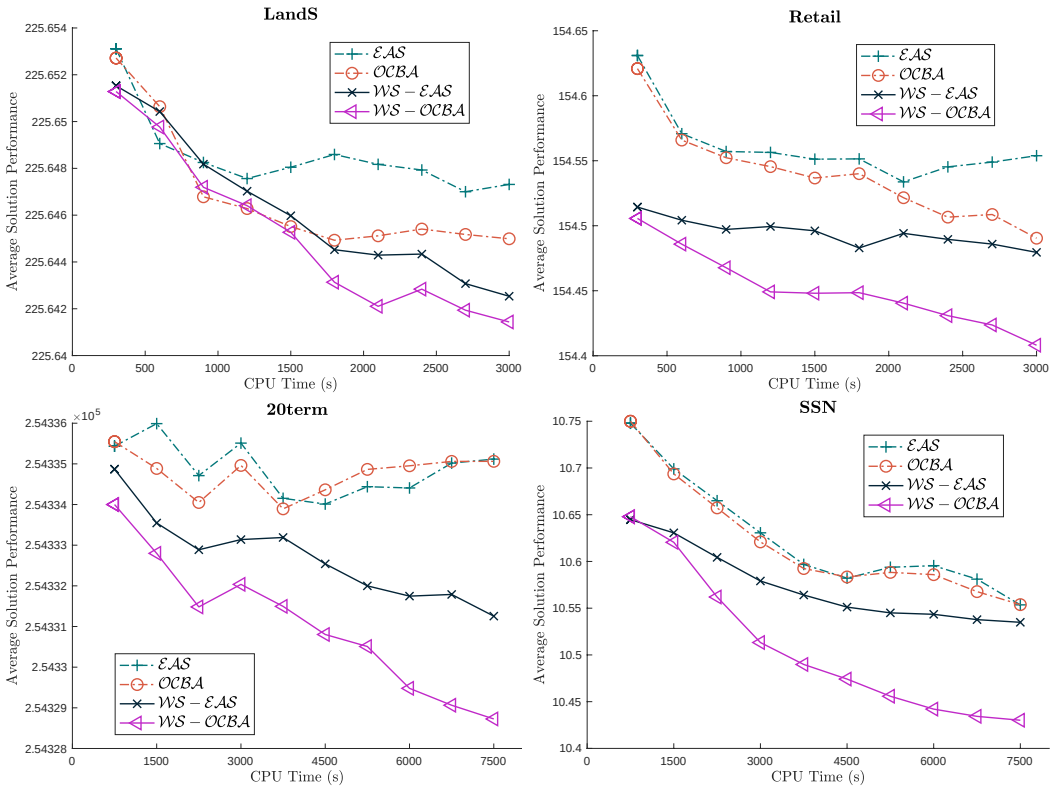

Figure 2.3: Convergence comparison with respect to CPU time.

However, $\mathcal{OCBA}$ performs worse than any of the $\mathcal{WS}$-based algorithms on all test instances, indicating that $\mathcal{WS}$ enhances performance of the solution selection algorithm for the case of numerous potential solutions. For $\mathcal{WS}$-$\mathcal{OCBA}$, the computational cost for initial estimation is greatly reduced. We observe that $\mathcal{WS}$-$\mathcal{OCBA}$ achieves the fastest convergence in all test instances, as this algorithm has more time to explore the performance of each promising solution compared with $\mathcal{OCBA}$.

Table 2.4: Average solution performance of the selected subset (mean ± std. err).

Problem Instances	CPU Time (s)	Algorithms			
		$\mathcal{EAS}$	$\mathcal{OCBA}$	$\mathcal{WS}$-$\mathcal{EAS}$	$\mathcal{WS}$-$\mathcal{OCBA}$
LandS	1,000	**225.648±0.002**	**225.647±0.001**	**225.648±0.002**	**225.647±0.001**
	2,000	225.648±0.001	225.645±0.001	**225.644±0.001**	**225.643±0.001**
	3,000	225.647±0.001	225.645±0.001	**225.643±0.001**	**225.641±0.001**
Retail	1,000	154.556±0.013	154.545±0.017	154.499±0.009	**154.449±0.003**
	2,000	154.545±0.012	154.506±0.008	154.490±0.009	**154.431±0.001**
	3,000	154.554±0.013	154.490±0.006	154.479±0.003	**154.408±0.001**
20term	2,500	254334.892±1.041	254334.267±1.050	254333.018±0.894	**254331.562±0.835**
	5,000	254334.172±0.952	254334.797±0.948	254332.134±0.707	**254330.358±0.665**
	7,500	254335.117±0.936	254335.062±0.931	254331.255±0.792	**254328.726±0.706**
SSN	2,500	10.654±0.015	10.645±0.014	10.595±0.014	**10.543±0.012**
	5,000	10.589±0.015	10.586±0.016	10.546±0.011	**10.430±0.009**
	7,500	10.553±0.013	10.553±0.014	10.534±0.011	**10.461±0.008**

We display the performance of the solutions obtained from the various algorithms averaged over 100 repetitions in Table 2.4. Best results and those statistically not different from best are highlighted in bold. The results confirm the importance of adopting solution screening in the selection procedure for numerous potential solutions. The $\mathcal{WS}$-based algorithms significantly outperform the others in all benchmark problems, although all algorithms display similar performance in LandS and 20terms when the CPU time is 1,000s. Moreover, we find that $\mathcal{WS}$-$\mathcal{OCBA}$ performs better than $\mathcal{WS}$-$\mathcal{EAS}$ due to its advanced computing budget allocation scheme.

2.8 Conclusions

The solution selection problem for large-scale two-stage problems is challenging for decision makers with a relatively limited computational budget when numerous potential solutions are present. Thus, we may consider removing several potential solutions from the simulation. This study shows that the worst-case solution performance in the corresponding Wasserstein-based regions satisfies the sequence of the corresponding Wasserstein distances. On the basis of this property, we propose a new solution screening approach and integrate this approach with an optimal computing budget allocation algorithm. Empirical results for various benchmark problems demonstrate the benefit of the Wasserstein-based screening and the advantage of applying the optimal computing budget allocation algorithm in selecting the near-best solutions.

Chapter 3

Efficient Sampling Strategies when Searching for Robust Solutions

3.1 Introduction

Given its ubiquity in many real-world problems, optimisation under uncertainty has gained increasing attention. Uncertainty may originate from various sources, such as imprecise model parameters or fluctuations in environmental variables. In the presence of uncertainty, it is often desirable to find a solution that is robust in the sense of performing well under a range of possible scenarios [Beyer and Sendhoff, 2007]. More specifically, our paper considers searching for robust solutions in the sense of high expected performance, given a distribution of disturbances to the decision variables. This is a typical scenario for example in manufacturing, where the actually manufactured products may differ from the design specification due to manufacturing tolerances.

Evolutionary algorithms (EAs) have been applied to solve various optimisation problems that involve uncertainties, see Jin and Branke [2005] for a survey. There are also different concepts related to robustness, including optimising the worst-case performance [Ong et al., 2006], robust optimisation over time [Fu et al., 2015], and active robustness [Salomon et al., 2016]. The most widely researched robustness concept however, and the concept we consider in this chapter, is to optimise a solution's expected fitness (often called effective fitness in the literature) over the possible disturbances [Beyer and Sendhoff, 2007].

To estimate an individual's effective fitness, sampling has been widely adopted

in practice, and two sampling methods can be distinguished. *Implicit* sampling refers to the idea that because EAs are population-based, an over-evaluated individual due to a favourable disturbance can be counterbalanced by an under-evaluated neighbouring individual, and so simply increasing the population size should help guiding the EA in the right direction, e.g. see Rattray and Shapiro [1998] and Beyer and Sendhoff [2006]. In fact, as shown in Tsutsui and Ghosh [1997], under the assumption of infinite population size and fitness-proportional selection, evaluating each solution at a single disturbed sample location instead of its actual location is equivalent to optimising the expected fitness directly. Beyer and Sendhoff [2006] proposed to increase the population size whenever the algorithm gets stalled. *Explicit* sampling, on the other hand, evaluates each individual multiple times and estimates its fitness as the average of the sampled evaluations. Obviously, while averaging over multiple evaluations increases the estimator accuracy, it is computationally rather expensive. A recent analytical study on the efficiency (progress rate) of implicit as well as explicit sampling-based evolution strategies can be found in Beyer and Sendhoff [2017].

Because of the large computational cost of sampling in case of expensive fitness functions, numerous studies have focused on allocating a limited sampling budget to improve the estimation accuracy, allowing to reduce the number of samples needed without degrading the performance of the EA. One possible approach for estimating effective fitness is to apply quadrature rules or variance reduction techniques, e.g. see Branke et al. [2001]; Di Pietro et al. [2004]; Huang and Du [2006] and Lee et al. [2009]. Some authors observed that allocating samples in a manner that increases the probability of correct selection is more important than the accuracy of estimation [Stagge, 1998; Branke and Schmidt, 2003].

Another approach, called archive sample approximation (ASA), stores all past evaluations in a memory and uses this information to improve expected fitness estimates in future generations (for further information, refer to Branke et al. [2001] and Kruisselbrink et al. [2010]). ASA can also be combined with the above sampling allocation methods to further enhance the accuracy of approximation. We have recently proposed an improved ASA method that uses the Wasserstein distance metric (a probability distance that quantifies the dissimilarity between two statistical objects) to identify the sample that is likely to provide the most valuable additional information to complement the knowledge available in the memory [Branke and Fei, 2016].

In this chapter, we propose a Wasserstein-based archive sample approximation (WASA) framework. The sampling strategy in our previous study Branke and

Fei [2016] is one of many possible approaches in the WASA framework. Here, a new sampling strategy based on WASA is presented, which improves the performance in three ways:

1. A heuristic that considers what sample might provide the most valuable information for *all* individuals in the population *simultaneously*, thereby enhancing the performance of the final solution and accelerating the convergence of the EA toward a high-performance solution;

2. Introducing the concept of an approximation region that could improve the performance of the sampling strategy when few samples exist; and

3. Proposing a sample budget mechanism to adjust the sampling budget in the WASA framework.

This chapter is structured as follows. Section 3.2 provides a brief overview of existing archive sample approximation methods. We then present our new approaches to allocating samples in Section 3.3. These approaches are evaluated empirically using benchmark problems, and their performances are compared with other approaches in Section 3.4. This chapter concludes with a summary.

3.2 Archive Sample Approximation

The problem of searching for robust solutions can be defined as follows. Consider an objective function $f : x \to \mathbb{R}$ with design variables $x \in \mathbb{R}^m$. The noise is defined on the probability space $(\Xi, \mathcal{B}(\Xi), \mathcal{P})$ where $\Xi = \prod_i^m [\ell_i, u_i]$ is a sample space, $\mathcal{B}(\Xi)$ is the Borel σ-algebra on Ξ and $\mathcal{P}$ is the probability measure on $\mathcal{B}(\Xi)$. For a particular solution x, location z_x is random under this noisy environment, which is defined as:

$$z_x = x + \xi, \quad \xi \in \Xi. \tag{3.1}$$

We then define the induced probability space for z_x as $(\mathcal{D}_x, \mathcal{B}(\mathcal{D}_x), \mathbb{P}_x)$, where $\mathcal{D}_x$ is the *disturbance region* as the set covering all possible locations as a result of disturbing solution x defined by:

$$\mathcal{D}_x = \prod_{i=1}^{m} [x_i + \ell_i, x_i + u_i]; \tag{3.2}$$

and P_x is the probability measure defined so that, for $\phi \in \mathcal{B}(\mathcal{D}_x)$,

$$\mathbb{P}_x(\phi) = \mathcal{P}(Z_x^{-1}(\phi)). \tag{3.3}$$

Without loss of generality, for minimisation, the goal is to minimise the effective fitness, that is, the expected performance over the disturbance region as follows:

$$\min_{x \in \mathcal{X}} \quad f_{eff}(x) = \min \int_{\mathcal{D}_x} f(z) d\mathbb{P}_x(z). \tag{3.4}$$

where $\mathcal{X} \subset \mathbb{R}^m$ is the feasibility set.

As the fitness formulation is unknown in many industrial applications, the integral cannot be computed directly. In practice, we can numerically compute this integral using sampling techniques. Let $\mathcal{N}$ denote the number of realisations to represent the disturbance region, and $\hat{Z}_x = \{z_n | \ n \in \mathcal{N}\}$ be the realisations. The empirical probability measure (i.e. probability distribution) of these samples is a discrete probability measure, which can be defined so that, for $\phi \in \mathcal{B}(\hat{Z}_x)$,

$$\hat{\mathbb{P}}_x(\phi) = \frac{1}{|N|} \sum_{n \in \mathcal{N}} \mathbb{1}_\phi(z_n), \tag{3.5}$$

where $\mathbb{1}_\phi(\cdot)$ is the indicator function. If the sample size $|N|$ is sufficiently large, the effective fitness can be well estimated as follows:

$$f_{eff}(x) \approx \sum_{n \in \mathcal{N}} f(z_n) \hat{\mathbb{P}}_x(z_n). \tag{3.6}$$

However, if evaluating the fitness function is computationally expensive, this may not be possible.

The ASA approach originally proposed in Branke [2001] that saves previously evaluated points in the search space and their corresponding fitness values in an archive $\mathcal{A}$. Generally, the archive $\mathcal{A}$ is a set of ordered pairs (z, ρ), where z is the sample location and ρ is its fitness value. The archive information can be reused when estimating the expected fitness of a new solution. ASA performs three main steps during a fitness evaluation.

1. The previously evaluated points which are in the solution's *disturbance region* are retrieved from the archive. Given the disturbance region $\mathcal{D}_x$, the available archive information can be identified as

$$\mathcal{A}_x = \{(z, \rho) \in \mathcal{A} | \ z \in \mathcal{D}_x\}.$$

2. A *sampling strategy* is used to determine at what additional locations samples should be collected. Let C_x denote the new sample locations. It is desirable to

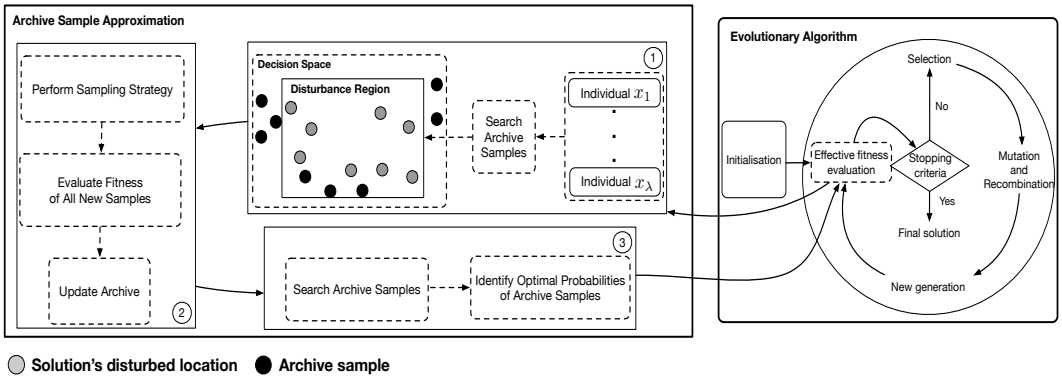

Figure 3.1: Illustration of integrating ASA into an EA.

select a set of sample points C_x which maximise our knowledge of the fitness landscape in the disturbance region.

3. All samples located within the disturbance region $\mathcal{D}_x$ need to be assigned probabilities. Then, the estimated effective fitness $\hat{f}_{eff}(x)$ can be calculated as

$$\hat{f}_{eff}(x) = \underbrace{\sum_{a \in \mathcal{A}_x} a^{(2)} \mathbb{Q}_x(a^{(1)})}_{\text{Archive information}} + \underbrace{\sum_{c \in C_x} f(c) \mathbb{Q}_x(c)}_{\text{New information}}$$

where $\mathbb{Q}_x(\cdot)$ is the point probability; $a^{(1)}$ and $a^{(2)}$ represent the first and second elements of the ordered pair in the archive $\mathcal{A}_x$.

The overall procedure integrating ASA into an EA is visualised in Figure 3.1.

In the simplest ASA, the sampling strategy randomly evaluates new points within the disturbance region and assigns equal probabilities to all available samples, see, e.g., Branke et al. [2001]. However, if the distribution of available samples in the disturbance region is not representative of $\mathcal{D}_x$, the resulting estimation of effective fitness may be very biased. In an attempt to fill "holes" in the disturbance region, the authors in the studies of Kruisselbrink et al. [2010]; Saha et al. [2011] and Cervantes et al. [2017] proposed to iteratively pick a sample point that has maximal distance from any archive sample point in the disturbance region.

3.3 The Wasserstein-based Archive Sample Approximation

Given the archive samples, we aim to provide an accurate estimation of effective fitness for each individual by sampling additional locations. The estimation error

with respect to effective fitness is defined as follows:

$$e_{eff}(x) = |f_{eff}(x) - \hat{f}_{eff}(x)|. \tag{3.7}$$

In reality, estimation error cannot be computed given that the actual effective fitness $f_{eff}(x)$ is unknown. Therefore, we use Wasserstein distance as an upper bound approximation for the estimation error. We call this framework Wasserstein-based archive sample approximation.

In the following subsections, we first explain the principle behind WASA and describe the underlying sampling problem of WASA. Then, we enhance the description of the sampling strategy used in our previous work [Branke and Fei, 2016] and propose a new population-based myopic sampling strategy.

3.3.1 The Upper Bound Approximation for Estimation Error

As discussed in Section 3.2, the effective fitness of a particular solution can be numerically computed by using a set of samples. If the number of samples is large enough, the corresponding empirical probability measure will converge to the actual probability. Given that the computational cost of evaluating a sample is expensive, we aim to approximate this "large" empirical probability measure with a "smaller" discrete probability measure. The challenge is how to estimate the corresponding estimation error. Since the actual effective fitness cannot be computed in practice, the estimation error cannot be obtained directly.

In the following, we provide an upper bound approximation of the estimation error that can be calculated without requiring any fitness information.

Theorem 3.1. Let $\hat{\mathbb{P}}$ and $\mathbb{Q}$ be two discrete probability distributions with samples $\mathcal{G} = \{g_j |\ j \in \mathcal{J}\} \subseteq \mathcal{X}$ and $\mathcal{H} = \{h_k |\ k \in \mathcal{K}\} \subseteq \mathcal{X}$, and corresponding probabilities $\{\hat{\mathbb{P}}(g_j) |\ j \in \mathcal{J}\}$ and $\{\mathbb{Q}(h_k) |\ k \in H\}$, respectively. Moreover, let $W(\hat{\mathbb{P}}, \mathbb{Q})$ denote the Wasserstein distance between probability distributions $\hat{\mathbb{P}}$ and $\mathbb{Q}$. The effective fitness is numerically computed by using the probability measure $\hat{\mathbb{P}}$; and $\mathbb{Q}$ is the probability measure approximating $\hat{\mathbb{P}}$. If the fitness function $f : x \to \mathbb{R}$ is Lipschitz continuous over domain $\mathcal{X}$, then the estimation error satisfies

$$\left| \sum_{j \in \mathcal{J}} f(g_j)\hat{\mathbb{P}}(g_j) - \sum_{k \in \mathcal{K}} f(h_k)\mathbb{Q}(h_k) \right| \leq \alpha\ W(\hat{\mathbb{P}}, \mathbb{Q}) \tag{3.8}$$

where $\alpha \geq \max\limits_{j \in \mathcal{J}, k \in \mathcal{K}} \dfrac{|f(g_j) - f(h_k)|}{d(g_j, h_k)}$ is a positive constant and $d(g_j, h_k)$ is the Eu-

clidean distance between g_j and h_k.

Proof. The effective fitness can be numerically computed by using the probability measure $\hat{\mathbb{P}}$; and $\mathbb{Q}$ is the probability measure approximating $\hat{\mathbb{P}}$. Hence, the estimation error can be written as

$$\left| \sum_{j \in \mathcal{J}} f(g_j) \hat{\mathbb{P}}(g_j) - \sum_{k \in \mathcal{K}} f(h_k) \mathbb{Q}(h_k) \right|. \tag{3.9}$$

Let ψ be one of joint probabilities with marginal probability distributions $\hat{\mathbb{P}}$ and $\mathbb{Q}$ such that the following equations hold

$$\sum_{j \in \mathcal{J}} \psi_{j,k} = \mathbb{Q}(h_k) \tag{3.10}$$

and

$$\sum_{k \in \mathcal{K}} \psi_{j,k} = \hat{\mathbb{P}}(g_j). \tag{3.11}$$

The sample sets $\mathcal{G}$ and $\mathcal{H}$ are independent of the probability measures $\mathbb{Q}$ and $\hat{\mathbb{P}}$, respectively. Hence, we can obtain

$$\sum_{j \in \mathcal{J}} f(g_j) \hat{\mathbb{P}}(g_j) = \sum_{k \in \mathcal{K}} \sum_{j \in \mathcal{J}} f(g_j) \psi_{j,k} \tag{3.12}$$

and

$$\sum_{k \in \mathcal{K}} f(h_k) \mathbb{Q}(h_k) = \sum_{k \in \mathcal{K}} \sum_{j \in \mathcal{J}} f(h_k) \psi_{j,k}. \tag{3.13}$$

Substituting (3.12) and (3.13) in (3.9), we obtain

$$\begin{aligned} &\left| \sum_{j \in \mathcal{J}} f(g_j) \hat{\mathbb{P}}(g_j) - \sum_{k \in \mathcal{K}} f(h_k) \mathbb{Q}(h_k) \right| \\ &= \left| \sum_{k \in \mathcal{K}} \sum_{j \in \mathcal{J}} f(g_j) \psi_{j,k} - \sum_{k \in \mathcal{K}} \sum_{j \in \mathcal{J}} f(h_k) \psi_{j,k} \right|, \forall \psi \in \Psi. \end{aligned} \tag{3.14}$$

From the definition of Lipschitz continuity, we can continue

$$\begin{aligned}
&\left|\sum_{j\in\mathcal{J}} f(g_j)\hat{\mathbb{P}}(g_j) - \sum_{k\in\mathcal{K}} f(h_k)\mathbb{Q}(h_k)\right| \\
&\leq \sum_{k\in\mathcal{K}}\sum_{j\in\mathcal{J}} |f(g_j) - f(h_k)|\psi_{j,k}, \forall\psi \in \Psi \\
&\leq \alpha \sum_{j\in\mathcal{J}}\sum_{k\in\mathcal{K}} d(g_j, h_k)\psi_{j,k}, \forall\psi \in \Psi,
\end{aligned} \tag{3.15}$$

where constant α is chosen such a way that

$$\alpha \geq \max_{j\in\mathcal{J},k\in\mathcal{K}} \frac{|f(g_j) - f(h_k)|}{d(g_j, h_k)}.$$

Since the measure ψ can be any joint joint probabilities with marginal probability distributions $\mathbb{Q}$ and $\hat{\mathbb{P}}$, we can have various upper bounds for the estimation error. Given the constant α, the tightest bound can be found by

$$\min_{\psi_{j,k}}\{\sum_{j\in\mathcal{J}}\sum_{k\in\mathcal{K}} d(g_j, h_k)\psi_{j,k}\} \tag{3.16}$$

which is exactly the definition of Wasserstein distance in Appendix A. Hence, we can write

$$\left|\sum_{j\in\mathcal{J}} f(g_j)\hat{\mathbb{P}}(g_j) - \sum_{k\in\mathcal{K}} f(h_k)\mathbb{Q}(h_k)\right| \leq \alpha \min_{\psi\in\Psi}\{\sum_{j\in\mathcal{J}}\sum_{k\in\mathcal{K}} d(g_j, h_k)\psi_{j,k}\} = \alpha W(\hat{\mathbb{P}}, \mathbb{Q}). \tag{3.17}$$

□

Theorem 3.1 implies that the estimation error with respect to two discrete probability measures can be upper bounded by the Wasserstein distance.

3.3.2 Probability Redistribution in the Wasserstein Metric

This section introduces the probability redistribution in Wasserstein metric. Suppose that the Wasserstein distance between probability measures $\hat{\mathbb{P}}$ and $\mathbb{Q}$ needs to be calculated, but the probability values for sample set H can be adjustable. We aim to identify the optimal probability values, which result in the smallest Wasserstein distance value. The probability redistribution problem $M(\mathcal{G}, \mathcal{H})$ can be formulated

as follows,

$$
\begin{aligned}
M(\mathcal{G},\mathcal{H}) = \min_{\mathbb{Q}(h_k)} \quad & W(\hat{\mathbb{P}},\mathbb{Q}) \doteq \\
\min_{\psi,\mathbb{Q}(h_k)} \quad & \sum_{k\in\mathcal{K}}\sum_{j\in\mathcal{J}} d(g_j,h_k)\psi_{j,k} \\
s.t. \quad & \sum_{k\in\mathcal{K}} \psi_{j,k} = \hat{\mathbb{P}}(g_j),\ \forall j \\
& \sum_{j\in\mathcal{J}} \psi_{j,k} = \mathbb{Q}(h_k),\ \forall k \\
& \sum_{k\in\mathcal{K}} \mathbb{Q}(h_k) = 1 \\
& \mathbb{Q}(h_k) \geq 0,\ \forall k \\
& \psi_{j,k} \geq 0,\ \forall j,k.
\end{aligned}
\tag{3.18}
$$

The above optimisation problem can be regarded as a transportation problem with fixed locations. We can solve it by using the following Proposition.

Proposition 3.1. The optimal solution of (3.18) is

$$
\mathbb{Q}(h_k)^* = \sum_{j\in\mathcal{J}} \mathbb{1}_{g_j}(h_k)\hat{\mathbb{P}}(g_j),\ \forall k \tag{3.19}
$$

where $\mathbb{1}_{g_j}(h_k)$ is an indicator function and defined as

$$
\begin{cases}
\mathbb{1}_{g_j}(h_k) = 1, & \text{if } h_k \text{ is the closest sample to } g_j \\
0, & \text{otherwise.}
\end{cases}
\tag{3.20}
$$

And, the corresponding Wasserstein distance is

$$
M(\mathcal{G},\mathcal{H}) = \sum_{j\in\mathcal{J}} \hat{\mathbb{P}}(g_j) \min_{h_k}\ d(g_j,h_k). \tag{3.21}
$$

Proof. For details, see the proof of Proposition 2.1 introduced by Dempster et al. [2011]. □

Algorithm 3.1 summaries the overall procedure for computing the Wasserstein distance after optimal probability redistribution.

Algorithm 3.1: Efficient method for computing $M(\mathcal{G}, \mathcal{H})$

input : samples $\mathcal{G} = \{g_j | j \in \mathcal{J}\}$ with probabilities $\{\hat{\mathbb{P}}(g_j) | j \in \mathcal{J}\}$;
samples $\mathcal{H} = \{h_k |\ k \in \mathcal{K}\}$.
output: Wasserstein distance after redistribution $M(\mathcal{G}, \mathcal{H})$;
optimal probabilities $\{\mathbb{Q}(h_k)^* |\ k \in \mathcal{K}\}$ for samples $\mathcal{H}$.

```
for j ∈ J, k ∈ K do
    compute Euclidean distance between samples g_j and h_k;
end
compute the distance value M(G, H) via (3.21);
compute the optimal probabilities {Q(h_k)* | k ∈ K} as in (3.19).
```

3.3.3 The WASA Sampling Problem

In this subsection, we introduce the sampling problem in WASA. Let x_l for $l \in \Lambda = \{1, \ldots, \lambda\}$ denote the l-th individual from a particular population during the EA search process. Using similar notation as in Section 3.2, let Z_l for $l \in \Lambda$ be the random location of individual x_l, which is defined on the probability space $(\hat{\mathcal{D}}_l, \mathcal{B}(\hat{\mathcal{D}}_l), \mathbb{P}_l)$. We then generate a large set $\hat{\mathcal{Z}}_l = \{z_{n,l} \in \hat{\mathcal{D}}_l |\ n \in \mathcal{N}\}$ to represent the disturbed locations of each individual. The empirical probability measure of these disturbed locations is denoted by $\hat{\mathbb{P}}_l$. Note that these large sample sets are used only when computing the Wasserstein distance value and as a set of candidates for evaluation.

Given an archive $\mathcal{A}$ that consists of previously evaluated solutions, the available archive samples for individual x_l can be retrieved as follows:

$$\mathcal{A}_l = \{(z, \rho) \in \mathcal{A} |\ z \in \hat{\mathcal{D}}_l\}. \tag{3.22}$$

Then, the previously evaluated sample locations $\mathcal{S}_l$ can be identified as

$$\mathcal{S}_l = \{a^{(1)} |\ a \in \mathcal{A}_l\}. \tag{3.23}$$

The *optimal sampling problem* is to minimise the total effective fitness estimation error over all individuals at the present EA's generation by searching:

- the best additional sample locations $\mathcal{C}_l$; and
- the optimal probability measure $\mathbb{Q}_l$ on $\mathcal{B}(\mathcal{C}_l \cup \mathcal{S}_l)$.

We can formulate the aforementioned problem by using the following optimisation

model:

$$\begin{aligned}
\min_{\mathcal{C}_l,\mathbb{Q}_l} \quad & \sum_{l\in\Lambda} e_{eff}(x_l) = \sum_{l\in\Lambda} |f_{eff}(x_l) - \hat{f}_{eff}(x_l)| \\
s.t. \quad & f_{eff}(x_l) = \sum_{n\in\mathcal{N}} f(z_{n,l})\hat{\mathbb{P}}_l(z_{n,l}), \forall l \\
& \hat{f}_{eff}(x_l) = \sum_{a\in\mathcal{A}_l} a^{(2)}\mathbb{Q}_l(a^{(1)}) + \sum_{c\in\mathcal{C}_l} f(c)\mathbb{Q}_l(c), \forall l \\
& \sum_{a\in\mathcal{A}_l} \mathbb{Q}_l(a^{(1)}) + \sum_{c\in\mathcal{C}_l} \mathbb{Q}_l(c) = 1,\ \mathbb{Q}_l(\cdot) \geq 0, \forall l \\
& \sum_{l\in\Lambda} |\mathcal{C}_l| = B,\ \mathcal{C}_l \subseteq \hat{\mathcal{Z}}_l, \forall l.
\end{aligned} \tag{3.24}$$

As previously discussed, some fitness information in (3.24) is unavailable. Therefore, the optimal sampling problem cannot be directly solved. By using Theorem 3.1, the objective function of the optimal sampling problem can be bounded by

$$\begin{aligned}
\sum_{l\in\Lambda} e_{eff}(x_l) &= \sum_{l\in\Lambda} \left| \sum_{n\in\mathcal{N}} f(z_{n,l})\hat{\mathbb{P}}_l(z_{n,l}) - \left[\sum_{a\in\mathcal{A}_l} a^{(2)}\mathbb{Q}_l(a^{(1)}) + \sum_{c\in\mathcal{C}_l} f(c)\mathbb{Q}_l(c) \right] \right| \\
&\leq \alpha \sum_{l\in\Lambda} W(\hat{\mathbb{P}}_l, \mathbb{Q}_l).
\end{aligned} \tag{3.25}$$

Then, we obtain the formulation of WASA sampling problem as follows,

$$\begin{aligned}
\min_{\mathcal{C}_l,\mathbb{Q}_l} \quad \alpha \sum_{l\in\Lambda} W(\hat{\mathbb{P}}_l, \mathbb{Q}_l) = \min_{\mathcal{C}_l} \quad & \alpha \sum_{l\in\Lambda} M(\hat{\mathcal{Z}}_l, \mathcal{C}_l \cup \mathcal{S}_l) \\
s.t. \quad & \sum_{l\in\Lambda} |\mathcal{C}_l| = B, \mathcal{C}_l \subseteq \hat{\mathcal{Z}}_l, \quad \forall l.
\end{aligned} \tag{3.26}$$

As α only appears in the objective function of (3.26), it can be disregarded because it does not influence the choice of optimal decision. Hence, the optimisation model in (3.26) can be re-written as

$$\begin{aligned}
\min_{\mathcal{C}_l,\mathbb{Q}_l} \quad \sum_{l\in\Lambda} W(\hat{\mathbb{P}}_l, \mathbb{Q}_l) = \min_{\mathcal{C}_l} \quad & \sum_{l\in\Lambda} M(\hat{\mathcal{Z}}_l, \mathcal{C}_l \cup \mathcal{S}_l) \\
s.t. \quad & \sum_{l\in\Lambda} |\mathcal{C}_l| = B, \mathcal{C}_l \subseteq \hat{\mathcal{Z}}_l, \quad \forall l.
\end{aligned} \tag{3.27}$$

This new formulation is referred to as the WASA sampling problem. Essentially, we minimise the upper bound approximation of the estimation error in order to provide heuristic sample locations without requiring any fitness information. In the derivation process, we use a constant α to represent the Lipschitz constants

which are unknown in the black-box objective function. Since α only appears in the objective function, we ignore it in the subsequent analysis. However, this approach might result in a loose upper bound. In the future work, we can surrogate-based approaches to estimate these Lipschitz constants and achieve a tighter upper bound approximation for the estimation error.

The WASA sampling problem is a nonlinear and non-convex optimisation problem. The optimal solution for this approximate problem formulation can be found by repeatedly solving the following four related steps:

1. Fixing the cardinality $|\mathcal{C}_l|$ for $l \in \Lambda$ such that $\sum_{l\in\Lambda} |\mathcal{C}_l| = B$.

2. Selecting the new samples $\mathcal{C}_l$ from $\hat{\mathcal{Z}}_l$.

3. Constructing the sample set $\mathcal{Y}_l$ for each individual x_l as follows,

$$\mathcal{Y}_l = \mathcal{S}_l \cup \mathcal{C}_l, \ \forall l. \tag{3.28}$$

4. Computing the Wasserstein distance $M(\hat{\mathcal{Z}}_l, \mathcal{Y}_l)$ for $l \in \Lambda$ by applying Algorithm 3.1 to solve the following optimisation model.

$$\begin{aligned}
\min_{\mathbb{Q}_l, \psi_{z,y}} \ & \sum_{z\in\hat{\mathcal{Z}}_l} \sum_{y\in\mathcal{Y}_l} d(z,y)\psi_{z,y} \\
s.t. \ & \sum_{y\in\mathcal{Y}_l} \psi_{z,y} = \hat{\mathbb{P}}_l(z), \ \forall z \in \hat{\mathcal{Z}}_l \\
& \sum_{z\in\hat{\mathcal{Z}}_l} \psi_{z,y} = \mathbb{Q}_l(y), \ \forall y \in \mathcal{Y}_l \\
& \sum_{y\in\mathcal{Y}_l} \mathbb{Q}_l(y) = 1 \\
& \mathbb{Q}_l(y) \geq 0, \ \forall y \in \mathcal{Y}_l \\
& \psi_{z,y} \geq 0, \ \forall z \in \hat{\mathcal{Z}}_l, y \in \mathcal{Y}_l.
\end{aligned} \tag{3.29}$$

Generally, the optimal solution of the WASA sampling problem cannot be obtained within a reasonable time. For instance, there exist two individuals x_1 and x_2 at a particular EA's iteration. Consider that the cardinalities of $\hat{\mathcal{Z}}_1$ and $\hat{\mathcal{Z}}_2$ are 200. Given the sampling budget $B = 2$, we have three ways to allocate this sampling budget to x_1 and x_2.

$$(|\mathcal{C}_1|, |\mathcal{C}_2|) \in \{(1,1), (2,0), (0,2)\}.$$

If we should take one sample from $\hat{\mathcal{Z}}_1$ and one from $\hat{\mathcal{Z}}_2$, then we have $40,000$ sample combinations. In case of taking two samples from $\hat{\mathcal{Z}}_1$ or two from $\hat{\mathcal{Z}}_2$, $19,900$ combinations are found. This basically results in $79,800$ repetitions for performing the four related steps to find the optimal solution. The small case shows that heuristic approaches to solve the WASA sampling problem are needed.

In the next subsection, we introduce two Wasserstein-based sampling strategies aimed at efficiently making sampling decisions for the above WASA sampling problem.

3.3.4 Equal Fixed Sampling Strategy

A straightforward way to simplify the WASA sampling problem is to consider each individual separately, and allocate to each individual the same fixed number of new samples. We call this strategy Equal Fixed Sampling (EFS). The sampling budget in EFS must be an integral multiple of the number of individuals. EFS is an iterative method that determines one new sample point at each step. Algorithm 3.2 describes the overall procedure of incorporating the EFS strategy into WASA.

For individual x_l, EFS first retrieves previously evaluated information $\mathcal{A}_l$ from the archive $\mathcal{A}$ and identifies the archive sample locations $\mathcal{S}_l$ by using (3.23). Then, EFS constructs the candidate sets $\mathcal{Y}_{n,l}$ for $n \in \mathcal{N}$ by uniting one of the individual's disturbed locations $z_{n,l}$ with archive points as follows:

$$\mathcal{Y}_{n,l} = z_{n,l} \cup \mathcal{S}_l, \ \forall n \in \mathcal{N}. \tag{3.30}$$

EFS evaluates the Wasserstein distance value $V_{EFS}(z_{n,l})$ of each disturbed location $z_{n,l}$ as

$$V_{EFS}(z_{n,l}) \doteq M(\mathcal{Y}_{n,l}, \hat{\mathcal{Z}}_l), \ \forall n \in \mathcal{N}. \tag{3.31}$$

EFS next selects and evaluates the new sample point z^* resulting in the smallest value, i.e.,

$$z^* = \arg\min\{V_{EFS}(z_{n,l})|\ n \in \mathcal{N}\}. \tag{3.32}$$

The fitness of this point is evaluated and the new sample information will be added into the archive $\mathcal{A}$. The process repeats until the algorithm runs out of sampling budget for this individual.

When this iterative process has been completed, EFS retrieves the archive samples $\mathcal{A}_l^*$ from the *updated* archive. The previously evaluated sample locations for

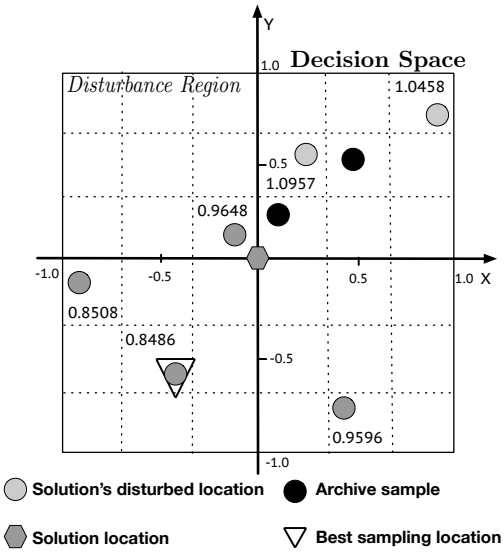

Figure 3.2: Description of computing V_{EFS} in EFS.

the individual x_l can be identified as follows,

$$\mathcal{Y}_l^* = \{a^{(1)} |\ a \in \mathcal{A}_l^*\}. \tag{3.33}$$

Now we compute the optimal probabilities $\mathbb{Q}_l$ for $\mathcal{Y}_l^*$ by using (3.29). Then, the estimated effective fitness of each individual can be calculated as

$$\hat{f}_{eff}(x_l) = \sum_{a \in \mathcal{A}_l^*} a^{(2)} \mathbb{Q}_l(a^{(1)}). \tag{3.34}$$

Figures 3.2 and 3.3 illustrate EFS by using a $2D$ example. Let us consider the solution located at $(0, 0)$, which is represented as a grey solid hexagon, assume the sampling budget of this solution is one, and the disturbance region is from -1 to 1 in x- and y-directions. We approximate the disturbance region by using Latin hypercube sampling to generate 6 disturbances. The corresponding disturbed locations of this solution are represented by grey solid circles. In the disturbance region, there are two archive points which are depicted as black solid circles. We now calculate the V_{EFS} value of each disturbed location. The results are shown in Figure 3.2. We can find that the best sample point is the grey solid circle whose V_{EFS} value is 0.8486. Figure 3.3 describes how to determine the probabilities of samples used to estimate the effective fitness. The probability of each disturbed location is 0.1667 because the total number of disturbed locations is six in this example. The black solid circle located at $(0.1600, 0.2000)$ is the point closest to the grey solid

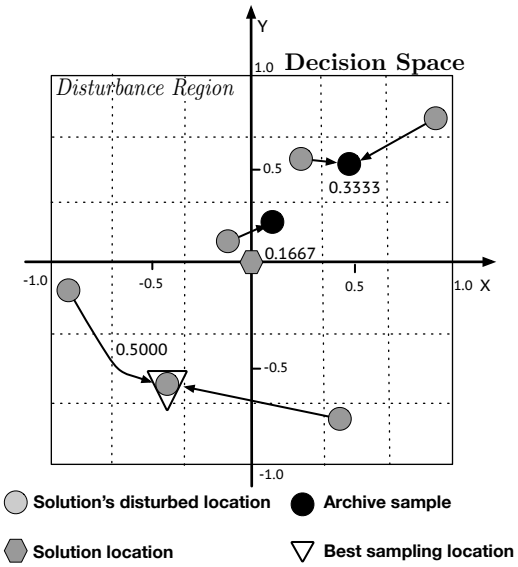

Figure 3.3: Description of calculating optimal probabilities in EFS.

circle $(-0.1863, 0.1745)$. According to probability re-allocation rules (3.19), the new probability for this black solid circle is 0.1667. The new sample point marked by a triangle is the best representation of the two grey solid circles. However, the new probability of this point is 0.5000 because this point is also the nearest to itself. The new probabilities of the other black solid circles can be assigned following the same procedure.

3.3.5 Population-based Myopic Sampling Strategy

EFS may spend unnecessary sampling on an individual that already has a small Wasserstein distance value but ignores the needs of an individual with large Wasserstein distance value. This inefficient sampling allocation frequently appears in the search process because of the exploration and exploitation strategies used in the EAs. The current promising area on the decision space is exploited repeatedly, and the archive set already contains adequate information for approximating the effective fitness landscape of individuals in this area. However, the archive does not contain information on previously unexplored areas, and in those areas new sampling is indispensable.

To alleviate this problem, we propose here population-based myopic sampling (PMS). The key idea is to allow sampling, at each step, the disturbed location of any individual that greatly minimises the average Wasserstein distance over the entire population. This sampling strategy is particularly effective when distur-

Algorithm 3.2: The procedure of including EFS into WASA

input : set of disturbed locations $\hat{\mathcal{Z}}_l = \{z_{n,l} |\ n \in \mathcal{N}\}$ with empirical probability measure $\hat{\mathbb{P}}_l$ for $l \in \Lambda$, sampling budget B, archive $\mathcal{A}$.
output: estimated effective fitness $\hat{f}_{eff}(x_l)$ for $l \in \Lambda$.

for $l \in \Lambda$ **do**
  retrieve archive information $\mathcal{A}_l$ using (3.22);
end
for $j \in \mathcal{J}$ **do**
  identify sample locations $\mathcal{S}_l$ from $\mathcal{A}_l$ as in (3.23);
end
for $b \in \{1, \ldots, B/\lambda\}$ **do**
  for $n \in \mathcal{N}$ **do**
    construct the candidate set $\mathcal{Y}_{n,l}$ using (3.30);
    compute the $V_{EFS}(z_{n,l})$ value via (3.31);
  end
  find $z^* = \arg\min\{V_{EFS}(z_{n,l}) |\ n \in \mathcal{N}\}$;
  evaluate the fitness of the best location $f(z^*)$;
  update the archive $\mathcal{A} \leftarrow \mathcal{A} \cup (z^*, f(z^*))$;
end
for $l \in \Lambda$ **do**
  retrieve the archive information $\mathcal{A}_l^*$ as in (3.22);
  construct the archive sample set $\mathcal{Y}_l^*$ via (3.33);
  calculate the probability measure $\mathbb{Q}_l$ on $\mathcal{Y}_l^*$ using (3.29);
  compute the effective fitness $\hat{f}_{eff}(x_l)$ as in (3.34).
end

bance regions partially overlap one another, because one additional sample point might contribute to improving the fitness estimate for several individuals. PMS furthermore introduces the concept of an approximation region which may be chosen different from the disturbance region and allows the use of archive samples also outside the immediate disturbance region. Finally, since the budget in a generation no longer needs to be a multiple of the population size, we introduce a mechanism to adapt the sampling budget of each generation depending on the change in the average Wasserstein distance over the entire population. Algorithm 3.3 describes the overall procedure of including PMS into WASA. In the following, we present the core ingredients of PMS, including approximation region, myopic sampling location selection, and sampling budget adjustment.

Approximation region

Generally, ASA begins with the search of available archive points which are located within the individual's disturbance region. If no or few archive points in the disturbance region, one ad-hoc method is to use archive points that are slightly outside

Algorithm 3.3: The procedure of including PMS into WASA

input : set of disturbed locations $\hat{\mathcal{Z}}_l = \{z_{n,l} |\ n \in \mathcal{N}\}$ with empirical probability measure $\hat{\mathbb{P}}_l$ for $l \in \Lambda$, sampling budget B, archive $\mathcal{A}$, approximation region parameter κ, average Wasserstein distance $\bar{W}^{t-1}$.

output: estimated effective fitness $\hat{f}_{eff}(x_l)$ for $l \in \Lambda$, average Wasserstein distance $\bar{W}^t$.

initialise $b \leftarrow 0$;
for $l \in \Lambda$ **do**
  set approximation region $\mathcal{R}_l$ via (3.35);
end
while $b < B_{upper}$ **do**
  retrieve archive samples $\mathcal{A}_l$ for $l \in \Lambda$ using (3.22);
  identify sample locations $\mathcal{S}_l$ from $\mathcal{A}_l$ for $l \in \Lambda$ via (3.23);
  for $l \in \Lambda, n \in \mathcal{N}$ **do**
    for $m \in \Lambda$ **do**
      construct the candidate set $\mathcal{Y}_{l,n,m}$ as in (3.36);
    end
    compute the $V_{PMS}(z_{n,l})$ value via (3.37);
  end
  find $z^* = \arg\min\{V_{PMS}(z_{n,l}) |\ n \in \mathcal{N}, l \in \Lambda\}$;
  evaluate the fitness of the best location $f(z^*)$;
  update the archive $\mathcal{A} \leftarrow \mathcal{A} \cup (z^*, f(z^*))$;
  compute the $\bar{W}^t$ value using (3.40);
end
if $\bar{W}^{t-1} > \bar{W}^t$ *and* $b > B_{lower}$ **then**
  $b = B_{upper}$ //stop sampling;
else
  set $b \leftarrow b + 1$;
end
for $l \in \Lambda$ **do**
  retrieve the archive information $\mathcal{A}_l^*$ as in (3.22);
  construct the archive sample set $\mathcal{Y}_l^*$ via (3.33);
  calculate the probability measure $\mathbb{Q}_l$ on $\mathcal{Y}_l^*$ using (3.29);
  compute the effective fitness $\hat{f}_{eff}(x_l)$ as in (3.34).
end

the disturbance region. Although the utilisation of those archive points goes against the setting of disturbance region, the estimation of the effective fitness might be improved. In PMS, we call this enlarged region *approximation region*. Let $\mathcal{R}_l$ denote the approximation region of individual x_l. Given the sample space Ξ of noise, the size of approximation region $\mathcal{R}_l$ is controlled by a parameter κ ($\kappa \geq 1$), which is defined as follows,

$$\mathcal{R}_l = \{x_l + \varsigma | \ \varsigma \in \prod_{i=1}^{m} [\kappa * \ell_i, \kappa * u_i]\}. \tag{3.35}$$

Obviously, the larger κ, the more archive samples are used for fitness estimation. On the other hand, if κ is chosen too large, the archive points may be located too far from the individual's disturbance region, thereby causing unpredictable errors in effective fitness estimation. So, κ needs to be chosen carefully.

Figure 3.4 illustrates the benefit of including additional archive points. The solution is located at $(0, 0)$, whilst its disturbance region is from -1 to 1 in x- and y-directions. The black solid circles denote the archive points, among which three are located within and two outside the disturbance region. The grey solid circles represent the disturbed locations of this solution. If we restrict ASA to the disturbance region, the minimum Wasserstein distance we can obtain from the three black solid points and one grey solid point to be additionally evaluated is 2.276. If we set the approximation region parameter κ as 120%, the grey squiggle in Figure 3.4 denotes the extension region with respect to the solution's disturbance region. Two additional archive points can be used to estimate effective fitness. Accordingly, the best Wasserstein distance value we can obtain from the five black and one grey points reduces to 1.2935. This example demonstrates that an enlarged approximation region may reduce the Wasserstein distance.

Myopic sampling location selection

Given the archive samples of each individual, PMS myopically selects the best sampling point. This myopic strategy iteratively builds candidate sets by combining a new sample at one disturbed location of any individual with the archive samples of any individual, as described in the following equation:

$$\mathcal{Y}_{l,n,m} = z_{n,l} \cup \mathcal{S}_m, \tag{3.36}$$

with $n \in \mathcal{N}$, $m \in \Lambda$ and $l \in \Lambda$. The myopic selection criterion is based on the average Wasserstein distance for the entire population. Let $V_{PMS}(z_{n,l})$ represent the Wasserstein distance value by sampling the disturbed location $z_{n,l}$. Then, we

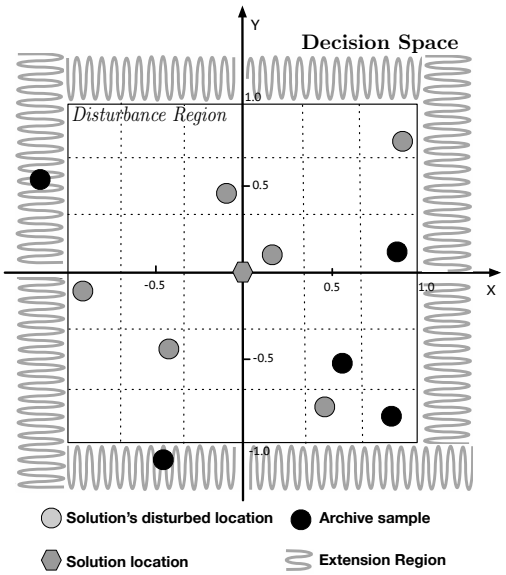

Figure 3.4: Using approximation region in WASA.

can write the $V_{PMS}(z_{n,l})$ value as follows:

$$V_{PMS}(z_{n,l}) = \sum_{m \in \Lambda} \mathbb{1}_{\mathcal{R}_m}(z_{n,l}) M(\mathcal{Y}_{l,n,m}, \hat{\mathcal{Z}}_m) \tag{3.37}$$

where $\mathbb{1}_{\mathcal{R}_m}(z_{n,l})$ is an indicator function, defined as

$$\begin{cases} \mathbb{1}_{\mathcal{R}_m}(z_{n,l}) = 1, & \text{if } z_{n,l} \in \mathcal{R}_m \\ 0, & \text{otherwise} \end{cases} \tag{3.38}$$

that controls that a sample only contributes to an individual's Wasserstein distance calculation if it is within its approximation region.

Figure 3.5 explains how to compute the V_{PMS} value in the PMS strategy. As shown in Figure 3.5, two candidate solutions x_1 and x_2 are respectively located at $(-0.666, -0.333)$ and $(0.666, 0.333)$. The disturbance region of each solution is represented by 6 Latin hypercube samples (grey solid circles). The approximation region parameter κ is set to be 130%. The grey squiggle is the extension region with respect to the disturbance region. Given the budget to sample one new point, we need to determine which grey point is the best sampling location for these two solutions (x_1 and x_2) when it is combined with the archive points (black). Note that the grey points outside the overlapped approximation region, for instance, $(-1.5340, -0.6312)$, only contribute to one solution. The V_{PMS} value for

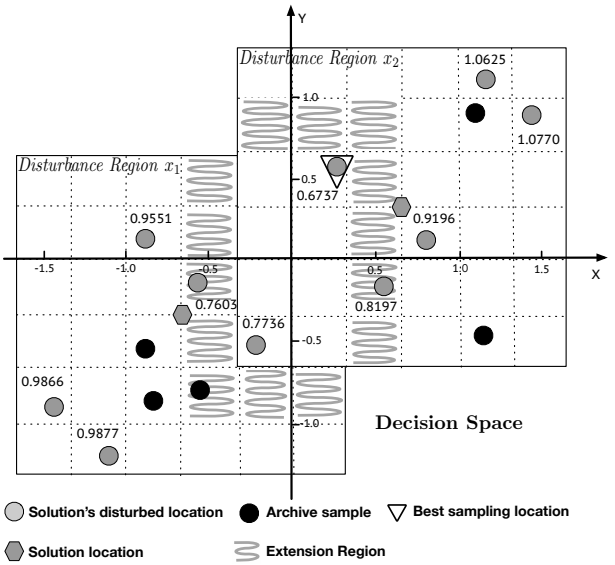

Figure 3.5: The V_{PMS} value computation in PMS.

each grey point is shown in Figure 3.5. It can be observed that the sample location $(0.3000, 0.5100)$ (triangle) allows for the smallest average Wasserstein distance for these two solutions.

Sampling budget adjustment

PMS allows the automatic reduction of the sampling budget used during an iteration of the EA if the archive already provides plenty of information on candidate individuals. The sampling budget is adaptively determined by the average Wasserstein distance over all individuals. We attempt to decrease this average distance monotonically. Considering a progressing evolution, we like to derive improved estimates for effective fitness. Given $V_{PMS}(z_{n,l})$ for $n \in \mathcal{N}$ and $l \in \Lambda$, the best sampling location can be identified as follows:

$$z^* = \arg\min\{V_{PMS}(z_{n,l})|\ n \in \mathcal{N}, l \in \Lambda\}. \tag{3.39}$$

Then, we can compute the Wasserstein distance $\bar{W}$ at the current population t as follows:

$$\bar{W}^t = \sum_{l \in \Lambda} \mathbb{1}_{\mathcal{R}_l}(z^*) M(z^* \cup \mathcal{S}_l, \hat{\mathcal{Z}}_l) + (1 - \mathbb{1}_{\mathcal{R}_l}(z^*)) M(\mathcal{S}_l, \hat{\mathcal{Z}}_l). \tag{3.40}$$

PMS stops sampling for the current population when the current average Wasserstein distance using the usable samples is less than the recorded average Wasserstein

distance of the previous population $t-1$. In other words,

$$\bar{W}^t < \bar{W}^{t-1}. \tag{3.41}$$

In practical use, we suggest to assign lower and upper limits for the number of new samples evaluated in each iteration. The lower limit ensures that a minimum additional knowledge on the underlying fitness landscape is collected in each iteration. It also prevents getting permanently stuck in an artificial optimum of a false approximation model. The upper bound prevents the spending of a large number of samples in the current population. Given that EAs are iterative search methods, the samples must also be allocated to the future population rather than extensively sampling the fitness for the current population.

3.4 Empirical Results

In this section, we examine the effect of varying algorithmic parameters on the performance of the PMS strategy. Moreover, we empirically compare the WASA strategies with several other ASA approaches from the literature using a variety of benchmark problems with different landscape features as well as a real-life robust design problem. All results are averaged over 30 independent runs with different random seeds.

3.4.1 Experimental Setup for Artificial Benchmark Problems

Overview of Artificial Benchmark Problems

We have chosen six artificial benchmark problems from the literature. Figure 3.6 provides one-dimensional visualisations of the original and effective fitness landscapes. The test problems have different characteristics:

- **TP 1**, taken from Paenke et al. [2006], has a discontinuous unimodal original fitness landscape. The peaks of the original and effective fitness landscapes are asymmetric and located next to each other. This problem characteristic can test an algorithm's ability to precisely identify the peak of effective fitness

at the discontinuous landscape.

$$f(x) = \min \quad 1 + \frac{1}{5}\sum_{i=1}^{5} Q_1(x_i), x_i \in [0, 10], \mathcal{U}(-0.5, 0.5)^5$$

with

$$Q_1(x_i) = \begin{cases} \frac{2-x_i}{6}, & x_i \leq 8 \\ 0, & \text{otherwise} \end{cases}$$

- **TP 2**, adopted from Paenke et al. [2006], can be viewed as the continuous version of TP 1.

$$f(x) = \min \quad 4.5 + \sum_{i=1}^{5} Q_2(x_i), x_i \in [0, 10], \mathcal{U}(-1, 1)^5$$

with

$$Q_2(x_i) = \begin{cases} -(8 - x_i)^{0.1} e^{-0.2(8-x_i)}, & x_i < 8 \\ 0, & \text{otherwise} \end{cases}$$

- **TP 3**, adopted from Branke [2001], has multimodal original and effective fitness landscapes. The global optimum of the original fitness ($x = 1$) is a local optimum of the effective fitness landscape, and vice versa. Poor effective fitness estimates in the early phase of optimisation may misdirect the EA towards the wrong region.

$$f(x) = \min \quad 5 - \sum_{i=1}^{5} Q_3(x_i), x_i \in [-2, 2], \mathcal{U}(-0.2, 0.2)^5$$

with

$$Q_3(x_i) = \begin{cases} -(x_i + 1)^2 + 1.4 - 0.8\,|sin(6.283x_i)| & -2 < x_i < 0 \\ 0.6 \cdot 2^{-8|x_i - 1|} + 0.958887 - 0.8\,|sin(6.283x_i)| & 0 \leq x_i < 2 \\ 0 & \text{otherwise} \end{cases}$$

- **TP 4**, taken from Paenke et al. [2006], has two global minima in the effective fitness landscape. The global minimum of the original fitness is a local

maximum of the effective fitness.

$$f(x) = \min \ \ 0.6 + \frac{1}{5}\sum_{i=1}^{5} Q_4(x_i), x_i \in [0,1], \mathcal{U}(-0.05, 0.05)^5$$

with

$$Q_4(x_i) = \begin{cases} -0.5e^{-0.5\frac{(x_i-0.4)^2}{0.05^2}}, & 0 \leq x_i < 0.4696 \\ -0.6e^{-0.5\frac{(x_i-0.5)^2}{0.02^2}}, & 0.4696 \leq x_i < 0.5304 \\ -0.5e^{-0.5\frac{(x_i-0.6)^2}{0.05^2}}, & \text{otherwise} \end{cases}$$

- **TP 5**, adapted from Paenke et al. [2006], combines the problem characteristics of TP 2 and TP 3.

$$f(x) = \min \ \ \sum_{i=1}^{5} Q_5(x_i) - 2, x_i \in [0,10], \mathcal{U}(-1,1)^5$$

with

$$Q_5(x_i) = 1 + 2sin(10e^{(-0.2x_i)}x_i)e^{(-0.25x_i)}$$

- **TP 6**, taken from Tsutsui and Ghosh [1997], has the global minima of the effective fitness landscape as being the local minima of the original fitness landscape.

$$f(x) = \min \ 1 - \frac{1}{5}\sum_{i=1}^{5} Q_6(x_i), x_i \in [0,1], \mathcal{U}(-0.0625, 0.0625)^5$$

with

$$Q_6(x_i) = \begin{cases} e^{-2(\frac{x_i-0.1}{0.8})^2 \ln 2} |\sin(5\pi x_i)|^{0.5}, & 0.4 < x_i \leq 0.6 \\ e^{-2(\frac{x_i-0.1}{0.8})^2 \ln 2} \sin^6(5\pi x_i), & \text{otherwise} \end{cases}$$

Evolutionary Algorithm

Essentially, the archive-based approximation is a part of the EA's fitness evaluation procedure. Therefore, it is straightforward to integrate our proposed sampling strategies into any evolutionary algorithm. We have adopted the covariance matrix adaptation evolution strategy (CMA-ES), proposed by Hansen and Ostermeier [2001], for the experiments. CMA-ES is one of advanced single-objective evolutionary strategies. We implement the CMA-ES algorithm that is based on the MATLAB

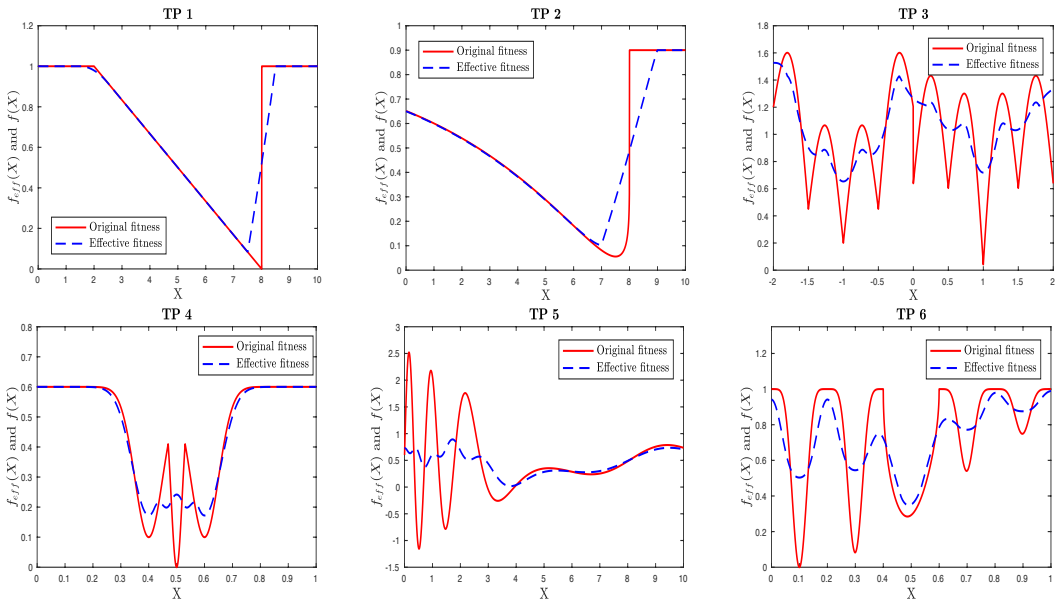

Figure 3.6: 1-D visualisation of original and effective fitness of the 5-D test problems. Solid line: original fitness landscape. Dash line: effective fitness landscape.

CMA-ES Parameters	Selected Settings
(μ, λ)	$(4, 8)$
initial standard deviation	$\frac{1}{4}$ search interval width
recombination	equally weighted recombination
initial point	centre of search interval
total number of evaluations	$1,600$

Table 3.1: The CMA-ES parameter settings.

toolbox [Hansen]. We have modified the fitness evaluation procedure in this toolbox and integrated the various sampling strategies. The used parameter settings of the CMA-ES toolbox are listed in Table 3.1.

Solution Selection & Performance Measure

We employ the best observed individual, i.e. the individual with best estimated effective fitness, at the final generation as the solution that would be reported to the decision-maker. The actual effective fitness of the selected solution is evaluated by $10,000$ Monte-Carlo samples generated from the underlying noise. Moreover, we look at the algorithm's convergence and the average effective fitness over the whole run. Ideally, we would expect the sampling approach to have a fast convergence and to provide a high-quality solution at the final generation.

Disturbance Generation for WASA

The performance of WASA is closely associated with the way the approximate uncertainty set $\hat{\mathcal{Z}}_l$ is generated. We use 243 Latin hypercube disturbances for all individuals within a generation, which reduces the variance in comparing the effective finesses in an uncertain environment. To avoid over-fitting to a specific set of disturbances, we change the set of disturbances at each iteration of the EA.

3.4.2 Performance of the PMS Strategy with Various Approximation Region Parameters

We numerically study the effect of using various approximation region parameters (κ=100%, 120% and infinity). The lower and upper limits of the sampling budget at each iteration are fixed at 4 and 8, respectively. Figure 3.7 presents the convergence of the average effective fitness of the best observed solution.

As can be seen, the convergence towards high-performance solutions can be accelerated by using a proper approximation region parameter. Given the same amount of evaluations, PMS-120% performs better than PMS-100%. The superiority of using PMS-120% is significant before $1,000$ evaluations, because early in the run, information in the archive is sparse, and the wider approximation region can use more archive samples in the individual's effective fitness estimation. At the end of the run, the PMS-120% strategy is able to find better solutions than all other strategies in all test problems, indicating that a proper setting of the approximation region prevents CMA-ES from early convergence towards local optima by incorporating more archive samples in the effective fitness estimation.

In all benchmark problems, the average effective fitness of using PMS-infinity deteriorates during the first 100 evaluations, though this value is improved quickly at later evaluations. The reason is that the new sampling locations determined by PMS-infinity are always located at the centre of a cluster of individuals when the archive is empty or contains only few samples, which actually misleads the EA search process. This negative effect becomes serious in test problems TP 5 and TP 6, because the problem characteristics demand that the robust approach is able to identify the correct search area early on. Otherwise, the EA search process converges towards local optima.

To support Figure 3.7, Table 3.2 reports on the average effective fitness obtained over all $1,600$ evaluations of the run. Again, we can observe that PMS-120% provides the fastest convergence pattern among these three strategies for all test functions. PMS-infinity displays good convergence in TP 1, 2 and 4. Addition-

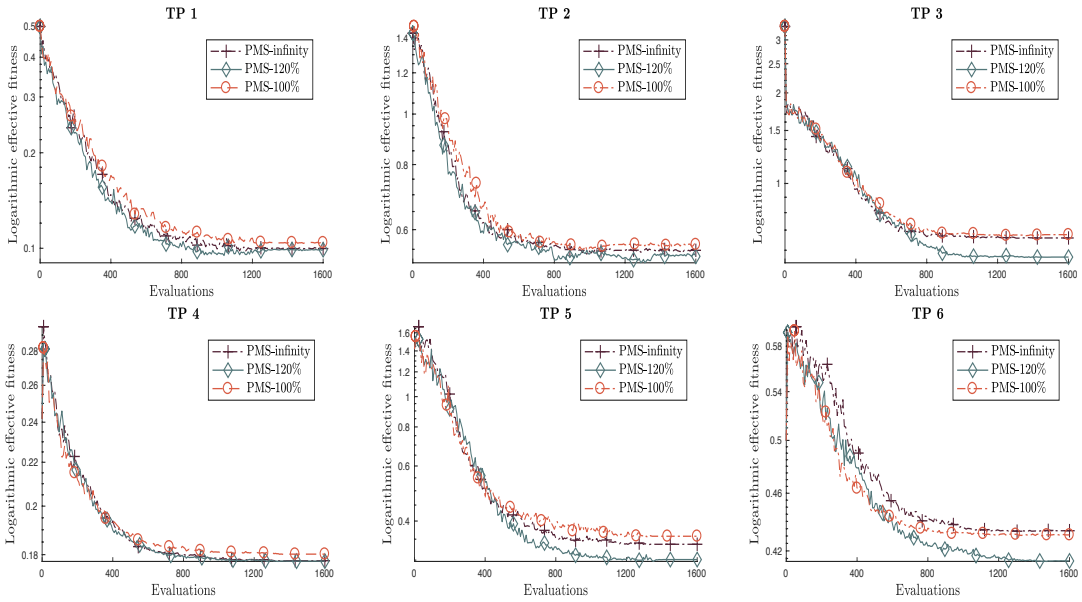

Figure 3.7: Convergence comparison with respect to evaluations for various approximation region parameters.

Table 3.2: Average effective fitness over 1,600 evaluations.

Test Problem	Mean ± Std. err.		
	PMS-100%	PMS-120%	PMS-infinity
TP 1	0.150±0.003	**0.140±0.003**	**0.145±0.002**
TP 2	0.674±0.014	**0.634±0.014**	**0.651±0.014**
TP 3	0.901±0.015	**0.848±0.025**	0.890±0.021
TP 4	**0.192±0.001**	**0.190±0.001**	**0.191±0.001**
TP 5	0.528±0.009	**0.495±0.018**	0.527±0.022
TP 6	0.458±0.006	**0.452±0.006**	0.469±0.008

Best results and those statistically not different from best are highlighted in bold.

ally, Table 3.3 presents the performance of the final solution at the end of the EA search process. These results show that PMS-120% has the best performance in all benchmark problems, and PMS-infinity presents a performance advantage over PMS-100%. However, PMS-infinity performs worse than PMS-120% in most of the test functions. The results confirm the theoretical ground discussed in Section 3.3, namely that a moderate increase of the approximation region beyond the disturbance region is useful, but too much may lose its benefit.

Table 3.3: Effective fitness of the solution at 1,600 evaluations.

Test Problem	Mean ± Std. err.		
	PMS-100%	PMS-120%	PMS-infinity
TP 1	**0.104±0.006**	**0.098±0.002**	**0.099±0.001**
TP 2	0.561±0.007	**0.533±0.006**	0.548±0.005
TP 3	0.676±0.031	**0.569±0.022**	0.659±0.027
TP 4	0.182±0.002	**0.177±0.002**	**0.177±0.001**
TP 5	0.358±0.011	**0.301±0.009**	0.337±0.004
TP 6	0.431±0.009	**0.413±0.006**	0.436±0.007

Best results and those statistically not different from best are highlighted in bold.

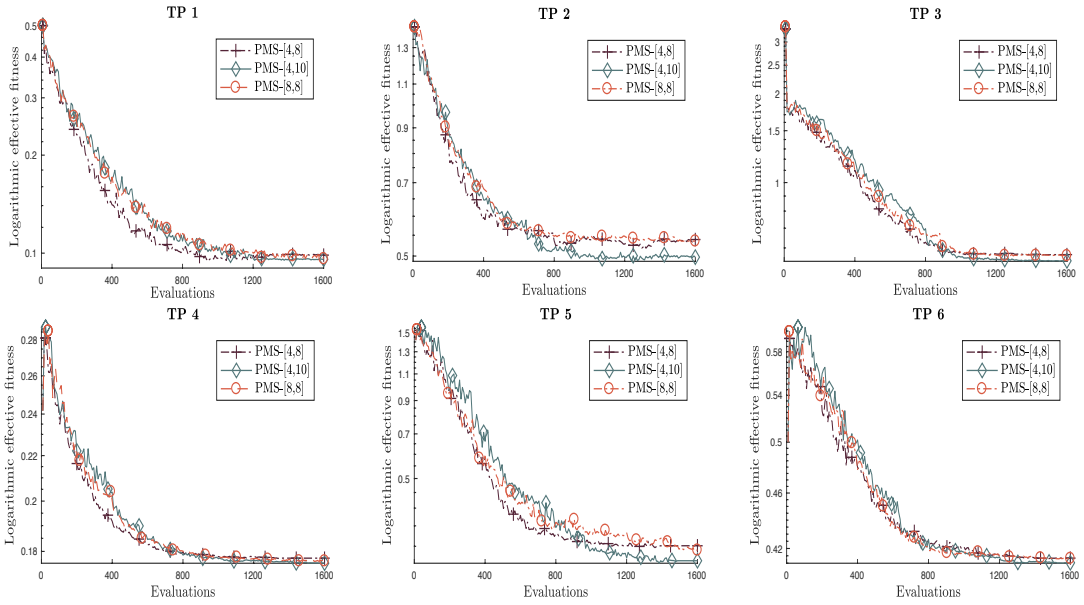

Figure 3.8: Convergence comparison with the evaluations for various sampling budget limits.

3.4.3 Performance of the PMS Strategy with Various Sampling Budget Limits

In this experiment, we investigate the convergence pattern with varying lower and upper sampling budget limitations. We consider the setting where the sampling budget per generation is allowed to change between 4 and 8 as default, which is abbreviated as the PMS-$[4, 8]$ strategy. We firstly test the effect of rising the lower limit of PMS-$[4, 8]$. As such, we consistently use 8 evaluations throughout the search process and denote this strategy as PMS-$[8, 8]$. Secondly, we study the impact of increasing the upper limit on the convergence behaviour. Therefore, we include PMS-$[4, 10]$ in this experiment, whose upper limit is set to 10. The approximation region is fixed at 120% of each direction of an individual's perturbed region. Fig-

Table 3.4: Average effective fitness over 1,600 evaluations.

Test Problem	Mean ± Std. err.		
	PMS-[4, 8]	PMS-[4, 10]	PMS-[8, 8]
TP 1	**0.140±0.003**	0.152±0.002	0.151±0.003
TP 2	**0.634±0.014**	**0.628±0.012**	**0.629±0.005**
TP 3	**0.848±0.025**	0.897±0.021	**0.876±0.027**
TP 4	**0.190±0.001**	0.194±0.001	0.193±0.001
TP 5	**0.495±0.018**	0.540±0.029	0.514±0.028
TP 6	**0.452±0.006**	**0.457±0.008**	**0.453±0.008**

Best results and those statistically not different from best are highlighted in bold.

Table 3.5: Effective fitness of the solution at 1,600 evaluations.

Test Problem	Mean ± Std. err.		
	PMS-[4, 8]	PMS-[4, 10]	PMS-[8, 8]
TP 1	0.098±0.002	**0.094±0.001**	**0.093±0.003**
TP 2	0.533±0.006	**0.498±0.009**	0.535±0.002
TP 3	**0.569±0.022**	**0.542±0.028**	**0.565±0.031**
TP 4	**0.177±0.002**	**0.175±0.001**	**0.176±0.001**
TP 5	0.301±0.009	**0.269±0.018**	**0.294±0.016**
TP 6	**0.413±0.006**	**0.410±0.008**	**0.413±0.008**

Best results and those statistically not different from best are highlighted in bold.

ure 3.8 displays the results of the various sampling strategies for all test problems.

As shown in Figure 3.8, PMS-[4, 8] and PMS-[8, 8] have similar convergence patterns at early search iterations because both strategies implement eight evaluations when the archive has few usable samples. Nevertheless, the convergence of PMS-[4, 8] becomes faster than that of PMS-[8, 8] once the archive has a sufficient number of samples, indicating that PMS-[4, 8] saves evaluations when the average Wasserstein distance is monotonically decreasing anyway. Moreover, we observe that the convergence of PMS-[4, 10] is slow at early iterations. This is because PMS-[4, 10] is allowed to spend more evaluations exploring the correct search directions at early iterations.

Again, Table 3.4 and 3.5 summarise the overall performance and the effective fitness value of the final solution, respectively. We observe that strategies PMS-[4, 8] and PMS-[8, 8] perform similarly at the end of EA search process, but PMS-[4, 8] converges faster than PMS-[8, 8], indicating that our sampling budget adjustment can save evaluations without sacrificing the performance of the final solution. PMS-[4, 10] provides the best final solution in all test problems, but converges more slowly.

3.4.4 Average Performance Comparison

We verify the performance of the proposed strategies, namely EFS and PMS, by comparing with the following archive-based approaches from the literature:

1. **SEM**: The strategy randomly samples one location within the individual's perturbed area [Tsutsui and Ghosh, 1997].

2. **SEMAR**: This is the SEM strategy integrated with the archive sample approximation strategy [Branke, 2001].

3. **ABRSS**: This strategy is based on an archive sample approximation approach consisting of two main steps. For each disturbed location, the first step is to identify the closest sample point in the archive, and the second step is to check whether this disturbed location is also the closest disturbed location of its selected archive sample point [Kruisselbrink et al., 2010]. If this is the case, the selected archive sample point will be used in the effective fitness estimation; otherwise, this disturbed location will be considered for an additional sampling.

4. **ABRSS+OP**: This strategy implements ABRSS to determine the additional sample points, but assigns the optimal probabilities that are obtained from the Wasserstein distance for all sample points involved in the effective fitness estimation.

Note that all sampling strategies are inserted into the same CMA-ES, so that all performance differences can be attributed to the sampling strategy alone.

We first examine the reduction in the approximation error achieved by various archive-based strategies. Given that SEM does not employ previously evaluated samples, this approach is excluded. We randomly generate 20 solutions and 30 available archive samples in this experiment. Second, we compute the average approximation error over these solutions before and after employing any sampling strategy. The sampling budgets for all strategies are identical and fixed at 20. Figure 3.9 summarises the changes of approximation error between before and after the sampling strategy implementation. EFS and PMS strategies effectively decrease the average approximation error for all test problems. The approximation errors obtained by ABRSS, ABRSS+OP, and SEMAR are larger than those obtained by those WASA strategies. Moreover, the results show that the non-WASA strategies, in several functions, cannot successfully reduce the approximation error, e.g., their approximation errors slightly increase in TP 5. The reason is that sampling bias

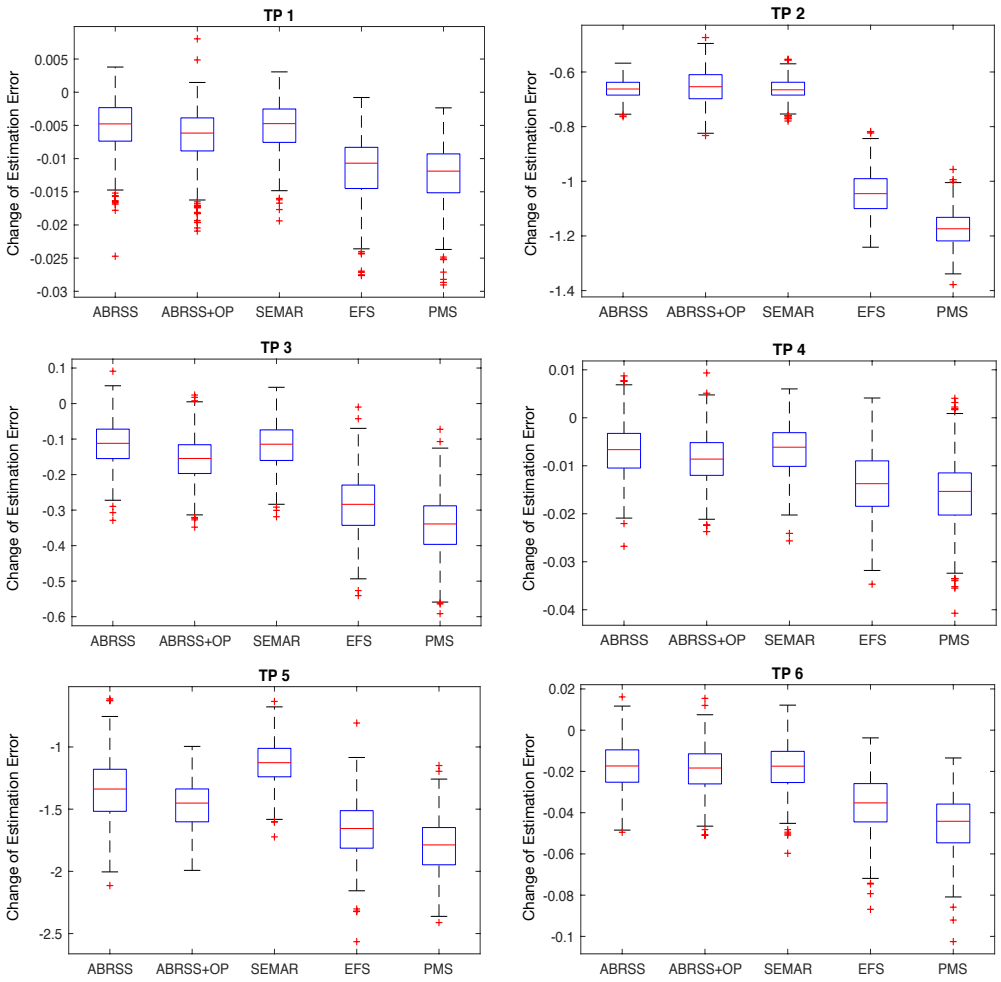

Figure 3.9: Comparison of approximation error reduction achieved various sampling strategies.

may affect the benefit of including additional samples. Assuming that the archive samples around a specific solution are negatively biased, if we include additional negatively biased samples and simply average these samples, the approximation would degrade. Nevertheless, the Wasserstein metric in WASA strategies, to some extent, diminish the bias effect by adjusting the probabilities of archive samples and new samples.

Next, we study the convergence of various sampling strategies in the CMA-ES search process. The PMS strategy sets the approximation region parameter κ as 120% and fixes the lower and upper limits for the number of samples per iteration at 4 and 8, respectively. Figure 3.10 displays the computational results of using various bounds for the sampling budget. Moreover, in Figure 3.11, we report the

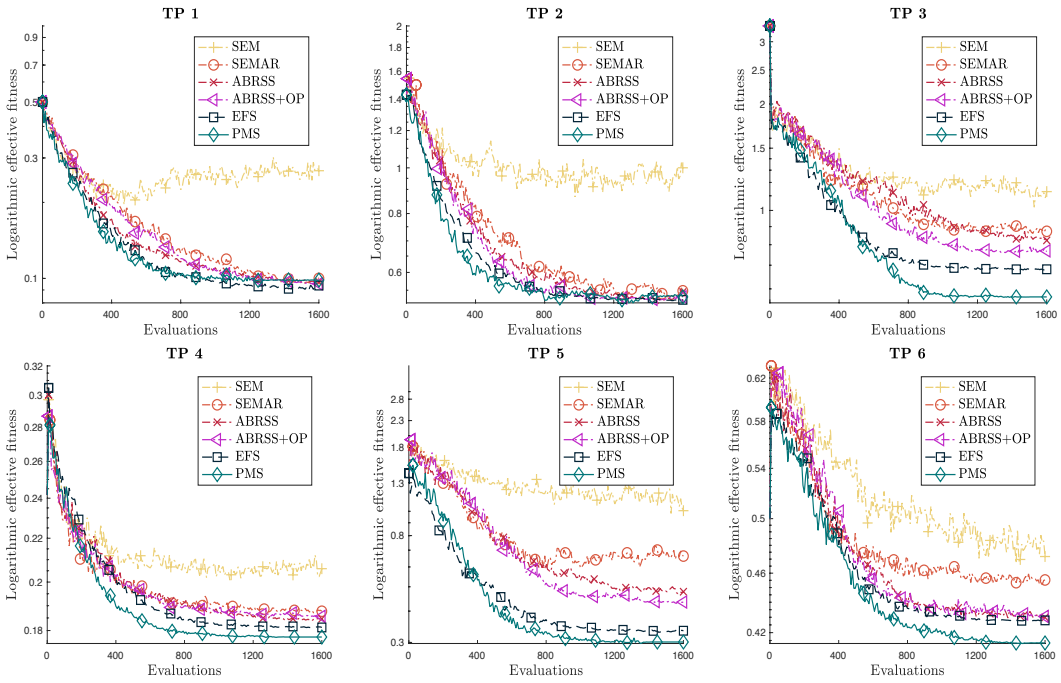

Figure 3.10: Convergence comparison of different sampling strategies with respect to evaluations.

average approximation error over the course of the run, which is defined by the mean squared error between the true and estimated effective fitness.

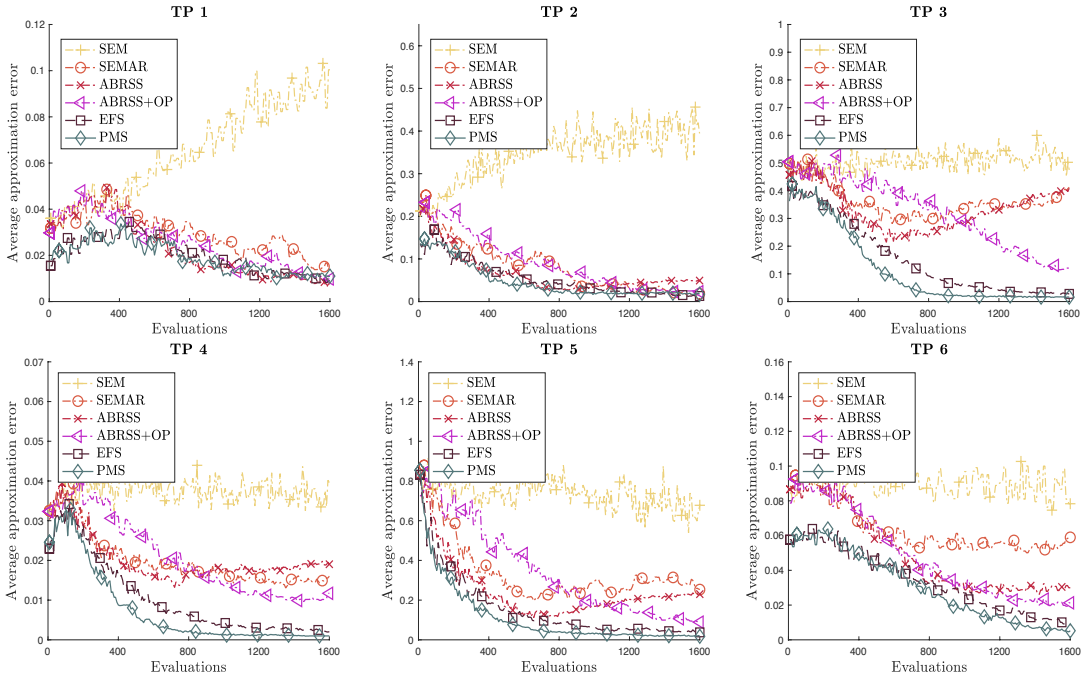

Figure 3.11: Average approximation error comparison of different sampling strategies with respect to evaluations.

Table 3.6: Average effective fitness over 1,600 evaluations.

Test Problem	Mean ± Std. Err.					
	SEM	SEMAR	ABRSS	ABRSS+OP	EFS	PMS
TP 1	0.260±0.006	0.171±0.007	0.156±0.004	0.160±0.006	0.144±0.003	**0.140±0.003**
TP 2	1.018±0.008	0.728±0.017	0.702±0.017	0.676±0.010	**0.625±0.015**	**0.634±0.014**
TP 3	1.304±0.024	1.121±0.035	1.164±0.031	1.012±0.040	0.902±0.035	**0.848±0.025**
TP 4	0.213±0.001	0.199±0.001	0.200±0.001	0.198±0.001	0.196±0.001	**0.190±0.001**
TP 5	1.280±0.010	0.853±0.021	0.783±0.022	0.753±0.031	0.497±0.018	**0.495±0.018**
TP 6	0.519±0.007	0.485±0.008	0.471±0.008	0.474±0.011	**0.462±0.007**	**0.452±0.006**

Best results and those statistically not different from best are highlighted in bold.

Table 3.7: Effective fitness of the solution at 1,600 evaluations.

Test Problem	Mean ± Std. Err.					
	SEM	SEMAR	ABRSS	ABRSS+OP	EFS	PMS
TP 1	0.267±0.020	**0.100±0.006**	**0.098±0.003**	**0.096±0.006**	**0.095±0.003**	**0.098±0.002**
TP 2	1.000±0.052	0.548±0.011	0.554±0.024	**0.535±0.007**	**0.528±0.002**	**0.533±0.006**
TP 3	1.131±0.075	0.873±0.054	0.823±0.071	0.769±0.042	0.682±0.031	**0.569±0.022**
TP 4	0.205±0.004	0.187±0.009	0.184±0.003	0.185±0.002	**0.181±0.003**	**0.177±0.002**
TP 5	1.005±0.076	0.662±0.044	0.479±0.057	0.434±0.053	0.335±0.012	**0.301±0.009**
TP 6	0.471±0.013	0.455±0.011	0.429±0.009	0.430±0.011	0.428±0.005	**0.413±0.006**

Best results and those statistically not different from best are highlighted in bold.

As shown in Figure 3.10, the SEM approach provides the worst results for all test problems, because this approach does not use an archive and one new sample is not sufficient to estimate the effective fitness. Compared to SEM, the archive-based approaches show a good convergence on most test problems. SEMAR and ABRSS exhibit similar convergence behaviour on most test problems. Since the ABRSS+OP approach implements the optimal probabilities, its convergence is faster than ABRSS and SEMAR in the most of test problems. Nevertheless, ABRSS+OP performs worse than the WASA-based strategies, i.e., EFS and PMS, indicating that the Wasserstein distance metric provides an advantage in selecting good sampling locations. Additionally, the results demonstrate the good performance of PMS which converges faster than EFS over the first $1,000$ evaluations in all problems.

The effective fitness of the best observed solution obtained from various strategies averaged over $1,600$ evaluations is reported in Table 3.6. The results confirm the importance of good sampling strategy design. We find that the WASA sampling strategies converge faster than other methods. For strategies PMS and EFS, we observe that the convergence of PMS is more rapid than EFS for all test problems except TP 2 and 6. Table 3.7 displays the performance of the final solution obtained from various approaches. The result once again confirms the superiority of WASA sampling strategies and the performance advantage of PMS over EFS.

The results of average estimation error in Figure 3.11 confirm our findings from previous convergence comparisons. The SEM approach consistently exhibits large effective fitness estimation errors. The SEMAR and ABRSS strategies reduce the average estimation error when the number of evaluations is small, because they reuse past sampling information. However, SEMAR and ABRSS lack a good mechanism to determine the probability weights for the samples used in the effective fitness estimation. This deficiency might lead to biases in the estimation. In some cases, the average estimation errors of SEMAR and ABRSS are actually increasing over the run. By contrast, the approaches that adopt optimal probabilities, i.e. ABRSS+OP, EFS and PMS, keep decreasing the average estimation error throughout the run; and PMS is the fastest approach in decreasing the average estimation error.

3.4.5 Robust Multi-point Airfoil Shape Optimisation under Uncertain Manufacturing Errors

Finally, we test the proposed strategies on a simple real-world problem. We consider a multi-point airfoil shape optimisation problem with manufacturing errors. Although advances in high-performance computing have reduced the CPU time with

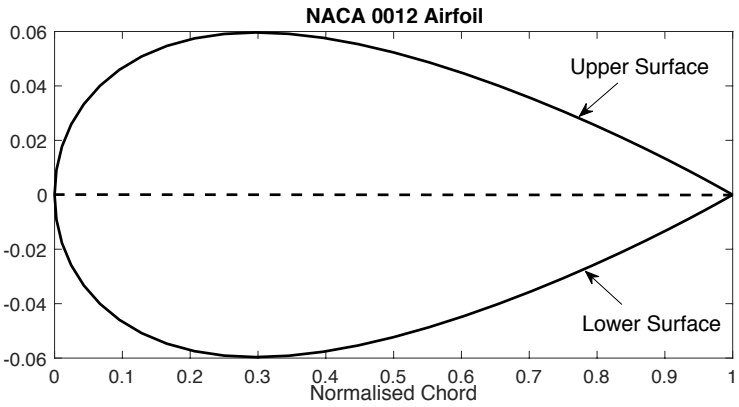

Figure 3.12: The baseline shape of the robust design problem.

respect to the performance evaluation of airfoil shape, this robust optimisation problem still requires a significant computational effort because algorithms might need thousands of evaluations under various possible manufacturing errors. Therefore, it is an ideal testbed for examining the performance of various sampling approaches under the condition of limited evaluation budget.

We consider the subsonic 2-D airfoil design problem, which is a variant obtained from Ong et al. [2006]. The baseline shape is NACA 0012 airfoil [Abbott and Von Doenhoff, 1959], which is illustrated in Figure 3.12. We implement 10 Hicks-Henne bump functions, for details see [Hicks and Henne, 1978], $f_i(z)$ for $i = 1, \ldots, 10$ with the upper and lower surfaces of NACA 0012 (denoted as $y_u^b(z)$ and $y_l^b(z)$) to parameterise the upper surface $y_u(z)$ and the lower surface $y_l(z)$. They are defined as follows:

$$y_u(z) = y_u^b(z) + \sum_{i=1}^{6} \theta_i f_i(z)$$

and

$$y_l(z) = y_l^b(z) + \sum_{i=7}^{10} \theta_i f_i(z),$$

where z is a non-dimensional abscissa, and θ_i is the design variable on the i-th Hicks-Henne bump function. These Hicks-Henne bump functions combined with design variables can be used to tune the upper and lower surfaces. The definitions of Hicks-Henne bump functions and the ranges of the design variables can be found as follows.

$$f_1(z) = sin^4(\pi z^{\ln 0.5/\ln 0.05}), \quad \theta_1 = [-0.001, 0.001]$$

$$f_2(z) = sin^4(\pi z^{\ln 0.5/\ln 0.15}), \quad \theta_2 = [-0.006, 0.006]$$

$$f_3(z) = sin^4(\pi z^{\ln 0.5/\ln 0.30}), \quad \theta_3 = [-0.009, 0.009]$$
$$f_4(z) = sin^4(\pi z^{\ln 0.5/\ln 0.45}), \quad \theta_4 = [-0.009, 0.009]$$
$$f_5(z) = sin^4(\pi z^{\ln 0.5/\ln 0.60}), \quad \theta_5 = [-0.006, 0.006]$$
$$f_6(z) = sin^4(\pi z^{\ln 0.5/\ln 0.80}), \quad \theta_6 = [-0.002, 0.002]$$
$$f_7(z) = sin^4(\pi z^{\ln 0.5/\ln 0.10}), \quad \theta_7 = [-0.001, 0.001]$$
$$f_8(z) = sin^4(\pi z^{\ln 0.5/\ln 0.10}), \quad \theta_8 = [-0.007, 0.007]$$
$$f_9(z) = sin^4(\pi z^{\ln 0.5/\ln 0.55}), \quad \theta_9 = [-0.007, 0.007]$$
$$f_{10}(z) = sin^4(\pi z^{\ln 0.5/\ln 0.80}), \quad \theta_{10} = [-0.002, 0.002]$$

The "fitness" of the airfoil shape is defined as the average lift-to-drag ratio over three flow velocities ($\mathbf{M}_1 = 0.5$ mach, $\mathbf{M}_2 = 0.55$ mach and $\mathbf{M}_3 = 0.6$ mach) when Reynolds number (Re) and Angle of Attack (AoA) are $300,000$ and $4°$, respectively. Thus, the fitness function $f(\theta)$ can be written as follows:

$$f(\theta) = \frac{1}{3}(C_1 + C_2 + C_3)$$
where
$$C_1 = C_{L/D}(\mathbf{M}_1, Re, AoA, \theta_1, \cdots, \theta_{10})$$
$$C_2 = C_{L/D}(\mathbf{M}_2, Re, AoA, \theta_1, \cdots, \theta_{10})$$
$$C_3 = C_{L/D}(\mathbf{M}_3, Re, AoA, \theta_1, \cdots, \theta_{10})$$

where $C_{L/D}$ denotes the lift-to-drag ratio. We assume that all design variables are affected by uniformly disturbed manufacturing errors. The range of manufacturing error for all design variables is fixed as $[-0.001, 0.001]$. The overall robust design problem can be formulated as

$$\max \quad f_{eff}(\theta) = \mathbb{E}[f(\theta + \xi)] = \mathbb{E}\big[\frac{1}{3}(\hat{C}_1 + \hat{C}_2 + \hat{C}_3)\big]$$
where
$$\hat{C}_1 = C_{L/D}(\mathbf{M}_1, Re, AoA, \theta_1 + \xi_1, \cdots, \theta_{10} + \xi_{10})$$
$$\hat{C}_2 = C_{L/D}(\mathbf{M}_2, Re, AoA, \theta_1 + \xi_1, \cdots, \theta_{10} + \xi_{10})$$
$$\hat{C}_3 = C_{L/D}(\mathbf{M}_3, Re, AoA, \theta_1 + \xi_1, \cdots, \theta_{10} + \xi_{10})$$
$$\xi_i \in \mathcal{U}(-0.001, 0.001), \quad i = 1, \ldots, 10.$$

In this experiment, the airfoil shape of lift-to-drag ratios at various Mach

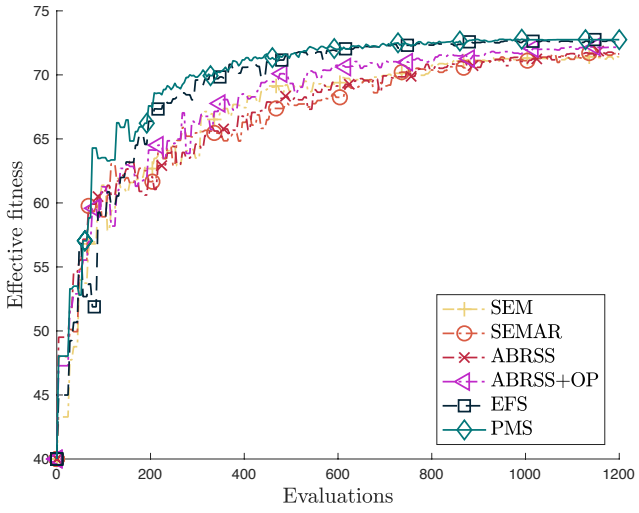

Figure 3.13: Convergence comparison of various sampling strategies with respect to evaluations in the robust airfoil shape optimisation problem.

numbers are evaluated by Drela's XFOIL [Drela, 1989]. This software is an open-source aerodynamic analysis package for subsonic isolated airfoils, which allows the use of relatively lower computational effort than advanced CFD programs. For this problem, we fix the (μ, λ) parameters in CMA-ES at $(5, 10)$ and choose the total evaluations as 1,200. The lower and upper limits of the evaluation budget at each iteration for the PMS strategy are fixed at 5 and 10, respectively; whereas the evaluation budget of other sampling approaches is fixed at 10, that is, each solution is evaluated once. The other experimental setup is exactly same as in previous experiments. Figure 3.13 shows the convergence of the average effective fitness of the currently best solution based on the estimated effective fitness values.

As can be seen in Figure 3.13, the convergence of WASA sampling strategies is considerably faster than other sampling approaches, which clearly confirms the superiority of WASA sampling strategies in this real-world problem. The non-WASA sampling strategies (SEM, SEMAR, ABRSS and ABRSS+OP) exhibit similar convergence patterns. For EFS and PMS, the solutions at 400 evaluations are even better than the solutions obtained from the other sampling approaches after $1,200$ evaluations. In comparison to EFS, we observe that PMS provides a good convergence rate at the first 200 evaluations due to its advanced sample selection method in the WASA framework, which is consistent with the results of artificial test functions presented in Section 3.4.

Table 3.8 reports both the average effective fitness over 1,200 evaluations (abbreviated as A.E.F.) and the effective fitness of the solution at 1,200 evaluations

Table 3.8: Performance of various sampling approaches in the robust airfoil shape optimisation problem.

Method	Mean ± Std. Err.	
	A.E.F.	E.F.
SEM	67.665±0.434	71.410±0.434
SEMAR	66.140±1.011	71.657±0.305
ABRSS	66.359±0.721	71.638±0.311
ABRSS+OP	67.742±0.987	72.035±0.277
EFS	68.453±0.239	72.614±0.089
PMS	**69.257±0.233**	**72.780±0.068**

Best results and those statistically not different from best are highlighted in bold.

(abbreviated as E.F.). The results show that PMS provides the fastest convergence among the six approaches. The convergence of EFS is worse than that of PMS but still much better than those of the non-WASA approaches. The optimal probabilities used in the effective fitness estimation have improved performance of ABRSS+OP in this test problem. Compared to ABRSS and SEMAR, ABRSS+OP has a faster convergence and provides better final solutions, though it is still worse than the WASA sampling strategies.

3.5 Conclusions

When using evolutionary algorithms to search for a robust solution, estimating the effective fitness is challenging. Storing previous evaluations in an archive and using this information to improve the fitness estimate, the so-called archive sample approximation method, has been proposed by several authors to improve the estimation accuracy without increasing the sampling budget. The crucial part is the sampling strategy, i.e., to decide what new solutions should be evaluated. In this chapter, we used the Wasserstein distance metric to approximate an upper bound for the error and proposed two Wasserstein-based sampling strategies to suggest promising sampling locations. Minimising the upper bound cannot guarantee minimisation of the actual error, however, knowing that we have no information on the fitness function apart from previous samples, it is a promising approach. One sampling strategy considers the sampling contribution from each individual's perspective, and allocates an equal number of evaluations to each individual. The second strategy approximates the sampling contribution for all individuals. The empirical results on various benchmark problems demonstrate the benefit of using our new sampling strategies and the advantage of considering the population contribution.

Chapter 4

Efficient Rollout Algorithms for Clinical Trial Scheduling and Resource Allocation in Drug Development under Uncertainty

4.1 Introduction

Research and Development (R&D) investments in new drug development have dramatically increased over the last few decades due to growing public healthcare demands [Paul et al., 2010]. However, pharmaceutical companies are struggling in recouping their investments due to the low R&D productivity. The average return on R&D investments over the mid-to-large cap pharmaceutical companies dropped from 17.4% to 11.9% during $2013-2017$ [Terry and Lesser, 2018]. No single cause of this recession in profitability has yet been identified because the new drug development process is influenced by a set of complex factors. For example, cycle times have steadily increased. The pursuit of new treatments in complex disease areas requires an additional time. Statistical studies have shown that the median interval duration of passing all clinical tests was 5.9 years during $1999-2001$ while this figure increased to 9.1 years for the period of $2008-2012$ [Kaitin and Cairns, 2003; Schuhmacher et al., 2016]. In addition to this, innovative treatments are often associated with low success rates. Hay et al. [2014] showed that only one of ten experimental drugs was finally approved by the regulatory authority. Not surprisingly, long cycle times coupled with low success rates translate into high overall expenditures. Paul et al. [2010] noted that "*without a dramatic increase in productivity, today's*

pharmaceutical industry cannot sustain sufficient innovation to replace the loss of revenues due to patent expirations for successful products". In this context, research on how to improve productivity will be highly beneficial for R&D managers in the pharmaceutical industry.

The development of new drugs involves important decisions such as clinical trial scheduling and resource allocation. A drug's clinical trial consists of three stages, namely Phases I, II and III. As soon as an experimental drug passes Phase III, the firm can submit a new drug application to the regulatory authority for a market approval. Nevertheless, the economic value obtained from an approved drug is to be decreased because the patent protection for innovative drugs generally lasts up to 20 years. Once a drug patent expires, a dramatic decline in product revenue may occur because generic alternatives (i.e., a copied drug that works similarly to the brand-name drug) may enter the market at a considerably lower price [Munos, 2009]. The pharmaceutical companies are thus usually eager to complete all requested clinical trials within the shortest time possible to recoup their R&D investments during the window spanning between a drug's commercial launch and its patent expiry. The main challenge is the uncertainty for resource planning. If an experimental drug fails at a particular test, then it will be withdrawn from the drug portfolio, and any further testing will be abandoned. Therefore, R&D managers must consider the consequence that some trial failures will seriously damage the overall profitability of the drug pipeline.

The decision-theoretic models have attracted considerable interest from the pharmaceutical industry because of their ability to improve the efficiency of drug project management and to facilitate timely completion of a drug project. Related studies mostly assume that the resource level of each trial is fixed before it starts, and additional resources are not allowed during the trial execution, e.g., see Schmidt and Grossmann [1996]; De Reyck and Leus [2008] and Colvin and Maravelias [2008]. Nevertheless, the new drug development is a race against time, and additional resources may be required to accelerate ongoing trials and maximise product revenue during the exclusive marketing period. For example, pharmaceutical companies may compress the development time by hiring additional staff, operating many test sites, or advertising medical research via social media and television commercials to attract clinical trial volunteers. Although these acceleration measures request further resources and increase overall cost, they may provide possible benefits of completing the drug project within a shortened time span. R&D managers can gain additional strategic flexibility by adjusting the resource level for ongoing trials to account for cases when a drug pipeline has idle resources.

In this study, we investigate effective drug development strategies by using a discrete-time finite Markov decision process (MDP) model. We propose a rollout algorithm that applies an adaptive sampling approach to assign the simulation budget for candidate actions and determines the action to be taken at each state of a decision tree. Through this sampling approach, a search tree is expanded by using an optimistic policy for estimating possible outcomes after the selected actions are implemented. Our contributions to the current literature in terms of modelling and solution techniques are threefold.

- First, we consider a clinical trial scheduling and resource allocation (CTSRA) problem, wherein the success or failure of a clinical trial is uncertain and the revenue to be received for drug projects depends on their approval times. We provide a stochastic dynamic programming model to formulate the CTSRA problem which aims to maximise the expected profit gained from drug projects over the exclusive marketing period. We assume that a firm has a finite amount of resources to complete drug projects, each of which comprises three clinical trials. Each clinical trial requires a certain amount of time to complete testing and has a specific resource requirement to ensure that the test proceeds smoothly. The firm can apply acceleration measures to compress the development time during the execution of a clinical trial on the ground of future potential profits. To the best of our knowledge, this clinical trial scheduling and resource allocation problem has not been considered in the literature yet.

- Second, we investigate problem-specific structures of the CTSRA model and propose an *optimistic policy* to guide the look-ahead search of rollout algorithms in a decision tree. The optimistic policy determines the actions for unexplored states in a rolling horizon fashion and provides an upper bound estimation of expected cumulative reward for rollout algorithms to identify the best action to be taken at the current state. We prove that the optimistic policy follows a sequential consistency property, thereby indicating that the performance of rollout algorithm is at least as good as the optimistic policy.

- Third, we introduce an *adaptive sampling* approach that exploits a variance reduction technique of common random numbers (CRN) as well as the empirical Bernstein inequality in a statistical racing procedure to balance the exploration and exploitation of the rollout algorithm. Specifically, the adaptive sampling approach calculates all pairwise differences between distinct candidate actions and applies the empirical Bernstein inequality to infer the confidence intervals

of each pairwise difference to be a certain significance level. Then inferior candidate actions are not re-evaluated during the simulation. This procedure is repeated until the simulation budget is exhausted. We further tailor the adaptive sampling strategy by developing a heuristic-based grouping rule that classifies candidate actions into separate groups on the basis of their resource usages. Our numerical experiments show that the adaptive sampling strategy integrated with the heuristic-based grouping rule can improve the efficiency of finding better actions.

The remainder of this chapter is structured as follows. In Section 4.2, we review the related studies. Section 4.3 introduces the MDP formulation used in this study. Section 4.4 explains the overall rollout algorithm framework. We describe the optimistic policy in Section 4.5 and the adaptive sampling strategies in Section 4.6. In Section 4.7, we study the efficacy of the proposed methods. Section 4.8 concludes the chapter by summarising our findings.

4.2 Review of Related Literature

The new drug development problem has been broadly considered as a resource-constrained project scheduling problem with uncertain trial outcomes and widely studied since the early 1990s. Schmidt and Grossmann [1996] applied a mixed-integer linear program to formulate a drug development problem with stochastic activity durations and costs. Choi et al. [2004] developed a dynamic programming formulation of a clinical trial scheduling problem with stochastic activity durations and proposed several state reduction approaches to improve efficiency of the approximate solution algorithm. Zapata et al. [2008] developed a simulation-optimisation framework for a drug development problem with discrete time-resource trade-offs, which can be seen as an extension of classical drug development problems in which the duration of each trial depends on resource usage. The resource allocation for on-going trials in this study cannot be adjusted, which might restrict the strategic flexibility of R&D managers. De Reyck and Leus [2008] introduced stochastic programming approaches in drug development with deterministic activity durations. Verderame et al. [2010] and Christian and Cremaschi [2015] later proposed several heuristic techniques to solve the stochastic programming models. Kouvelis et al. [2017] considered continuous-time MDPs to address an optimal investment problem for drug projects in Phase III clinical tests.

Among the solution methods, the rollout algorithm, which was first introduced by Bertsekas et al. [1997], is a lookahead search technique used to approxi-

mately solve MDP models. The success of this algorithm mainly relies on creative use of a base policy to estimate future outcomes after a candidate action has been taken at a particular state. Therefore, the algorithm is sensitive to estimation errors caused by the base policy. Bertsekas et al. [1997] considered deterministic MDPs and showed that the rollout algorithm does not degrade the performance of the base policy when this policy satisfies sequentially improving or consistency properties. Secomandi [2008] generalised these results for stochastic MDPs. These studies also showed that the rollout algorithm is comparable to other approximate dynamic programming approaches for addressing various real-world problems, such as the multidimensional knapsack problem [Bertsimas and Demir, 2002], revenue management [Bertsimas and Popescu, 2003] and transportation problem [Secomandi, 2001; Goodson et al., 2013]. For a comprehensive review on the rollout algorithm, the interested readers are referred to Goodson et al. [2017].

To the best of our knowledge, the sampling approaches within the rollout algorithm have not been explored, although they are widely used to leverage simulation challenges arising from MDP solution algorithms. For instance, the Monte Carlo tree search [Browne et al., 2012] and multi-stage adaptive sampling algorithm [Chang et al., 2005] used bandit-based strategies to guide the growth of the search tree toward optimal actions. Moreover, some population-based direct policy search algorithms exploit a statistical racing procedure to improve the efficiency of identifying elite actions, e.g., see Heidrich-Meisner and Igel [2009] and Busa-Fekete et al. [2014]. Statistical racing was originally designed to address a model selection problem [Maron and Moore, 1997]. All candidate models are considered as competitors in a race, and their performances are quantified with Hoeffding inequality-based confidence intervals. During the simulation, certain candidate models performing worse than others are eliminated from the race. The race continues until the winner has been found. Given that Hoeffding inequality-based confidence intervals are relatively conservative, Audibert et al. [2009] and Szepesvári [2010] suggested the empirical Bernstein inequality for inferring a tighter confidence interval in the statistical racing procedure. In this study, we exploit CRN and the empirical Bernstein inequality in statistical racing to determine the best action. In particular, the proposed method is different from the canonical statistical race considering the confidence interval for the performance of a potential action, whereas the pairwise differences of all action pairs are used to eliminate the inferior actions in our approach.

4.3 Problem Statement

In this section, we first introduce the necessary notations, assumptions and system dynamics, and then move on to the dynamic programming formulation of the CTSRA problem.

4.3.1 Notation and Problem Description

Table 4.1: Description of notation.

Sets/indicies	
$\mathcal{T}$	time periods, indexed by $t \in \mathcal{T} = \{1, \ldots, T\}$
$\mathcal{I}$	drug projects, indexed by $i \in \mathcal{I} = \{1, \ldots, I\}$
$\mathcal{J}$	development stages, indexed by $j \in \mathcal{J} = \{1 \text{ (Phase I)}, 2 \text{ (Phase II)}, 3 \text{ (Phase III)}\}$
$\mathcal{K}$	resource types, indexed by $k \in \mathcal{K} = \{1, \ldots, K\}$
$\mathcal{L}$	project acceleration measures, indexed by $\ell \in \mathcal{L} = \{1, \ldots, L\}$
Uncertainties	
$\xi_{i,j} \in \{0, 1\}$	1 if task (i, j) is successfully completed
$p_{i,j}$	success probability of task (i, j)
Model Parameters	
$h_{i,j,k}$	amount of resources of type k to conduct task (i, j)
$\hat{h}^{\ell}_{i,j,k}$	amount of resources of type k to use acceleration l for task (i, j)
$\varrho_{i,j}$	time required to complete task (i, j) without using any acceleration
$\hat{\varrho}^{\ell}_{i,j}$	processing time reduction if measure l is used for one time period for task (i, j)
$c_{i,j}$	cost of conducting task (i, j)
$\hat{c}^{\ell}_{i,j}$	cost of acceleration measure l is used for task (i, j)
Γ_i	maximum revenue if drug project i gets the market approval
γ^p_i	loss due to the shortened patent life for experimental drug i
γ^d_i	loss due to the reduced market shares for experimental drug i
Actions	
$x_{i,j,t} \in \{0, 1\}$	1 if task (i, j) starts at time t
$y^{\ell}_{i,j,t} \in \{0, 1\}$	1 if acceleration measure ℓ is used for task (i, j) at time t
States/auxiliary variables	
$r_{k,t}$	amount of resources of type k that is available at time t
$\phi_{i,j,t}$	remaining time to complete the task (i, j) at time t
$d_{i,j,t} \in \{0, 1\}$	1 if task (i, j) is ready to be performed at time t
$e_{i,j,t} \in \{0, 1\}$	1 if task (i, j) is processing at time t
$\mathbb{1}_{i,j} \in \{0, 1\}$	1 if task (i, j) is completed at the current time period

Table 4.1 summarises the notation used for the formulation of the CTSRA problem. We assume that a firm has a set of experimental drugs (indexed as $i \in \mathcal{I}$) that must be tested in clinical trials over a finite planning horizon of T periods. In our model, T denotes the drug patent expiration time (usually 20 years). The

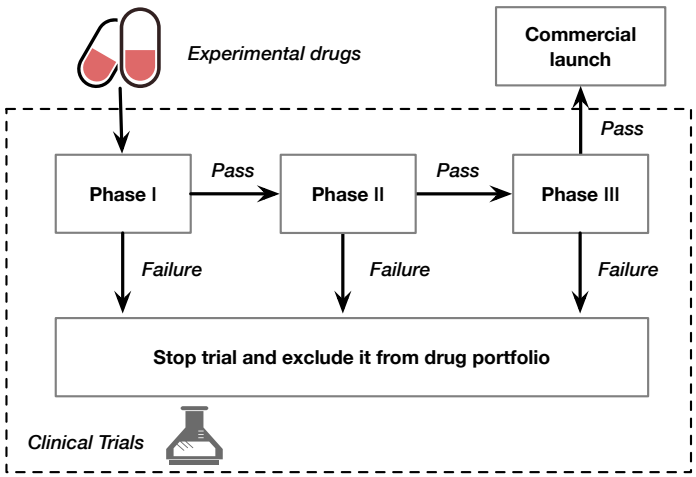

Figure 4.1: An illustration of the clinical trials process.

drug development process mainly involves three stages, namely, Phases I, II, and III clinical tests that are denoted as $j = 1, 2$ and 3 respectively. A task (i, j) refers to the clinical trial j of experimental drug i. For task (i, j), the firm is expected to take $\varrho_{i,j}$ time periods to prove the safety and efficacy of experimental drug if no acceleration measure is used. The possible outcome of task (i, j) can be either a success or a failure. If an experimental drug fails at any phase of its trial, then it will be removed from the drug portfolio, and testing the remaining phases will be abandoned. However, as long as the experimental drugs pass Phase III, the firm can submit the new drug application to regulatory agencies for market approval. The entire process of drug development is illustrated in Figure 4.1.

A set of indivisible and reusable resources, indexed as $k \in \mathcal{K}$, is initially given, which might be the research staff, experimental devices, and laboratories. The resources are shared among development tasks, each of which has specific requirements $h_{i,j,k}$ for resource type $k \in \mathcal{K}$. The development cost $c_{i,j}$ needs to be paid when task (i, j) is conducted. Moreover, several acceleration measures (indexed with $\ell \in \mathcal{L}$) can be utilised with additional cost $\hat{c}^{\ell}_{i,j}$ to compress the cycle time for task (i, j). These acceleration measures can be increasing staff number, operating additional test sites, increasing advertising expenditures, or offering high-value monetary incentives. We assume that the processing time will decrease a certain amount of time $\hat{\varrho}^{\ell}_{i,j}$ if acceleration measure ℓ is used for task (i, j).

At each period, we can start processing available tasks of which the predecessor tasks have been successfully completed and use acceleration measures for the tasks in progress. Let $x_{i,j,t}$ be a binary variable representing whether task (i, j) is conducted at time t, and $y^{\ell}_{i,j,t}$ denote a binary variable that represents whether

acceleration measure l is used for task (i,j) at time t. Note that the scheduling action $x_{i,j,t}$ is irrevocable and remains in force until a clinical test is completed; and the action $y^{\ell}_{i,j,t}$ lasts for a single period. For the sake of readability, we write the aforementioned action variables at time t in a two-tuple $a_t = (X_t, Y_t)$ where X_t and Y_t respectively denote the sets of $x_{i,j,t}$ and $y^{\ell}_{i,j,t}$ for $i \in \mathcal{I}, j \in \mathcal{J}$ and $\ell \in \mathcal{L}$.

States of the system comprise the status of resources as well as tasks. Let $r_{k,t}$ denote the amount of type k resource that is available at time t, and $d_{i,j,t}$ be a binary variable that takes 0 if task (i,j) is not available at time t, and 1 otherwise. In particular, $d_{i,1,1} = 1, i \in \mathcal{I}$ represents that the Phase I clinical test of all drug projects is ready to be performed at the beginning of the planning horizon $t = 1$. Moreover, we consider two auxiliary variables in order to describe the system dynamics. Let binary variable $e_{i,j,t}$ represent whether task (i,j) is in progress at time t and $\phi_{i,j,t}$ denote the remaining time to complete task (i,j) at time t. At the initial stage $t = 1$, we have $\phi_{i,j,1} = \varrho_{i,j}$ for $i \in \mathcal{I}$ and $j \in \mathcal{J}$. We define a state $\mathcal{S}_t$ of the system as four-tuple:

$$\mathcal{S}_t = (\mathcal{D}_t, \mathcal{E}_t, \mathcal{R}_t, \Phi_t)$$

where $\mathcal{D}_t = \{d_{i,j,t} |\ \ i \in \mathcal{I}, j \in \mathcal{J}\}$, $\mathcal{E}_t = \{e_{i,j,t} |\ \ i \in \mathcal{I}, j \in \mathcal{J}\}$, $\mathcal{R}_t = \{r_{k,t} |\ \ k \in \mathcal{K}\}$ and $\Phi_t = \{\phi_{i,j,t} |\ \ i \in \mathcal{I}, j \in \mathcal{J}\}$. Moreover, ξ is a vector involving all possible trial outcomes, where $\xi_{i,j}$ is a binary variable that takes 1 if task (i,j) is successful and 0, otherwise. We assume that the outcome $\xi_{i,j}$ of task (i,j) follows a Bernoulli distribution with success probability $p_{i,j}$. The state $\mathcal{S}_t$ can be recursively updated as follows:

$$f(S_t, a_t, \xi) \doteq \Big\{ (\mathcal{D}_{t+1}, \mathcal{E}_{t+1}, \mathcal{R}_{t+1}, \Phi_{t+1}) |\ \ d_{i,1,t+1} = d_{i,1,t} - x_{i,1,t}, \quad \forall i \in \mathcal{I}, \tag{4.1}$$

$$d_{i,j,t+1} = \xi_{i,j-1} \times \mathbb{1}_{i,j-1} + (d_{i,j,t} - x_{i,j,t}) \times (1 - \mathbb{1}_{i,j-1}), \quad \forall i \in \mathcal{I}, j \in \{2,3\}, \tag{4.2}$$

$$e_{i,j,t+1} = e_{i,j,t} + x_{i,j,t} - \mathbb{1}_{i,j}, \quad \forall i \in \mathcal{I}, j \in \mathcal{J}, \tag{4.3}$$

$$r_{k,t+1} = r_{k,t} + \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{J}} (\mathbb{1}_{i,j} - x_{i,j,t}) \times h_{i,j,k}, \quad \forall k \in \mathcal{K}, \tag{4.4}$$

$$\phi_{i,j,t+1} = \max\{0, \phi_{i,j,t} - (x_{i,j,t} + e_{i,j,t} + \sum_{\ell \in \mathcal{L}} y^{\ell}_{i,j,t} \times \hat{\varrho}^{\ell}_{i,j})\}, \quad \forall i \in \mathcal{I}, j \in \mathcal{J} \Big\}, \tag{4.5}$$

where $\mathbb{1}_{i,j}$ is an indicator function that takes a value of 1 if $\phi_{i,j,t} > 0$ and $\phi_{i,j,t+1} \leq 0$. Equations (4.1) and (4.2) express that the successive tasks at time $t+1$ become performable only when the predecessor tasks have been successfully completed. Equation (4.3) implies that the state of a clinical trial will be marked as "in progress" at

the next period if the trial has started, but not completed yet. The number of available resources is then updated as in Equation (4.4). At the next period, resources of the completed and on-going tasks will be recovered and deducted, respectively. The auxiliary variables Φ_t are updated as in Equation (4.5).

4.3.2 Dynamic Programming Formulation

On the basis of state variables, the feasible action space $\mathcal{A}_t$ at time t can be defined as

$$\mathcal{A}_t \doteq \Big\{(X_t, Y_t)| \;\; x_{i,j,t} \leq d_{i,j,t}, \;\; \forall i \in \mathcal{I}, j \in \mathcal{J}, \tag{4.6}$$

$$y^{\ell}_{i,j,t} \leq x_{i,j,t} + e_{i,j,t}, \;\; \forall i \in \mathcal{I}, j \in \mathcal{J}, \ell \in \mathcal{L}, \tag{4.7}$$

$$\sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{J}} (x_{i,j,t} \times h_{i,j,k} + \sum_{\ell \in \mathcal{L}} y^{\ell}_{i,j,t} \times \hat{h}^{\ell}_{i,j,k}) \leq r_{k,t}, \;\; \forall k \in \mathcal{K} \Big\}. \tag{4.8}$$

Constraint (4.6) refers to the possible action $x_{i,j,t}$ for task (i, j) at time t to be selected from the performable tasks. Constraint (4.7) expresses that acceleration measures can be only implemented for any ongoing clinical trials. The regulation measure in which all possible resource allocation plans do not exceed the capacity is presented in Constraint (4.8).

Let $Rv_{i,t}$ be the projected revenue to be received from drug project i if it already passed the Phase III clinical trial. As suggested by Colvin and Maravelias [2008], the projected revenue $Rv_{i,t}$ needs to be discounted due to the shortened patent life and reduced market shares. Let γ^p_i and γ^d_i respectively denote the loss per period resulting from shortened patent life and reduced market shares for drug project i. The projected revenue $Rv_{i,t}$ is defined as

$$Rv_{i,t} = \Gamma_i - \gamma^p_i \times t - \gamma^d_i \times \max\{0, t - \sum_{j \in \mathcal{J}} \varrho_{i,j}\} \tag{4.9}$$

where Γ_i denotes the maximum projected revenue when drug i is approved. Note that the penalty γ^p_i immediately occurs at the beginning of planning horizon, while the penalty γ^d_i only happens if the actual completion time is greater than the earliest completion time. Early completions can lead to comparatively high projected revenues, and also propel the completion of the drug development process to the earliest time possible. Given a state $\mathcal{S}_t$, the reward function can be defined in terms

of projected revenues and expenses after an action $a_t \in \mathcal{A}_t$ has been undertaken:

$$R\left(\mathcal{S}_t, a_t, \xi\right) \doteq \text{total projected revenues - development costs}$$
$$= \sum_{i \in \mathcal{I}} \left[Rv_{i,t} \times \mathbb{1}_{i,3} \times \xi_{i,3} - \sum_{j \in \mathcal{J}} (c_{i,j} \times x_{i,j,t} + \sum_{\ell \in \mathcal{L}} y^{\ell}_{i,j,t} \times \hat{c}^{\ell}_{i,j}) \right]. \tag{4.10}$$

The optimal action at state $\mathcal{S}_t$ is obtained by solving the following Bellman's equation:

$$V_t(S_t) = \max_{a_t \in \mathcal{A}_t} \mathbb{E}\left\{ R(\mathcal{S}_t, a_t, \xi) + V_{t+1}(f(\mathcal{S}_t, a_t, \xi)) \right\} \tag{4.11}$$

where $V_t(\mathcal{S}_t)$ denotes the expected cumulative reward from state $\mathcal{S}_t$ until the terminal state to be satisfied.

Bellman's equation (4.11) is difficult to solve using analytical solution methods because the reward-to-go function does not have a compact representation. The computational difficulty of finding an optimal policy grows exponentially as the number of possible state increases. To illustrate such challenge, let us consider an example of a no-possible delay case where we have 5 drug projects. There are several possible ways of assigning resources to each clinical trial. If there exists 10 possible processing times for each task given various resource allocation strategies, then we have up to $(4^{10})^5$ states. If the simplification assumption is abandoned, then the number of possible states obviously increases further.

4.4 The Rollout Algorithm

The rollout algorithm is a look-ahead search method to approximately solve Bellman's equation (4.11). Let us consider the rollout algorithm applied on the state $\mathcal{S}_t$. The search process relies on a decision tree whose structure is illustrated in Figure 4.2. The top layer of the decision tree is *root node* $\mathcal{S}_t$, which can be specified as a non-terminal state. At root node $\mathcal{S}_t$, several candidate actions may satisfy resource constraints. After a candidate action $a_t \in \mathcal{A}_t$ has been performed, the system state is represented as a *child node* $\mathcal{S}_{t+1} = f(\mathcal{S}_t, a_t, W)$ at the next period. Afterwards, the rollout algorithm utilises a base policy to guide movement at the child node. The base policy can be comprised of either a heuristic rule, a mathematical program or a search method.

Consider that child node $\mathcal{S}_{t+1}$ has a set of drug projects $\tilde{\mathcal{I}} \subseteq \mathcal{I}$ that have not failed in any clinical tests in the past. The awaiting tests of drug $i \in \tilde{\mathcal{I}}$ are denoted as $\tilde{\mathcal{J}}_i = \{j'_i \leq j \leq 3\}$ where j'_i is the current development stage of drug project i. Let

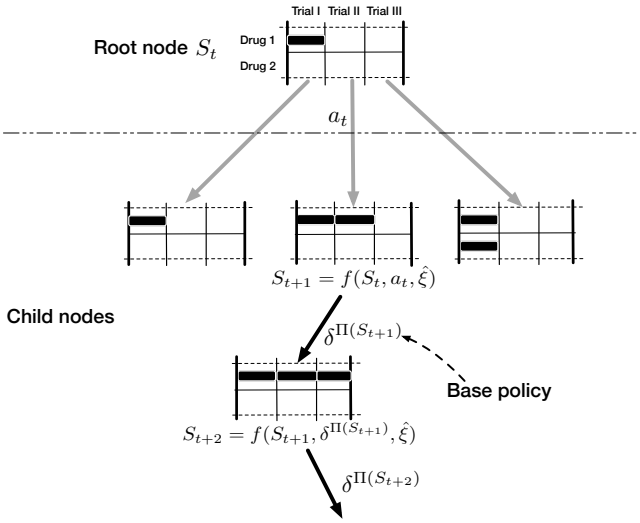

Figure 4.2: An illustration of the decision tree structure.

Π denote a base policy that provides an action $\delta^{\Pi(\mathcal{S}_\tau)}$ at child node $\mathcal{S}_\tau$ for $\tau \geq t+1$. By using the base policy Π, one possible trajectory after taking action $a_t \in \mathcal{A}_t$ at root node $\mathcal{S}_t$ can be written as follows:

$$\left\{(\mathcal{S}_t, \mathcal{S}_{t+1}, \ldots, \mathcal{S}_T) \middle| \ \mathcal{S}_{t+1} \in f(\mathcal{S}_t, a_t, \hat{\xi}), \ldots, \mathcal{S}_T \in f(\mathcal{S}_{T-1}, \delta^{\Pi(\mathcal{S}_{T-1})}, \hat{\xi})\right\}. \quad (4.12)$$

where $\hat{\xi} \equiv (\hat{\xi}_{i,j})_{i \in \tilde{\mathcal{I}}, j \in \tilde{\mathcal{J}}_i}$ is a particular sample of random task outcomes ξ. We can employ the Markovian property to compute the cumulative reward $\mathcal{Z}^{\Pi}(\mathcal{S}_t, a_t, \hat{\xi})$ that is received on the abovementioned trajectory:

$$\mathcal{Z}^{\Pi}(\mathcal{S}_t, a_t, \hat{\xi}) = R(\mathcal{S}_t, a_t, \hat{\xi}) + \sum_{\tau=t+1}^{T} R(\mathcal{S}_\tau, \delta^{\Pi(\mathcal{S}_\tau)}, \hat{\xi}). \quad (4.13)$$

By generating M possible trajectories $\{\hat{\xi}^1, \hat{\xi}^2, \ldots, \hat{\xi}^M\}$, the reward-to-go function $V(\mathcal{S}_t)$ in (4.11) can be approximated by

$$\begin{aligned} V_t(\mathcal{S}_t) &= \max_{a_t \in \mathcal{A}_t} \mathbb{E}\left\{ R(\mathcal{S}_t, a_t, \hat{\xi}) + V_{t+1}(f(\mathcal{S}_t, a_t, \hat{\xi})) \right\} \\ &\approx \max_{a_t \in \mathcal{A}_t} \frac{1}{M} \sum_{m=1}^{M} \left\{ R(\mathcal{S}_t, a_t, \hat{\xi}^m) + \sum_{\tau=t+1}^{T} R(\mathcal{S}_\tau, \delta^{\Pi(\mathcal{S}_\tau)}, \hat{\xi}^m) \right\} \\ &\approx \max_{a_t \in \mathcal{A}_t} \frac{1}{M} \sum_{m=1}^{M} \left\{ \mathcal{Z}^{\Pi}(\mathcal{S}_t, a_t, \hat{\xi}^m) \right\}. \end{aligned} \quad (4.14)$$

We can obtain a rollout decision rule at the current root node by solving the approximate reward-to-go function. The running time complexity of the rollout algorithm is $\mathcal{O}(M \times |\mathcal{A}_t|)$ where $|\mathcal{A}_t|$ denotes the size of action feasibility set at the root node $\mathcal{S}_t$. If the rollout algorithm is applied to each possible state of the system, then we obtain a so-called rollout policy Π^R to nearly solve the CTSRA problem.

4.5 The Optimistic Policy

In this section, we present a base policy called "optimistic policy" to guide the moves of the rollout algorithm at the child nodes. The optimistic policy (denoted as Π^O) is based on the rolling horizon optimisation framework (see Sethi and Sorger [1991] and Chand et al. [2002]), which replaces the uncertain outcome of drug projects with their sample-based approximations in a mathematical programming model and periodically updates the model formulation as the system state evolves. Moreover, we adopt the following assumption for reducing the computational difficulty of obtaining an optimistic policy. Suppose that the search process remains at the child node $\mathcal{S}_{t+1}$. We ignore the resource capacity constraints at the lower-layer nodes $\mathcal{S}_\tau$ for $\tau > t+1$, whilst preserving the capacity constraint for the child node $\mathcal{S}_{t+1}$. In such a way, we obtain an upper bound estimation for the expected cumulative reward after taking a particular action at the child node $\mathcal{S}_{t+1}$ because the delay due to resource shortage for any drug project is not taken into account. Subsequently, Proposition 4.1 describes the formulation of the optimistic policy based on the rolling horizon framework.

Proposition 4.1. For task (i, j), where $i \in \tilde{\mathcal{I}}$ and $j \in \tilde{\mathcal{J}}_i$, let $g_{i,j}$ describe the corresponding time periods to complete this task, and $\hat{g}^l_{i,j}$ denotes time periods when the acceleration measure l is used for this task. In addition, let $\tilde{\mathcal{R}}_i$ be the expected profit of drug project i for $i \in \tilde{\mathcal{I}}$ if it passes the Phase III clinical test. The corresponding development cost associated with drug project i is denoted by $\tilde{\mathcal{C}}_{i,j}$ for $j \in \tilde{\mathcal{J}}_i$. The optimistic policy produces an action $\delta^{\Pi(S_{t+1})} = \{(x_{i,j'_i,t+1}, y^\ell_{i,j'_i,t+1}) | i \in \tilde{\mathcal{I}}, \ell \in \mathcal{L}\}$ for

child node $\mathcal{S}_{t+1}$ by solving the following linear mixed-integer programming model:

$$\max_{\substack{g_{i,j},\hat{g}^{\ell}_{i,j},\tilde{\mathcal{C}}_{i,j}\\ x_{i,j'_i,t+1},y^{\ell}_{i,j'_i,t+1}}} \sum_{i\in\tilde{\mathcal{I}}} \tilde{\mathcal{R}}_i = \begin{cases} \left[Rv_{i,t+1+\sum_{j\in\tilde{\mathcal{J}}_i} g_{i,j}} - \sum\limits_{j\in\tilde{\mathcal{J}}_i} \tilde{\mathcal{C}}_{i,j} \right] \times \prod\limits_{j\in\tilde{\mathcal{J}}_i} p_{i,j} - \\ \sum\limits_{j\in\tilde{\mathcal{J}}_i} \tilde{\mathcal{C}}_{i,j} \times (1-p_{i,3}) \times \prod\limits_{j\in\{1,2\}} p_{i,j} - \\ \sum\limits_{j\in\{1,2\}} \tilde{\mathcal{C}}_{i,j} \times (1-p_{i,2}) \times p_{i,j'} - \tilde{\mathcal{C}}_{i,1} \times (1-p_{i,1}), & \text{if } j'_i = 1 \\ \left[Rv_{i,t+1+\sum_{j\in\tilde{\mathcal{J}}_i} g_{i,j}} - \sum\limits_{j\in\tilde{\mathcal{J}}_i} \tilde{\mathcal{C}}_{i,j} \right] \times \prod\limits_{j\in\tilde{\mathcal{J}}_i} p_{i,j} - \\ \sum\limits_{j\in\tilde{\mathcal{J}}_i} \tilde{\mathcal{C}}_{i,j} \times (1-p_{i,3}) \times p_{i,2} - \tilde{\mathcal{C}}_{i,2} \times (1-p_{i,2}), & \text{if } j'_i = 2 \\ \left[Rv_{i,t+1+g_{i,3}} - \tilde{\mathcal{C}}_{i,3} \right] \times p_{i,3} - \tilde{\mathcal{C}}_{i,3} \times (1-p_{i,3}), & \text{if } j'_i = 3 \end{cases} \tag{4.15}$$

s.t. (Development Times)

$$g_{i,j'_i} \geq \phi_{i,j'_i,t+1} - x_{i,j'_i,t+1} - \sum_{\ell\in\mathcal{L}} \hat{\varrho}^{\ell}_{i,j'_i} \times (y^{\ell}_{i,j'_i,t+1} + \hat{g}^{\ell}_{i,j'_i}), \quad \forall i \in \tilde{\mathcal{I}} \tag{4.16}$$

$$g_{i,j} \geq \varrho_{i,j} - \sum_{\ell\in\mathcal{L}} \hat{\varrho}^{\ell}_{i,j'} \times \hat{g}^{\ell}_{i,j}, \quad \forall i \in \tilde{\mathcal{I}}, \quad j \in \tilde{\mathcal{J}}_i \backslash \{j'_i\} \tag{4.17}$$

(Development Costs)

$$\tilde{\mathcal{C}}_{i,j'_i} \geq c_{i,j'_i} \times d_{i,j'_i,t+1} + \sum_{\ell\in\mathcal{L}} \hat{c}^{\ell}_{i,j'_i} \times (\hat{g}^{\ell}_{i,j'_i} + y^{\ell}_{i,j'_i,t+1}), \quad \forall i \in \tilde{\mathcal{I}} \tag{4.18}$$

$$\tilde{\mathcal{C}}_{i,j} \geq c_{i,j} + \sum_{\ell\in\mathcal{L}} \hat{c}^{\ell}_{i,j} \times \hat{g}^{\ell}_{i,j}, \quad \forall i \in \tilde{\mathcal{I}}, \quad j \in \tilde{\mathcal{J}}_i \backslash \{j'_i\} \tag{4.19}$$

$$\hat{g}^{\ell}_{i,j} \in \mathbb{N}, \quad \forall i \in \tilde{\mathcal{I}}, \quad j \in \tilde{\mathcal{J}}_i, \quad \ell \in \mathcal{L}; \quad g_{i,j} \in \mathbb{N}, \tilde{\mathcal{C}}_{i,j} \in \mathbb{R}, \quad \forall i \in \tilde{\mathcal{I}}, \quad j \in \tilde{\mathcal{J}}_i$$

$$(x_{i,j'_i,t+1}, y^{\ell}_{i,j'_i,t+1}) \in \mathcal{A}_{t+1}, \ \forall i \in \tilde{\mathcal{I}}, \quad \ell \in \mathcal{L}$$

Proof. Consider that optimistic decision rule is applied to the child node S_{t+1}. Our goal is to identify the actions $x_{i,j'_i,t+1}$ and $y_{i,j'_i,t+1}$ should be taken at this node. Following the notations used in Section 4.2, we first start deriving the objective function. For drug project $i \in \tilde{\mathcal{I}}$ at time $t+1$, the completion time of Phase III clinical test can be stated as

$$t + 1 + \sum_{j\in\tilde{\mathcal{J}}_i} g_{i,j}.$$

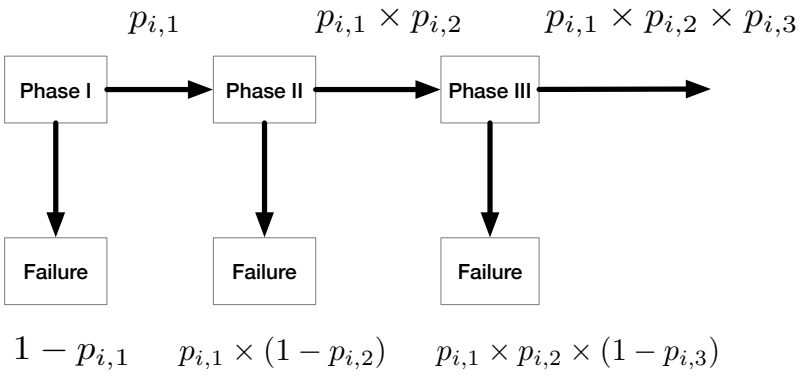

Figure 4.3: Probability of observing a particular trajectory for the single drug product given decision rules at the lower layer child nodes.

The actual duration $g_{i,j}$ not only depends on development time $\varrho_{i,j}$, but also on how many time periods the acceleration measure $\hat{\varrho}^{\ell}_{i,j'}$ is applied in the planning process. Note that, for task (i, j'_i), the development time is not only affected by actions $x_{i,j'_i,t+1}$ and $y^{l}_{i,j'_i,t+1}$ at child node S_{t+1} but also the time periods of using acceleration measure $\hat{g}^{l}_{i,j'_i}$ at lower-layer nodes S_τ for $\tau > t+1$. Constraints (4.16) and (4.17) describe the calculation of actual duration $g_{i,j}$ in the optimistic policy.

As shown in (4.15), the optimistic policy aims to maximise the expected profit over all possible task outcomes. The probability of various development outcomes is shown in Figure 4.3. When this drug project remains at the second or third clinical test, we can replace $p_{i,1}$ and/or $p_{i,2}$ by 1. Only in case of passing all clinical tests, firm will receive revenue $Rv_{i,t+1+\sum_{j\in\tilde{\mathcal{J}}_i} g_{i,j}}$ of which calculation process can be found in (4.9). The development cost depends on clinical tests and acceleration measures. For task (i, j'_i), the development cost $\tilde{\mathcal{C}}_{i,j'_i}$ can be calculated as in the right hand side of Constraint (4.18). The first term implies the cost of performing task (i, j'_i) if it is ready to start. The second term is related to the total cost of using acceleration measures. On the other hand, the development cost $\tilde{\mathcal{C}}_{i,j}$ for $j \in \tilde{\mathcal{J}}_i \backslash \{j'_i\}$ that is not influenced by actions at node S_{t+1} can be computed as in the right hand side of Constraint (4.19). □

The complexity of solving the linear mixed-integer programming model depends on the current development stage of each project. At the worst-case scenario, all drug projects remain at the Phase I trial. In this case, the above model has $I \times (J \times L + 2 \times J + 1 + L)$ decision variables and $I \times (1 + L + 2 \times J + J \times L) + K$ linear constraints.

Next, we introduce the definition of a sequentially consistent policy, which

plays the key role in establishing rollout improvement property.

Definition 4.1: Sequentially Consistent Policy [Bertsekas, 2005]. A base policy is sequentially consistent if this policy produces a partial trajectory

$$\left\{\mathcal{S}_t, \mathcal{S}_{t+1}, \mathcal{S}_{t+2}, \ldots, \mathcal{S}_T\right\}$$

from state $\mathcal{S}_t$, it also generates the same subsequent states

$$\left\{\mathcal{S}_{t+1}, \mathcal{S}_{t+2}, \ldots, \mathcal{S}_T\right\}$$

starting from state $\mathcal{S}_{t+1}$.

Next, we investigate the performance improvement property when the optimistic policy is used with the rollout algorithm.

Proposition 4.2. Suppose that the rollout policy Π^R is constructed by using the rollout algorithm with the optimistic policy Π^O. The following inequality holds, for any non-terminal states,

$$V^{\Pi^R}(\mathcal{S}_t) \geq V^{\Pi^O}(\mathcal{S}_t), \quad \forall t \in \mathcal{T}. \tag{4.20}$$

Proof. We prove that the inequality (4.20) holds by induction. Assume that $\mathcal{Z}_t^{\Pi^O}(\mathcal{S}_t, a_t, \hat{\xi}^m)$ is one possible trajectory generated by using the optimistic policy Π^O. The optimistic policy is computed by solving a rolling horizon optimisation model. The model parameters are only related to state variables. Therefore, it is sufficient to say that the optimistic policy satisfies the property of sequential consistency. As shown in (4.14), the reward-to-go of the rollout policy Π^R at the root node $\mathcal{S}_t$ can be computed as follows:

$$V^{\Pi^R}(\mathcal{S}_t) =$$
$$\max_{a_t \in \mathcal{A}_t} \frac{1}{M} \sum_{m=1}^{M} \left\{ R(\mathcal{S}_t, a_t, \hat{\xi}^m) + \sum_{\tau=t+1}^{T} R(\mathcal{S}_\tau, \delta^{\Pi(\mathcal{S}_\tau)}, \hat{\xi}^m) | \mathcal{S}_\tau \in f(\mathcal{S}_{\tau-1}, \delta^{\Pi(\mathcal{S}_{\tau-1})}, \hat{\xi}^m) \right\}$$

The actions produced by using the optimistic policy is one element of the feasible action set on the root node $\mathcal{S}_t$. Because the trajectories constructed by using the

optimistic policy are sequentially consistent, the following inequality holds,

$$
\begin{aligned}
&V^{\Pi^R}(\mathcal{S}_t) \\
&\geq \frac{1}{M}\sum_{m=1}^{M}\left\{R(\mathcal{S}_t,\delta^{\Pi(\mathcal{S}_t)},\hat{\xi}^m)+\sum_{\tau=t+1}^{T}R(\mathcal{S}_\tau,\delta^{\Pi(\mathcal{S}_\tau)},\hat{\xi}^m)|\mathcal{S}_\tau\in f(\mathcal{S}_{\tau-1},\delta^{\Pi(\mathcal{S}_{\tau-1})},\hat{\xi}^m)\right\} \\
&= V^{\Pi}(\mathcal{S}_t).\square
\end{aligned}
$$

Proposition 4.2 shows that the reward-to-go function computed by using the optimistic policy Π^O and averaging all possible cumulative rewards received from state $\mathcal{S}_t$ to any terminal state is as good as or better than that of purely using optimistic policy Π^O.

To obtain the rollout policy Π^R, we need to identify the best action at each state by repeatedly using the optimistic policy for all possible actions several times to estimate and compare their expected performances over various task outcomes. Finding a good action used for the root node remains challenging although the computational effort is significantly reduced by adopting the resource constraint relaxation. This motivates the use of adaptive sampling strategies in the rollout algorithm to reduce the required number of evaluations.

4.6 Adaptive Sampling Strategies

In this section, we first describe the benefit of using CRN in the simulation procedure and then introduce two adaptive sampling strategies that employ the variance reduction technique of CRN and the empirical Bernstein inequalities in a statistical racing procedure.

4.6.1 Variance Reduction using CRN

Suppose that performances of two candidate actions $a_{1,t}$ and $a_{2,t}$ need to be differentiated at the root node $\mathcal{S}_t$. Let $\Xi^1=\{\tilde{\xi}^{1,1},\tilde{\xi}^{1,2},\ldots,\tilde{\xi}^{1,M}\}$ and $\Xi^2=\{\tilde{\xi}^{2,1},\tilde{\xi}^{2,2},\ldots,\tilde{\xi}^{2,M}\}$ denote the samples of uncertain task outcomes that have been explored for actions $a_{1,t}$ and $a_{2,t}$, respectively. To compare their performances, we can estimate the sample mean $\hat{\mu}_{1,2}$ as performance difference between $a_{1,t}$ and $a_{2,t}$:

$$
\hat{\mu}_{1,2}=\frac{1}{M}\sum_{m=1}^{M}\left[\mathcal{Z}^{\Pi^O}(\mathcal{S}_t,a_{1,t},\tilde{\xi}^{1,m})-\mathcal{Z}^{\Pi^O}(\mathcal{S}_t,a_{2,t},\tilde{\xi}^{2,m})\right]. \tag{4.21}
$$

Let $\mathbb{V}ar(a_{1,t})$ and $\mathbb{V}ar(a_{2,t})$ denote variances of the cumulative rewards of actions $a_{1,t}$ and $a_{2,t}$, respectively. The covariance between two groups of cumulative rewards is denoted as $\mathbb{C}ov(a_{1,t}, a_{2,t})$. Sample variance of estimator $\hat{\mu}_{1,2}$ can be computed as

$$\mathbb{V}ar_{1,2} = \mathbb{V}ar_1 + \mathbb{V}ar_2 - 2 \times \mathbb{C}ov_{1,2}. \tag{4.22}$$

If Ξ^1 and Ξ^2 are two independent samples, then $\mathbb{C}ov(a_{1,t}, a_{2,t})$ becomes zero. On the other hand, if CRN is used for sample generation (i.e. $\Xi^1 = \Xi^2$), then the covariance term is likely to be positive [Law et al., 1991]. And thus the variance of the differences is smaller than with independent samples, making it easier to detect significant differences.

4.6.2 New Sampling Strategy

We now present a CRN-based sampling strategy (denoted as $\mathcal{CB}$) that iteratively eliminates inferior actions from simulation by comparing the Bernstein-based confidence interval for the sample mean of differences between two distinct actions. Algorithm 4.1 describes the overall procedure of integrating $\mathcal{CB}$ within the rollout algorithm. The main steps of the $\mathcal{CB}$ strategy are explained as follows.

Consider a set of surviving candidate actions $A \subseteq \mathcal{A}_t$ in the statistical racing. For action $a_{n,t} \in A$, let Ξ denote a set of uncertain task outcomes (samples) explored in the previous simulation stages. Since $\mathcal{CB}$ applies the setting of common random number, surviving candidate actions should have explored the exactly same samples. Let $Z = \{\mathcal{Z}^{\Pi^O}(\mathcal{S}_t, a_{n,t}, \hat{\xi}) |\ \ a_{n,t} \in A, \hat{\xi} \in \Xi\}$ correspond to the expanded search tree, i.e. a set of cumulative rewards for each candidate action collected from the trajectories generated by using the optimistic policy Π^O and samples Ξ. First, the $\mathcal{CB}$ sampling strategy computes the sample mean as performance difference between two distinct candidate actions $a_{n,t}$ and $a_{n',t}$ as follows;

$$\hat{\mu}_{n,n'} = \frac{1}{|\Xi|} \sum_{\hat{\xi} \in \Xi} \left[\mathcal{Z}^{\Pi^O}(\mathcal{S}_t, a_{n,t}, \hat{\xi}) - \mathcal{Z}^{\Pi^O}(\mathcal{S}_t, a_{n',t}, \hat{\xi}) \right], \tag{4.23}$$

and the sample variance is calculated as follows,

$$\mathbb{V}ar_{n,n'} = \frac{\sum_{\hat{\xi} \in \Xi} \left[(\mathcal{Z}^{\Pi^O}(\mathcal{S}_t, a_{n,t}, \hat{\xi}) - \mathcal{Z}^{\Pi^O}(\mathcal{S}_t, a_{n',t}, \hat{\xi}) - \hat{\mu}_{n,n'})^2 \right]}{|\Xi| - 1}. \tag{4.24}$$

The $\mathcal{CB}$ strategy employs the empirical Bernstein inequality proposed by Audibert

Algorithm 4.1: An integration of $\mathcal{CB}$ into the rollout algorithm

input : root node $\mathcal{S}_t$, optimistic policy Π^O, a set of candidate actions A, significance level α, expanded search tree Z, evaluated test outcomes Ξ, indifference coefficient β.
output: action with the highest sample mean, updated search tree Z.

simulation stage:
if an action has not been evaluated yet, **then** perform an initial estimation for this action;
while $|A| > 1$ **do**
  if *simulation budget has been reached* **then**
    go to the final selection stage
  else
    for *all distinct* $a_{n,t}, a_{n',t} \in A$ **do**
      compute $\hat{\mu}_{n,n'}$ and $\mathbb{V}ar_{n,n'}$ using (4.23) and (4.24);
      calculate the upper confidence bound as in (4.25);
    end
    if $a_{n,t} \in A$ satisfies the criterion in (4.27), **then** $A \leftarrow A \backslash \{a_{n,t}\}$;
    generate a new CRN sample $\hat{\xi}$ of uncertain task outcomes;
    for $a_{n,t} \in A$ **do**
      compute the cumulative reward $\mathcal{Z}_t^{\Pi^O}(\mathcal{S}_t, a_{n,t}, \hat{\xi})$ using (4.13);
      update the search tree $Z \leftarrow Z \cup \mathcal{Z}_t^{\Pi^O}(\mathcal{S}_t, a_{n,t}, \hat{\xi})$;
    end
  end
end

et al. [2009] to construct a confidence interval for the sample mean $\hat{\mu}_{n,n'}$ as

$$\left[\hat{\mu}_{n,n'} - CI^{\alpha}_{n,n'} \ , \ \hat{\mu}_{n,n'} + CI^{\alpha}_{n,n'}\right] \tag{4.25}$$

where

$$CI^{\alpha}_{n,n'} = \sqrt{\frac{2 \times \mathbb{V}ar_{n,n'} \times \log(3/\alpha)}{|\Xi|}} + \frac{3 \times \Theta_{n,n'} \times \log(3/\alpha)}{|\Xi|}.$$

Here $\alpha \geq 0$ denotes the significance level and $\Theta_{n,n'}$ is the greatest performance gap between actions $a_{n,t}$ and $a_{n',t}$. Let $Z^{max}(a_{n,t})$ and $Z^{min}(a_{n,t})$ denote the largest and smallest cumulative rewards of action $a_{n,t}$, respectively. The value of $\Theta_{n,n'}$ can be calculated by

$$\Theta_{n,n'} = \max\left\{\left|Z^{max}(a_{n,t}) - Z^{min}(a_{n',t})\right|, \left|Z^{max}(a_{n',t}) - Z^{min}(a_{n,t})\right|\right\}. \tag{4.26}$$

Before conducting simulation, the largest and smallest cumulative rewards of an action are unknown, but can be roughly estimated by using a small batch of simulation results.

Next, the sampling algorithm identifies whether any candidate action should be removed from the next simulation stage. The inferior actions do not help to improve our knowledge about potentially promising actions and need to be eliminated from the simulation procedure to reduce the computational effort. The candidate action $a_{n,t} \in A$ will be eliminated if the related upper confidence bounds satisfy the following elimination criterion. Given an indifference coefficient $\beta \geq 0$, if

$$\hat{\mu}_{n,n'} + CI^{\alpha}_{n,n'} \leq \beta, \quad \forall a_{n',t} \in A \backslash \{a_{n,t}\}, \tag{4.27}$$

holds, then the sampling strategy updates the candidate set A by eliminating action $a_{n,t}$. If this is not the case, candidate action $a_{n,t}$ will be reserved in the statistical racing procedure. At the end of the current simulation stage, we generate a new sample of uncertain task outcomes, compute the cumulative rewards for all remaining actions, and update the expanded search tree Z.

The sampling algorithm iteratively performs the simulation and elimination procedure until the stoping rule is satisfied. $\mathcal{CB}$ utilises the number of surviving candidates to define the stoping rule as follows

$$|A| \leq 1.$$

Once the search is interrupted, a candidate action must be chosen for use in the current root node. Let W^n denote the samples of uncertain task outcomes explored for candidate a_n. In the last simulation stage, we select the candidate action with largest expected cumulative reward.

We illustrate the elimination mechanism of the $\mathcal{CB}$ sampling strategy in Figure 4.4. Assume that the root node $\mathcal{S}_t$ has three candidate actions $a_{1,t}$, $a_{2,t}$ and $a_{3,t}$, and the indifference coefficient β is fixed at 0.1. We now compute the sample mean of differences between any two distinct actions by using information of the expanded search tree. The upper confidence bounds of sample means are shown in Figure 4.4. The results show that $a_{1,t}$ and $a_{2,t}$ are competing actions at the current simulation stage. On the other hand, $a_{3,t}$ is worse than the other two actions. The upper confidence bounds for $\mu_{3,1}$ and $\mu_{3,2}$ are -0.4 and 0.05, respectively. Both upper bounds are smaller than the indifference coefficient β. We can stop further sampling for action $a_{3,t}$.

4.6.3 Augmented Sampling Strategy

Now, we introduce another sampling method to improve the efficiency of finding the best candidate in the case of numerous candidate actions at relatively small

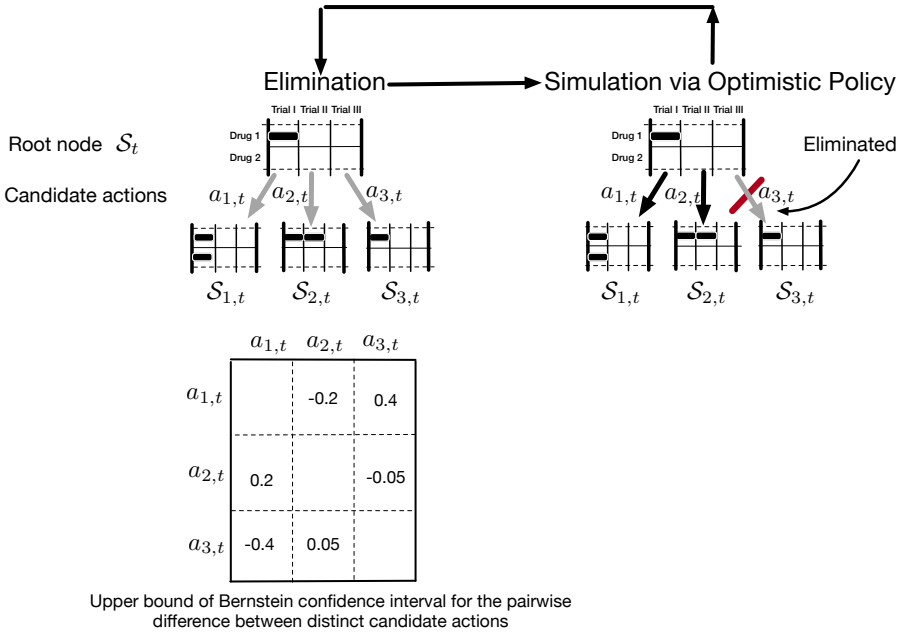

Figure 4.4: The use of the $\mathcal{CB}$ sampling strategy.

simulation budget. This method uses a heuristic-based grouping rule to adjust the exploration priority in the simulation process.

We first introduce a heuristic-based grouping rule (abbreviated as $\mathcal{HG}$) that clusters the candidate actions according to their resource usages at the root node. Algorithm 4.2 summarises the overall procedure using $\mathcal{HG}$. Let $\hat{r}_{n,k}$ be the number of idle type k resources associated with candidate action $a_{n,t} = (X_{n,t}, Y_{n,t})$ where

$$X_{n,t} = \{x^n_{i,j,t} |\ \ i \in \mathcal{I}, j \in \mathcal{J}\} \text{ and } Y_{n,t} = \{y^{\ell,n}_{i,j,t} |\ \ i \in \mathcal{I}, j \in \mathcal{J}, \ell \in \mathcal{L}\}.$$

To cluster actions $A \in \mathcal{A}_t$, we compute the idle type k resources for each action $a_{n,t} \in A$ by

$$\tilde{r}_{n,k} = r_{k,t} - \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{J}} \left[(\sum_{\ell \in \mathcal{L}} y^{\ell,n}_{i,j,t} \times \hat{h}^{\ell}_{i,j,k}) + x^n_{i,j,t} \times h_{i,j,k} \right], \quad \forall k \in \mathcal{K}. \tag{4.28}$$

Let us define sets B^x_n and B^y_n for candidate action $a_{n,t} = (X_{n,t}, Y_{n,t})$ as follows,

$$B^x_n = \{(i,j) |\ \ x^n_{i,j,t} \in X_{n,t}, \quad x^n_{i,j,t} = 0\}, \tag{4.29}$$

and

$$B^y_n = \{(i,j,\ell) |\ \ y^{\ell,n}_{i,j,t} \in Y_{n,t}, \quad y^{\ell,n}_{i,j,t} = 0\} \tag{4.30}$$

Algorithm 4.2: Heuristic-based Grouping Rule ($\mathcal{HG}$)

input : a set of candidate actions A
output: grouping results $A^0, A^1, A^2, \ldots$

set $\theta = 0$;
while $A \neq \emptyset$ **do**
  for $a_{n,t} \in A$ **do**
    compute the idle resources using (4.28);
    identify unused scheduling and acceleration measures via (4.29) and (4.30);
    if set $G(\theta, a_{n,t})$ is empty **then**
    update $A^\theta \leftarrow A^\theta \cup a_{n,t}$ and $A \leftarrow A \backslash \{a_{n,t}\}$;
    end
  end
  set $\theta \leftarrow \theta + 1$
end

These sets basically identify unused scheduling and acceleration measures. respectively. We define parameter $\theta \in \mathbb{N}$, which regulates the maximum number of unused scheduling and acceleration measures that can be performed by using the idle resources. A group of candidate actions associated with the parameter θ is denoted as A^θ. The grouping procedure begins with $\theta = 0$, which implies that no additional scheduling and acceleration measures can be performed. We identify the affiliation of each candidate action by verifying if the following set $G(\theta, a_{n,t})$ associated with candidate action $a_{n,t} \in A$ is empty,

$$G(\theta, a_{n,t}) = \Bigg\{ \sum_{(i,j,\ell) \in B_n^y} \bar{y}_{i,j,t}^{\ell,n} + \sum_{(i,j) \in B_n^x} barx_{i,j,t}^n > \theta, \tag{4.31}$$

$$\bar{x}_{i,j,t}^n \leq d_{i,j,t}, \quad \forall (i,j) \in B_n^x, \tag{4.32}$$

$$\bar{y}_{i,j,t}^{\ell,n} \leq \bar{x}_{i,j,t}^n + e_{i,j,t}, \quad \forall (i,j,\ell) \in B_n^y, \tag{4.33}$$

$$\sum_{(i,j,\ell) \in B_n^y} \bar{y}_{i,j,t}^{\ell,n} \times \hat{h}_{i,j,k}^{\ell} + \sum_{(i,j) \in B_n^x} \bar{x}_{i,j,t}^n \times h_{i,j,k} \leq \tilde{r}_{n,k}, \quad \forall k \in \mathcal{K}, \tag{4.34}$$

$$\bar{x}_{i,j,t}^n \in \{0,1\}, \quad \forall (i,j) \in B_n^x; \quad \bar{y}_{i,j,t}^{\ell,n} \in \{0,1\}, \quad \forall (i,j,\ell) \in B_n^y \Bigg\}.$$

Constraint (4.31) determines whether the number of unused scheduling and acceleration measures to be performed is more than θ. In Constraints (4.32), (4.33) and (4.34), we regulate that performing used measures should satisfy the current state of the system. The verification process can be regarded as solving a constraint satisfaction problem. We examine the existence of a feasible solution by using the constraint

Algorithm 4.3: Augmented Sampling Procedure $\mathcal{HG}$-$\mathcal{CB}$

input : root node $\mathcal{S}_t$, optimistic policy Π^O, a set of candidate actions A, significance level α, indifference coefficient β
output: action with the highest sample mean

```
run Algorithm 4.2 to cluster candidate actions: A^0, A^1, ...;
set a* = ∅;
Simulation stage:
while simulation budget exists do
    for θ = 0, 1, ... do
        set A^sim ← A^θ ∪ a*;
        run Algorithm 4.1 for A^sim to identify the best candidate a*;
    end
end
```

propagation method [Apt, 2003]. The same steps are repeated for incremental values of θ until all candidate actions have found their affiliations.

We now introduce an augmented sampling strategy (abbreviated as $\mathcal{HG}$-$\mathcal{CB}$) as described in Algorithm 4.3, that takes advantage of the grouping rule $\mathcal{HG}$. This strategy starts the exploration from the group with smaller θ values. This is because intuitively, it should be beneficial to avoid idle resources. Given the grouping results $A^0, A^1, A^2, \ldots$ and significance level α, we first use the $\mathcal{CB}$ strategy to identify the best action a^* from group A^0. Then, we shift the simulation process to the group A^1. Let A^{sim} denote the set of candidate actions for the new simulation problem. At this new round, the set A^{sim} is comprised of the best candidate a^* identified from the group A^0 and candidate actions in the current group A^1, i.e. $A^{sim} = \{A^1 \cup a^*\}$. After identifying the best action a^* for A^{sim}, $\mathcal{HG}$-$\mathcal{CB}$ carries on the aforementioned steps for the remaining groups.

4.7 Numerical Experiments

In this section, we describe the experimental setup and report computational results to demonstrate the efficacy of using the optimistic policy, and $\mathcal{CB}$ and $\mathcal{HG}$-$\mathcal{CB}$ sampling strategies in the rollout algorithm.

4.7.1 Experimental Setup

We have created three benchmark problems (abbreviated as **BP-S**, **BP-M** and **BP-L**) for the numerical experiments. Data for these problems can be downloaded from

Table 4.2: Description of test problems.

Problem Instances	Time Periods (weeks)	Resource Types	Drug Projects	Acceleration Measures	Units of Resources
BP-S	1040	2	5	30	type 1: 50 type 2: 60
BP-M	1040	2	7	42	type 1: 55 type 2: 65
BP-L	1040	2	10	90	type 1: 65 type 2: 65

the website[1]. A brief description of benchmark problems (in terms of resource types, number of available resources, drug projects and acceleration measures) is presented in Table 4.2. We assume that the planning horizon is 20 years, with each period corresponding to one week. Therefore, this experimental setup yields $T = 1040$. Two types of resources are allocated to drug projects. **BP-S** is assumed to have 50 and 60 units of types 1 and 2 resources, respectively. On the other hand, **BP-M** has 55 units of type 1 resource and 65 units of type 2 resource, while **BP-L** has 65 units of types 1 and 2 resources. The test instances **BP-S** and **BP-M** respectively involve five and seven drug projects and have two kinds of acceleration measures for each clinical trial of a drug project. The large-sized benchmark problem **BP-L** has 10 drug projects and three acceleration measures for each clinical trial.

The algorithmic parameter settings of $\mathcal{CB}$ and $\mathcal{HG}$-$\mathcal{CB}$ sampling strategies is set as follows: the significance level α is fixed at 5×10^{-8} and the indifference coefficient is set to be 1. Moreover, if any actions has not been explored, we apply the CRN method to generate 20 samples of uncertain task outcomes and evaluate the corresponding cumulative rewards.

We run the numerical experiments on a machine with Intel i7-6700K CPU and 32GB memory. These experiments are coded in Python 3.6. The optimistic decision rule at each state is obtained by using the CPLEX mathematical programming solver of which the relative gap tolerance is set to be 0.005. The heuristic-based grouping rule implemented is based on the CPLEX constraint programming solver.

4.7.2 Average Performance Comparison of Base Policies

We are first concerned with the performance of optimistic policy Π^O by comparing with the following heuristic policies (used in the study of Choi et al. [2004]) as

[1] `https://www.researchgate.net/publication/327108858_Benchmark_Problems_BP-S_BP-M_BP-L`.

Table 4.3: Average cumulative rewards obtained by different base policies integrated within the rollout algorithm.

Problems	Sampling	Base Policies		
Instances	Strategies	Π^{O}	$\Pi^{\mathcal{HSPT}}$	$\Pi^{\mathcal{HERP}}$
BP-S	$\mathcal{CB}$	**3059.23**	2235.95	2642.68
	$\mathcal{HG}$-$\mathcal{CB}$	**3089.81**	2263.07	2713.41
BP-M	$\mathcal{CB}$	**5176.79**	3952.51	3249.61
	$\mathcal{HG}$-$\mathcal{CB}$	**5226.67**	4005.18	3413.08
BP-L	$\mathcal{CB}$	**8059.91**	6135.72	6437.82
	$\mathcal{HG}$-$\mathcal{CB}$	**8211.53**	6220.57	6575.74

The best results (highest rewards) are highlighted in bold.

benchmark approaches.

- **High success probability task first** ($\Pi^{\mathcal{HSPT}}$): This approach determines the action for each child node according to the success probabilities of available tasks.
- **High expected revenue project first** ($\Pi^{\mathcal{HERP}}$): This is a greedy-like scheduling heuristic that computes the conditionally expected profit of each drug project without using any acceleration measures and gives priority to the drug projects with highest conditionally expected profits.

The experimental settings are as follows. As the performance criterion for the comparison of different base policies, we use the average cumulative reward over various sample trajectories. For each problem instance, we generate 100 CRN samples of uncertain task outcomes to evaluate the performance of various base policies integrated within the rollout algorithm. The $\mathcal{CB}$ and $\mathcal{HG}$-$\mathcal{CB}$ sampling strategies are implemented to support the rollout algorithm. We employ a measure of "*average evaluation*" (a.e.) to specify the total simulation budget at each node. For instance, if the root node has 200 candidate actions, 50 a.e. corresponds to $200 \times 50 = 10,000$ simulation replications. In this experiment, the average evaluation of $\mathcal{CB}$ and $\mathcal{HG}$-$\mathcal{CB}$ at each state is fixed at 150. Table 4.3 displays the average cumulative rewards obtained by different base policies integrated within the rollout algorithm. Table 4.4 presents the Bernstein confidence interval for the pairwise performance difference between distinct base policies under the significance level $\alpha = 0.025$.

As shown in Tables 4.3 and 4.4, the optimistic decision rule has the best performance in all test problems, indicating the importance of considering the overall profit of drug projects in the base policy design. We also observe that $\Pi^{\mathcal{HERP}}$

Table 4.4: Bernstein confidence interval for the pairwise performance difference between distinct base policies (significance level $\alpha = 0.025$).

BP-S

$\mathcal{CB}$		Π^{O}	$\Pi^{\mathcal{HSPT}}$	$\Pi^{\mathcal{HERP}}$	$\mathcal{HG}$-$\mathcal{CB}$		Π^{O}	$\Pi^{\mathcal{HSPT}}$	$\Pi^{\mathcal{HERP}}$
	Π^{O}		823.28±421.53	416.55±362.05		Π^{O}		826.74±494.87	376.40±359.68
	$\Pi^{\mathcal{HSPT}}$	-823.28±421.53		-406.73±542.19		$\Pi^{\mathcal{HSPT}}$	-826.74±494.87		-450.34±465.92
	$\Pi^{\mathcal{HERP}}$	-416.55±362.05	406.73±542.19			$\Pi^{\mathcal{HERP}}$	-376.40±359.68	450.34±465.92	

BP-M

$\mathcal{CB}$		Π^{O}	$\Pi^{\mathcal{HSPT}}$	$\Pi^{\mathcal{HERP}}$	$\mathcal{HG}$-$\mathcal{CB}$		Π^{O}	$\Pi^{\mathcal{HSPT}}$	$\Pi^{\mathcal{HERP}}$
	Π^{O}		1224.28±683.07	1927.18±556.90		Π^{O}		1221.49±657.40	1813.59±528.26
	$\Pi^{\mathcal{HSPT}}$	-1224.28±683.07		702.90±625.41		$\Pi^{\mathcal{HSPT}}$	1221.49±657.40		592.10±629.71
	$\Pi^{\mathcal{HERP}}$	-1927.18±556.90	-702.90±625.41			$\Pi^{\mathcal{HERP}}$	1813.59±528.26	-592.10±629.71	

BP-L

$\mathcal{CB}$		Π^{O}	$\Pi^{\mathcal{HSPT}}$	$\Pi^{\mathcal{HERP}}$	$\mathcal{HG}$-$\mathcal{CB}$		Π^{O}	$\Pi^{\mathcal{HSPT}}$	$\Pi^{\mathcal{HERP}}$
	Π^{O}		1924.19±794.92	1622.09±790.01		Π^{O}		1990.96±891.40	1635.79±593.73
	$\Pi^{\mathcal{HSPT}}$	1924.19±794.92		-302.10±199.30		$\Pi^{\mathcal{HSPT}}$	-1990.96±891.40		-355.17±319.58
	$\Pi^{\mathcal{HERP}}$	-1622.09±790.01	302.10±199.30			$\Pi^{\mathcal{HERP}}$	-1635.79±593.73	355.17±319.58	

Table 4.5: Base policy comparison: average CPU time per decision in seconds

Problem Instances	Base Policies		
	Π^{O}	$\Pi^{\mathcal{HSPT}}$	$\Pi^{\mathcal{HERP}}$
BP-S	0.51	0.32	0.38
BP-M	0.57	0.37	0.43
BP-L	0.63	0.45	0.46

presents a performance advantage over $\Pi^{\mathcal{HSPT}}$ in most test instances although the difference is small relative to the difference of these two base policies to Π^{O}. In particular, the differences between $\Pi^{\mathcal{HERP}}$ and $\Pi^{\mathcal{HSPT}}$ for the problem instances **BP-S** and **BP-M** are not statistically significant at the 0.025 significance level when applying the $\mathcal{HG}$-$\mathcal{CB}$ sampling strategy. On average, Π^{O} is by 31% better than the average other base policies. From the performance comparison of $\mathcal{CB}$ and $\mathcal{HG}$-$\mathcal{CB}$, we observe that $\mathcal{HG}$-$\mathcal{CB}$ contributes to the performance of all base policies more than $\mathcal{CB}$. This result confirms the advantage of integrating the grouping rule within the sampling strategy.

Table 4.5 presents the average CPU time needed for various base policies to identify an action in three test instances. The results show that the computational effort of Π^{O} is slightly larger than that of $\Pi^{\mathcal{HERP}}$ and $\Pi^{\mathcal{HSPT}}$. It is because Π^{O} identifies the action by using the rollout horizon optimisation framework. The CPU time of using Π^{O} policy at first several periods might be large, though we have applied the complexity reduction method that ignores the resource constraints for the future states.

Table 4.6: Average cumulative rewards obtained by different sampling strategies at varying simulation budget.

Problem Instances	Simulation Budgets (a.e.)	Sampling Strategies				
		$\mathcal{CB}$	$\mathcal{HG}$-$\mathcal{CB}$	UCB1	ϵ-greedy	Boltzmann
BP-S	50	2996.86	**3021.26**	2805.11	2924.57	2893.67
	100	**3037.17**	**3044.15**	2832.02	2898.91	2894.89
	150	3059.23	**3089.81**	2841.48	2902.54	2901.26
BP-M	50	5119.53	**5181.02**	5084.96	5126.18	5124.51
	100	5148.67	**5186.65**	5091.78	5137.63	5141.66
	150	5176.84	**5226.67**	5099.50	5155.84	5185.59
BP-L	50	8038.60	**8123.52**	7735.25	7826.10	7858.53
	100	8048.68	**8182.97**	7739.52	7862.68	7961.68
	150	8059.91	**8211.53**	7747.83	7930.93	8030.60

The best results (highest values) and those statistically not different from the best are highlighted in bold.

4.7.3 Average Performance Comparison with Benchmark Sampling Strategies

In this experiment, we investigate the average performance of the proposed sampling strategies, namely $\mathcal{CB}$ and $\mathcal{HG}$-$\mathcal{CB}$. We consider the following bandit policies for the performance comparison.

- **The UCB1 algorithm** is a sampling algorithm that uses the Chernoff-Hoeffding confidence bound for estimating the expected cumulative reward [Auer et al., 2002]. At each simulation stage, the algorithm performs a new evaluation for the action with the largest upper confidence bound.

- **The ϵ-greedy policy** exploits the best action so far (with probability $1 - \epsilon$) and randomly simulates an action with probability ϵ [Kuleshov and Precup, 2014]. The ϵ is fixed at 0.05.

- **The Boltzmann policy** applies the Boltzmann distributions of each candidate action to determine the evaluation at each simulation stage [Fernández and Veloso, 2006]. The temperature is specified as 0.05.

Note that all benchmark methods are non-CRN sampling strategies, which use independent samples in the simulation process.

We generate 100 CRN samples of uncertain task outcomes for each problem instance. These samples are used to evaluate the average performances of various sampling approaches so that all performance differences can be attributed to the

sampling strategy alone. In this experiment, we consider three different average evaluations of 50, 100 and 150 as simulation budgets for each sampling approach. Table 4.6 presents the performance of various sampling methods on test instances **BP-S**, **BP-M**, and **BP-L** in terms of average cumulative reward (from $t = 1$ to $t = 1040$) over 100 samples of uncertain task outcomes. Table 4.7 reports the Bernstein confidence intervals for the pairwise difference between sampling strategies at the significance level $\alpha = 0.025$.

The results show that both $\mathcal{CB}$ and $\mathcal{HG}$-$\mathcal{CB}$ perform consistently better than the other approaches in all test instances, because the CRN technique can induce positive correlations for the pairwise comparison procedure and provides statistically fair comparisons for all candidate actions. The benchmark approaches using independent samples display a very similar performance pattern. In the drug development model, the sample of task uncertain outcomes significantly influences the cumulative reward. When the performances of candidate actions are evaluated by using different samples, the sampling strategies can be misled and tends to select an incorrect candidate action. We also observe that $\mathcal{HG}$-$\mathcal{CB}$ presents performance advantages over $\mathcal{CB}$ for all test instances at three various simulation budgets. These results once again confirm the advantage of integrating the heuristic-based grouping rule with a sampling strategy.

Table 4.7: Bernstein confidence interval for the pairwise difference between distinct strategies (significance level $\alpha = 0.025$).

BP-S

		$\mathcal{CB}$	$\mathcal{HG}$-$\mathcal{CB}$	UCB1	ϵ-greedy	Boltzmann
50	$\mathcal{CB}$		-24.40±20.50	191.75±176.44	72.29±68.81	103.19±97.52
	$\mathcal{HG}$-$\mathcal{CB}$	24.40±20.50		216.15±181.80	96.69±63.71	127.59±84.27
	UCB1	-191.75±176.44	-216.15±181.80		-119.46±83.16	-88.56±69.71
a.e.	ϵ-greedy	-72.29±68.81	-96.69±63.71	119.46±83.16		30.9±34.2
	Boltzmann	-103.19±97.52	-127.59±84.27	88.56±69.71	-30.9±34.2	

		$\mathcal{CB}$	$\mathcal{HG}$-$\mathcal{CB}$	UCB1	ϵ-greedy	Boltzmann
100	$\mathcal{CB}$		-6.98±10.51	205.15±169.40	138.26±94.82	142.28±137.51
	$\mathcal{HG}$-$\mathcal{CB}$	6.98±10.51		212.13±178.71	145.24±126.58	149.26±135.72
	UCB1	-205.15±169.40	-212.13±178.71		-66.89±60.38	-62.87±53.84
a.e.	ϵ-greedy	-138.26±94.82	-145.24±126.58	66.89±60.38		4.02±8.57
	Boltzmann	-142.28±137.51	-149.26±135.72	62.87±53.84	-4.02±8.57	

		$\mathcal{CB}$	$\mathcal{HG}$-$\mathcal{CB}$	UCB1	ϵ-greedy	Boltzmann
150	$\mathcal{CB}$		-30.58±28.11	217.75±197.45	156.69±119.32	157.97±104.64
	$\mathcal{HG}$-$\mathcal{CB}$	30.58±28.11		248.33±185.91	187.27±155.84	188.55±142.63
	UCB1	-217.75±197.45	-248.33±185.91		-61.06±57.89	-59.78±54.59
a.e.	ϵ-greedy	-156.69±119.32	-187.27±155.84	61.06±57.89		1.28±5.61
	Boltzmann	-157.97±104.64	-188.55±142.63	59.78±54.59	-1.28±5.61	

BP-M

		$\mathcal{CB}$	$\mathcal{HG}$-$\mathcal{CB}$	UCB1	ϵ-greedy	Boltzmann
50	$\mathcal{CB}$		-61.49±52.54	34.57±23.16	-6.65±8.21	-4.98±6.14
	$\mathcal{HG}$-$\mathcal{CB}$	61.49±52.54		96.06±79.30	54.84±46.51	56.51±49.36
	UCB1	-34.57±23.16	-96.06±79.30		-41.22±39.13	-39.55±34.87
a.e.	ϵ-greedy	6.65±8.21	-54.84±46.51	41.22±39.13		1.67±3.35
	Boltzmann	4.98±6.14	-56.51±49.36	39.55±34.87	-1.67±3.35	

		$\mathcal{CB}$	$\mathcal{HG}$-$\mathcal{CB}$	UCB1	ϵ-greedy	Boltzmann
100	$\mathcal{CB}$		-37.98±32.26	56.89±54.40	11.04±10.63	7.01±9.52
	$\mathcal{HG}$-$\mathcal{CB}$	37.98±32.26		94.87±72.10	49.02±45.34	44.99±43.41
	UCB1	-56.89±54.40	-94.87±72.10		-45.85±36.28	-49.88±45.86
a.e.	ϵ-greedy	-11.04±10.63	-49.02±45.34	45.85±36.23		-4.03±8.33
	Boltzmann	-7.01±9.52	-44.99±43.41	49.88±45.86	4.03±8.33	

		$\mathcal{CB}$	$\mathcal{HG}$-$\mathcal{CB}$	UCB1	ϵ-greedy	Boltzmann
150	$\mathcal{CB}$		-49.83±41.87	77.34±69.50	21.00±17.56	-8.75±7.96
	$\mathcal{HG}$-$\mathcal{CB}$	49.83±41.87		127.17±109.79	70.83±59.04	41.08±30.94
	UCB1	-77.34±69.50	-127.17±109.79		-56.34±48.93	-86.09±79.55
a.e.	ϵ-greedy	-21.00±17.56	-70.83±59.04	56.34±48.93		-29.75±28.08
	Boltzmann	8.75±7.96	-41.08±30.94	86.09±79.55	29.75±28.08	

BP-L

		$\mathcal{CB}$	$\mathcal{HG}$-$\mathcal{CB}$	UCB1	ϵ-greedy	Boltzmann
50	$\mathcal{CB}$		-84.92±70.16	303.35±276.59	212.50±207.44	180.07±196.07
	$\mathcal{HG}$-$\mathcal{CB}$	84.92±70.16		388.27±389.80	297.42±275.11	264.99±241.58
	UCB1	-303.35±276.59	-388.27±389.80		-90.85±98.47	-123.28±122.41
a.e.	ϵ-greedy	-212.50±207.44	-297.42±275.11	90.85±98.47		-32.43±43.58
	Boltzmann	-180.07±196.07	-264.99±241.58	123.28±122.41	32.43±43.58	

		$\mathcal{CB}$	$\mathcal{HG}$-$\mathcal{CB}$	UCB1	ϵ-greedy	Boltzmann
100	$\mathcal{CB}$		-134.29±100.21	309.16±337.20	186.00±178.44	87.00±85.03
	$\mathcal{HG}$-$\mathcal{CB}$	134.29±100.21		443.45±414.91	320.29±285.60	221.29±274.22
	UCB1	-309.16±337.20	-443.45±414.91		-123.16±102.49	-222.16±212.39
a.e.	ϵ-greedy	-186.00±178.44	-320.29±285.60	123.16±102.49		-99.00±96.31
	Boltzmann	-87.00±85.03	-221.29±274.22	222.16±212.39	99.00±96.31	

		$\mathcal{CB}$	$\mathcal{HG}$-$\mathcal{CB}$	UCB1	ϵ-greedy	Boltzmann
150	$\mathcal{CB}$		-151.62±149.54	312.08±258.83	128.98±126.54	29.31±35.13
	$\mathcal{HG}$-$\mathcal{CB}$	151.62±149.54		463.70±452.64	280.60±267.85	180.93±171.49
	UCB1	-312.08±258.83	-463.70±452.64		-183.10±179.30	-282.77±261.49
a.e.	ϵ-greedy	-128.98±126.54	-280.60±267.8	183.10±179.30		-99.66±85.91
	Boltzmann	-29.31±35.13	-180.93±171.49	282.77±261.49	99.66±85.91	

4.8 Conclusions

In this Chapter, we introduced a discrete-time finite MDP formulation of the clinical trial scheduling and resource allocation problem, where the outcome of a clinical trial is uncertain and the revenue of the approved drug depends on the project completion time. In the proposed model, acceleration measures may be applied to compress the processing time of a clinical trial. We propose an adaptive rollout algorithm to identify a cost-effective scheduling and resource allocation strategy. The adaptive rollout algorithm uses an optimistic policy to construct corresponding search trees for estimating possible outcomes after the selected actions are implemented. We show that the rollout policy is as good as or better than the optimistic policy. To improve the simulation efficiency, we propose a $\mathcal{CB}$ sampling strategy to guide the growth of the search tree for candidate actions. We show that $\mathcal{CB}$ can terminate within a finite simulation budget and have a certain performance guarantee. Moreover, we introduce an augmented sampling strategy ($\mathcal{HG}$-$\mathcal{CB}$) that uses a heuristic-based grouping rule to adjust the exploration priority in the simulation procedure. In the numerical experiments, we demonstrate the advantage of applying optimistic policy, $\mathcal{CB}$ and $\mathcal{HG}$-$\mathcal{CB}$ in the rollout algorithm.

Chapter 5

Summary and Future Work

In this thesis, we have developed new sampling strategies that address information collection problems in three different stochastic optimisation frameworks. In sampling-based stochastic programs, the information collection problem is related to a post-processing procedure with numerous potential solutions and a relatively limited simulation budget. We develop a two-stage measurement strategy, in which the key component is a Wasserstein-based screening rule that can remove non-promising solutions from the simulation stage. The results show that the screening approach can improve the efficiency of optimal computing budget allocation approach and that better solutions can be found within the given computational budget.

In the context of stochastic black-box optimisation, an evolutionary strategy is used to search for the solution with the best mean performance over a disturbance region. In each iteration, the evolutionary strategy identifies a promising search direction by generating several potential solutions and comparing their mean performances. To reduce the computational cost, we use an archive sample approximation approach that takes into account previous simulation results when estimating the mean performance. A limited computational budget is available to evaluate the fitness value of a few additional points in the decision space. In this context, it is desirable to identify the best sampling locations that most improve the estimation accuracy of solutions' mean performances. We develop new sampling strategies that use the Wasserstein distance metric to suggest promising sampling locations. We empirically examine the performance of our new sampling strategies on six artificial benchmark problems and one robust airfoil shape optimisation problem. The superiority of using our sampling strategies is confirmed in all test instances.

For the drug development problem investigated in this thesis, an adaptive rollout algorithm is introduced to approximately solve the dynamic programming

model. The adaptive rollout algorithm involves two methodological innovations. One is an optimistic policy that utilises the rolling horizon optimisation framework to find the appropriate move at the child nodes. Another innovation is the adaptive sampling strategy that addresses the information collection problem arising from the rollout algorithm. We exploit the common random number technique and empirical Bernstein confidence bound in a statistical racing procedure to improve measurement efficiency. Our results show that the adaptive sampling strategy contributes to the performance of a rollout algorithm more than heuristic bandit strategies.

This thesis has presented several new sampling approaches and empirically demonstrated their performance advantage over existing state-of-the-art approaches on various test problems. Several potential extensions can be envisaged to the proposed approaches.

- The two-stage measurement strategy in Chapter 2 might be tailored by developing a sophisticated method for choosing the number of potential solutions to be evaluated in the second-stage simulation process. One also can include some stopping rules to decide when to stop measurement. If the simulation time is sufficient, then the optimal computing budget allocation approach can be applied to the potential solutions that are eliminated by the first-stage screening process. The screening rule may also be extended to multi-stage stochastic programs.

- The efficient sampling strategies in Chapter 3 can be further developed in several possible directions. The idea of sampling budget control can be refined with more sophisticated control strategies. The approximation region is only helpful at the beginning of the run, so the parameter controlling the extension can be reduced over the run. Moreover, surrogate-based approaches can be employed to estimate the gradient information and achieve a tighter upper bound approximation for the estimation error.

- The adaptive sampling strategy in Chapter 4 can be integrated with the Monte Carlo tree search algorithm to address more complex real-world problems, such as playing board games [Silver et al., 2017], card games [Cowling et al., 2012] and car-sharing problems [Jiang et al., 2017]. The upper confidence bound policy is the dominant sampling approach of the Monte Carlo tree search algorithm. However, in the rollout settings, the numerical results have shown that our sampling strategy is better than the upper confidence bound policy. It would be interesting to compare our sampling strategy with the upper confidence bound policy in Monte-Carlo tree search.

Appendix A

The Wasserstein Metric

The Wasserstein metric (also known as the Earth mover's distance) is a statistical distance between two probability measures, which can be computed by solving the Monge-Kantorovich transportation problem [Villani, 2008]. The two probability measures in the Wasserstein distance are considered as warehouses and destinations. The distance value between two probability measures is determined by the minimum transportation cost from warehouses to destinations. Let $\mathbb{P}$ and $\mathbb{Q}$ denote two discrete probability distributions with samples $G = \{g_j |\ j = 1, \ldots, J\}$ and $H = \{h_k |\ k = 1, \ldots, K\}$. Let ψ represent the joint probability distribution with marginal probabilities $\mathbb{P}$ and $\mathbb{Q}$. The Wasserstein distance $W(\mathbb{P}, \mathbb{Q})$ between discrete probability measures $\mathbb{P}$ and $\mathbb{Q}$ can be formulated as the following optimisation problem.

$$
\begin{aligned}
W(\mathbb{P}, \mathbb{Q}) = \min_{\psi_{j,k}} & \sum_{k=1}^{K} \sum_{j=1}^{J} d(g_j, h_k) \psi_{j,k} \\
s.t. \ & \sum_{j=1}^{J} \psi_{j,k} = \mathbb{Q}(h_k), \ \forall k \\
& \sum_{k=1}^{K} \psi_{j,k} = \mathbb{P}(g_j), \ \forall j \\
& \sum_{k=1}^{K} \mathbb{Q}(h_k) = 1, \sum_{j=1}^{J} \mathbb{P}(g_j) = 1 \\
& \psi_{j,k} \geq 0, \ \forall j, k,
\end{aligned}
\tag{A.1}
$$

where $d(g_j, h_k) = \|g_j - h_k\|_2$ is the Euclidean distance between g_j and h_k.

Bibliography

I.H. Abbott and A.E. Von Doenhoff. *Theory of wing sections, including a summary of airfoil data*. Dover Books on Aeronautical Engineering Series. Dover Publications, New York, 1959.

Mohamed A Ahmed and Talal M Alkhamis. Simulation-based optimization using simulated annealing with ranking and selection. *Computers & Operations Research*, 29(4):387–402, 2002. doi:10.1016/s0305-0548(00)00073-3.

K. Apt. *Principles of constraint programming*. Cambridge University Press, 2003. doi:10.1017/cbo9780511615320.

Jean-Yves Audibert, Rémi Munos, and Csaba Szepesvári. Tuning bandit algorithms in stochastic environments. In *International Conference on Algorithmic Learning Theory*, pages 150–165. Springer, 2007. doi:10.1007/978-3-540-75225-7_15.

Jean-Yves Audibert, Rémi Munos, and Csaba Szepesvári. Exploration–exploitation tradeoff using variance estimates in multi-armed bandits. *Theoretical Computer Science*, 410(19):1876–1902, 2009. doi:10.1016/j.tcs.2009.01.016.

Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47(2-3):235–256, 2002.

Gulay Barbarosoğlu and Yasemin Arda. A two-stage stochastic programming framework for transportation planning in disaster response. *Journal of the Operational Research Society*, 55(1):43–53, 2004. doi:10.1057/palgrave.jors.2601652.

Evelyn ML Beale. On minimizing a convex function subject to linear inequalities. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 173–184, 1955.

Robert E Bechhofer. A single-sample multiple decision procedure for ranking means of normal populations with known variances. *The Annals of Mathematical Statistics*, pages 16–39, 1954.

Patrizia Beraldi, Domenico Conforti, and Antonio Violi. A two-stage stochastic programming model for electric energy producers. *Computers & Operations Research*, 35(10):3360–3370, 2008. doi:10.1016/j.cor.2007.03.008.

Dimitri P Bertsekas. Dynamic programming and suboptimal control: A survey from ADP to MPC. *European Journal of Control*, 11(4-5):310–334, 2005. doi:10.3166/ejc.11.310-334.

Dimitri P. Bertsekas, John N. Tsitsiklis, and Cynara Wu. Rollout algorithms for combinatorial optimization. *Journal of Heuristics*, 3(3):245–262, Dec 1997. ISSN 1572-9397. doi:10.1023/A:1009635226865.

Dimitris Bertsimas and Ramazan Demir. An approximate dynamic programming approach to multidimensional knapsack problems. *Management Science*, 48(4): 550–565, 2002. doi:10.1287/mnsc.48.4.550.208.

Dimitris Bertsimas and Ioana Popescu. Revenue management in a dynamic network environment. *Transportation Science*, 37(3):257–277, 2003. doi:10.1287/trsc.37.3.257.16047.

H-G Beyer and Bernhard Sendhoff. Evolution strategies for robust optimization. In *2006 IEEE International Conference on Evolutionary Computation*, pages 1346–1353, 2006. doi:10.1109/CEC.2006.1688465.

H.-G. Beyer and Bernhard Sendhoff. Robust optimization – A comprehensive survey. *Computer Methods in Applied Mechanics and Engineering*, 196(33):3190–3218, 2007. ISSN 0045-7825. doi:http://dx.doi.org/10.1016/j.cma.2007.03.003.

Hans-Georg Beyer and Bernhard Sendhoff. Towards a steady-state analysis of an evolution strategy on a robust optimization problem with noise-induced multimodality. *IEEE Transactions on Evolutionary Computation*, 21(4):629–643, 2017. doi:10.1109/tevc.2017.2668068.

John R Birge and Francois V Louveaux. A multicut algorithm for two-stage stochastic linear programs. *European Journal of Operational Research*, 34(3):384–392, 1988. doi:10.1016/0377-2217(88)90159-2.

Djallel Bouneffouf and Raphael Féraud. Multi-armed bandit problem with known trend. *Neurocomputing*, 205:16–21, 2016. doi:10.1016/j.neucom.2016.02.052.

Juergen Branke and Xin Fei. Efficient sampling when searching for robust solutions. In *International Conference on Parallel Problem Solving from Nature*, pages 237–246. Springer, 2016. doi:10.1007/978-3-319-45823-6_22.

Jurgen Branke. *Evolutionary optimization in dynamic environments*. Kluwer Academic Publishers, Norwell, MA, USA, 2001. ISBN 0792376315. doi:10.1007/978-1-4615-0911-0.

Jürgen Branke and Christian Schmidt. Selection in the presence of noise. In *Proceedings of the 2003 International Conference on Genetic and Evolutionary Computation*, pages 766–777. Springer, 2003. ISBN 3-540-40602-6. doi:10.1007/3-540-45105-6_91.

Jürgen Branke, Christian Schmidt, and Hartmut Schmeck. Efficient fitness estimation in noisy environments. In *Proceedings of the 3rd Annual Conference on Genetic and Evolutionary Computation*, pages 243–250. Morgan Kaufmann, 2001. ISBN 1-55860-774-9.

Jürgen Branke, Stephen E Chick, and Christian Schmidt. Selecting a selection procedure. *Management Science*, 53(12):1916–1932, 2007. doi:10.1287/mnsc.1070.0721.

Cameron B Browne, Edward Powley, Daniel Whitehouse, Simon M Lucas, Peter I Cowling, Philipp Rohlfshagen, Stephen Tavener, Diego Perez, Spyridon Samothrakis, and Simon Colton. A survey of monte carlo tree search methods. *IEEE Transactions on Computational Intelligence and AI in games*, 4(1):1–43, 2012. doi:10.1109/tciaig.2012.2186810.

Róbert Busa-Fekete, Balázs Szörényi, Paul Weng, Weiwei Cheng, and Eyke Hüllermeier. Preference-based reinforcement learning: evolutionary direct policy search using a preference-based racing algorithm. *Machine Learning*, 97(3): 327–351, 2014. doi:10.1007/s10994-014-5458-8.

Olivier Cappé, Aurélien Garivier, Odalric-Ambrym Maillard, Rémi Munos, Gilles Stoltz, et al. Kullback–leibler upper confidence bounds for optimal sequential allocation. *The Annals of Statistics*, 41(3):1516–1541, 2013. doi:10.1214/13-aos1119.

Alejandro Cervantes, David Quintana, and Gustavo Recio. Efficient dynamic resampling for dominance-based multiobjective evolutionary optimization. *Engineering Optimization*, 49(2):311–327, 2017. doi:10.1080/0305215X.2016.1187729.

Suresh Chand, Vernon Ning Hsu, and Suresh Sethi. Forecast, solution, and rolling horizons in operations management problems: A classified bibliography. *Manufacturing & Service Operations Management*, 4(1):25–43, 2002. doi:10.1287/msom.4.1.25.287.

Hyeong Soo Chang, Michael C Fu, Jiaqiao Hu, and Steven I Marcus. An adaptive sampling algorithm for solving Markov decision processes. *Operations Research*, 53(1):126–139, 2005. doi:10.1287/opre.1040.0145.

Hyeong Soo Chang, Michael C Fu, Jiaqiao Hu, and Steven I Marcus. Recursive learning automata approach to Markov decision processes. *IEEE Transactions on Automatic Control*, 52(7):1349–1355, 2007. doi:10.1109/tac.2007.900859.

Chun-Hung Chen. An effective approach to smartly allocate computing budget for discrete event simulation. In *Proceedings of 1995 34th IEEE Conference on Decision and Control*, volume 3, pages 2598–2603. IEEE, 1995.

Chun-Hung Chen, Jianwu Lin, Enver Yücesan, and Stephen E Chick. Simulation budget allocation for further enhancing the efficiency of ordinal optimization. *Discrete Event Dynamic Systems*, 10(3):251–270, 2000.

Raymond K Cheung and Chuen-Yih Chen. A two-stage stochastic network model and solution methods for the dynamic empty container allocation problem. *Transportation Science*, 32(2):142–162, 1998. doi:10.1287/trsc.32.2.142.

Stephen E Chick and Koichiro Inoue. New procedures to select the best simulated system using common random numbers. *Management Science*, 47(8):1133–1149, 2001. doi:10.1287/mnsc.47.8.1133.10229.

Stephen E Chick, Jürgen Branke, and Christian Schmidt. Sequential sampling to myopically maximize the expected value of information. *INFORMS Journal on Computing*, 22(1):71–80, 2010. doi:10.1287/ijoc.1090.0327.

Jaein Choi, Matthew J Realff, and Jay H Lee. Dynamic programming in a heuristically confined state space: a stochastic resource-constrained project scheduling application. *Computers & Chemical Engineering*, 28(6-7):1039–1058, 2004. doi:10.1016/j.compchemeng.2003.09.024.

Brianna Christian and Selen Cremaschi. Heuristic solution approaches to the pharmaceutical R&D pipeline management problem. *Computers & Chemical Engineering*, 74:34–47, 2015. doi:10.1016/j.compchemeng.2014.12.014.

Matthew Colvin and Christos T Maravelias. A stochastic programming approach for clinical trial planning in new drug development. *Computers & Chemical Engineering*, 32(11):2626–2642, 2008. doi:10.1016/j.compchemeng.2007.11.010.

Peter I Cowling, Colin D Ward, and Edward J Powley. Ensemble determinization in Monte-Carlo tree search for the imperfect information card game magic: The gathering. *IEEE Transactions on Computational Intelligence and AI in Games*, 4(4):241–257, 2012. doi:10.1109/tciaig.2012.2204883.

George B Dantzig. Linear programming under uncertainty. *Management Science*, 1 (3-4):197–206, 1955. doi:10.1287/mnsc.1040.0261.

George B Dantzig and Philip Wolfe. Decomposition principle for linear programs. *Operations Research*, 8(1):101–111, 1960. doi:10.1287/opre.8.1.101.

Bert De Reyck and Roel Leus. R&D project scheduling when activities may fail. *IIE transactions*, 40(4):367–384, 2008. doi:10.1080/07408170701413944.

Boris Defourny, Damien Ernst, and Louis Wehenkel. Scenario trees and policy selection for multistage stochastic programming using machine learning. *INFORMS Journal on Computing*, 25(3):488–501, 2013. doi:10.1287/ijoc.1120.0516.

Boris Defourny, Ilya O Ryzhov, and Warren B Powell. Optimal information blending with measurements in the l_2 sphere. *Mathematics of Operations Research*, 40(4): 1060–1088, 2015. doi:10.1287/moor.2015.0712.

Michael AH Dempster, Elena A Medova, and Yee Sook Yong. Comparison of sampling methods for dynamic stochastic programming. In *Stochastic Optimization Methods in Finance and Energy*, pages 389–425. Springer, 2011.

Anthony Di Pietro, Lyndon While, and Luigi Barone. Applying evolutionary algorithms to problems with noisy, time-consuming fitness functions. In *Proceedings of the 2004 Congress on Evolutionary Computation*, volume 2, pages 1254–1261, 2004. doi:10.1109/CEC.2004.1331041.

Mary Dillon, Fabricio Oliveira, and Babak Abbasi. A two-stage stochastic programming model for inventory management in the blood supply chain. *International Journal of Production Economics*, 187:27–41, 2017. doi:10.1016/j.ijpe.2017.02.006.

Mark Drela. XFOIL: An analysis and design system for low Reynolds number airfoils. In *Low Reynolds Number Aerodynamics*, pages 1–12. Springer, 1989. doi:10.1007/978-3-642-84010-4_1.

Jitka Dupačová. Portfolio optimization via stochastic programming: Methods of output analysis. *Mathematical Methods of Operations Research*, 50(2):245–270, 1999. doi:10.1007/s001860050097.

Jitka Dupačová, Nicole Gröwe-Kuska, and Werner Römisch. Scenario reduction in stochastic programming. *Mathematical Programming*, 95(3):493–511, 2003. doi:10.1007/s10107-002-0331-0.

Andre Esteva, Brett Kuprel, Roberto A Novoa, Justin Ko, Susan M Swetter, Helen M Blau, and Sebastian Thrun. Dermatologist-level classification of skin cancer with deep neural networks. *Nature*, 542(7639):115, 2017. doi:10.1038/nature21056.

Weiwei Fan, L Jeff Hong, and Barry L Nelson. Indifference-zone-free selection of the best. *Operations Research*, 64(6):1499–1514, 2016.

Yonghan Feng and Sarah M. Ryan. Solution sensitivity-based scenario reduction for stochastic unit commitment. *Computational Management Science*, 13(1):29–62, sep 2014. doi:10.1007/s10287-014-0220-z.

Fernando Fernández and Manuela Veloso. Probabilistic policy reuse in a reinforcement learning agent. In *Proceedings of the 5-th International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 720–727. ACM, 2006.

Peter Frazier, Warren Powell, and Savas Dayanik. The knowledge-gradient policy for correlated normal beliefs. *INFORMS journal on Computing*, 21(4):599–613, 2009. doi:10.1287/ijoc.1080.0314.

Haobo Fu, Bernhard Sendhoff, Ke Tang, and Xin Yao. Robust optimization over time: Problem difficulties and benchmark problems. *IEEE Transactions on Evolutionary Computation*, 19(5):731–745, 2015. doi:10.1109/tevc.2014.2377125.

Michael C Fu, Jian-Qiang Hu, Chun-Hung Chen, and Xiaoping Xiong. Simulation allocation for determining the best design in the presence of correlated sampling. *INFORMS Journal on Computing*, 19(1):101–111, 2007.

Siyang Gao and Weiwei Chen. Efficient subset selection for the expected opportunity cost. *Automatica*, 59:19–26, 2015. doi:10.1016/j.automatica.2015.06.005.

Aurélien Garivier and Eric Moulines. On upper-confidence bound policies for switching bandit problems. In *International Conference on Algorithmic Learning Theory*, pages 174–188. Springer, 2011. doi:10.1007/978-3-642-24412-4_16.

John Gittins, Kevin Glazebrook, and Richard Weber. *Multi-armed bandit allocation indices*. John Wiley & Sons, 2011. doi:10.1002/9780470980033.

John C Gittins. Bandit processes and dynamic allocation indices. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 148–177, 1979.

KD Glazebrook, DJ Hodge, and Christopher Kirkbride. Monotone policies and indexability for bidirectional restless bandits. *Advances in Applied Probability*, 45 (1):51–85, 2013. doi:10.1017/s0001867800006194.

Kevin D Glazebrook, Diego Ruiz-Hernandez, and Christopher Kirkbride. Some indexable families of restless bandit problems. *Advances in Applied Probability*, 38(3):643–672, 2006. doi:10.1017/s000186780000121x.

Justin C Goodson, Jeffrey W Ohlmann, and Barrett W Thomas. Rollout policies for dynamic solutions to the multivehicle routing problem with stochastic demand and duration limits. *Operations Research*, 61(1):138–154, 2013. doi:10.1287/opre.1120.1127.

Justin C Goodson, Barrett W Thomas, and Jeffrey W Ohlmann. A rollout algorithm framework for heuristic solutions to finite-horizon stochastic dynamic programs. *European Journal of Operational Research*, 258(1):216–229, 2017. doi:10.1016/j.ejor.2016.09.040.

Nalan Gülpınar, Berç Rustem, and Reuben Settergren. Simulation and optimization approaches to scenario tree generation. *Journal of Economic Dynamics and Control*, 28(7):1291–1315, 2004. doi:10.1016/s0165-1889(03)00113-1.

Gül Gürkan, A Yonca Özge, and Stephen M Robinson. Sample-path optimization in simulation. In *Proceedings of the 26th conference on Winter simulation*, pages 247–254. Society for Computer Simulation International, 1994.

Nikolaus Hansen. CMA-ES source code. URL `http://cma.gforge.inria.fr/cmaes_sourcecode_page.html`.

Nikolaus Hansen and Andreas Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, 9(2):159–195, 2001. doi:10.1162/106365601750190398.

Michael Hay, David W Thomas, John L Craighead, Celia Economides, and Jesse Rosenthal. Clinical development success rates for investigational drugs. *Nature Biotechnology*, 32(1):40, 2014. doi:10.1038/nbt.2786.

Donghai He, Stephen E Chick, and Chun-Hung Chen. Opportunity cost and OCBA selection procedures in ordinal optimization for a fixed number of alternative

systems. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 37(5):951–961, 2007. doi:10.1109/tsmcc.2007.900656.

Verena Heidrich-Meisner and Christian Igel. Hoeffding and Bernstein races for selecting policies in evolutionary direct policy search. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 401–408. ACM, 2009. doi:10.1145/1553374.1553426.

Holger Heitsch and Werner Römisch. Scenario tree reduction for multistage stochastic programs. *Computational Management Science*, 6(2):117, 2009. doi:10.1007/s10287-008-0087-y.

Holger Heitsch, Hernan Leövey, and Werner Römisch. Are Quasi-Monte Carlo algorithms efficient for two-stage stochastic programs? *Computational Optimization and Applications*, 65(3):567–603, 2016. doi:10.1007/s10589-016-9843-z.

Yale T Herer, Michal Tzur, and Enver Yücesan. The multilocation transshipment problem. *IIE Transactions*, 38(3):185–200, 2006. doi:10.1080/07408170500434539.

Raymond M Hicks and Preston A Henne. Wing design by numerical optimization. *Journal of Aircraft*, 15(7):407–412, 1978. doi:10.2514/3.58379.

Tito Homem-de Mello and Güzin Bayraksan. Monte Carlo sampling-based methods for stochastic optimization. *Surveys in Operations Research and Management Science*, 19(1):56–85, 2014. doi:10.1016/j.sorms.2014.05.001.

Kjetil Høyland, Michal Kaut, and Stein W Wallace. A heuristic for moment-matching scenario generation. *Computational Optimization and Applications*, 24 (2):169–185, 2003. doi:10.1023/A:1021853807313.

Beiqing Huang and Xiaoping Du. A robust design method using variable transformation and Gauss-Hermite integration. *International Journal for Numerical Methods in Engineering*, 66(12):1841–1858, 2006. ISSN 1097-0207. doi:10.1002/nme.1577.

L Jeff Hong. Fully sequential indifference-zone selection procedures with variance-dependent sampling. *Naval Research Logistics (NRL)*, 53(5):464–476, 2006. doi:10.1002/nav.20155.

Daniel R Jiang, Lina Al-Kanj, and Warren B Powell. Monte Carlo tree search with sampled information relaxation dual bounds. *arXiv preprint arXiv:1704.05963*, 2017.

Yaochu Jin and Jürgen Branke. Evolutionary optimization in uncertain environments-a survey. *IEEE Transactions on Evolutionary Computation*, 9(3): 303–317, 2005. ISSN 1089-778X. doi:10.1109/TEVC.2005.846356.

Henrik Joensson, Kurt Jörnsten, and Edward A Silver. Application of the scenario aggregation approach to a two-stage, stochastic, common component, inventory problem with a budget constraint. *European Journal of Operational Research*, 68 (2):196–211, 1993. doi:10.1016/0377-2217(93)90303-5.

Donald R Jones, Matthias Schonlau, and William J Welch. Efficient global optimization of expensive black-box functions. *Journal of Global optimization*, 13(4): 455–492, 1998.

Kenneth I Kaitin and Catherine Cairns. The new drug approvals of 1999, 2000, and 2001: drug development trends a decade after passage of the Prescription Drug User Fee Act of 1992. *Drug Information Journal*, 37(4):357–371, 2003. doi:10.1177/009286150303700403.

Seong-Hee Kim and Barry L Nelson. A fully sequential procedure for indifference-zone selection in simulation. *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, 11(3):251–273, 2001. doi:10.1145/502109.502111.

Anton J Kleywegt, Alexander Shapiro, and Tito Homem-de Mello. The sample average approximation method for stochastic discrete optimization. *SIAM Journal on Optimization*, 12(2):479–502, 2002. doi:10.1137/s1052623499363220.

Panos Kouvelis, Joseph Milner, and Zhili Tian. Clinical trials for new drug development: Optimal investment and application. *Manufacturing & Service Operations Management*, 19(3):437–452, 2017. doi:10.1287/msom.2017.0616.

Johannes Kruisselbrink, Michael Emmerich, and Thomas Bäck. An archive maintenance scheme for finding robust solutions. In *International Conference on Parallel Problem Solving from Nature*, pages 214–223. Springer, 2010. doi:10.1007/978-3-642-15844-5_22.

Volodymyr Kuleshov and Doina Precup. Algorithms for multi-armed bandit problems. *arXiv preprint arXiv:1402.6028*, 2014.

Tze Leung Lai and Herbert Robbins. Asymptotically efficient adaptive allocation rules. *Advances in Applied Mathematics*, 6(1):4–22, 1985. doi:10.1016/0196-8858(85)90002-8.

Averill M Law, W David Kelton, and W David Kelton. *Simulation modeling and analysis*, volume 2. McGraw-Hill New York, 1991. doi:10.2307/1269971.

Loo Hay Lee, Ek Peng Chew, and Puvaneswari Manikam. A general framework on the simulation-based optimization under fixed computing budget. *European Journal of Operational Research*, 174(3):1828–1841, 2006. doi:10.1016/j.ejor.2005.02.052.

S. H. Lee, W. Chen, and B. M. Kwak. Robust design with arbitrary distributions using Gauss-type quadrature formula. *Structural and Multidisciplinary Optimization*, 39(3):227–243, 2009. ISSN 1615-1488. doi:10.1007/s00158-008-0328-2.

Hernan Leövey and Werner Römisch. Quasi-Monte Carlo methods for linear two-stage stochastic programming problems. *Mathematical Programming*, 151(1):315–345, 2015. doi:10.1007/s10107-015-0898-x.

Lihong Li, Wei Chu, John Langford, and Robert E Schapire. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th international conference on World wide web*, pages 661–670. ACM, 2010. doi:10.1145/1772690.1772758.

Jeff Linderoth and Stephen Wright. Decomposition algorithms for stochastic programming on a computational grid. *Computational Optimization and Applications*, 24(2-3):207–250, 2003.

Jeff Linderoth, Alexander Shapiro, and Stephen Wright. The empirical behavior of sampling methods for stochastic programming. *Annals of Operations Research*, 142(1):215–241, 2006. doi:10.1007/s10479-006-6169-8.

Changzheng Liu, Yueyue Fan, and Fernando Ordóñez. A two-stage stochastic programming model for transportation network protection. *Computers & Operations Research*, 36(5):1582–1590, 2009. doi:10.1016/j.cor.2008.03.001.

François V Louveaux and Yves Smeers. Optimal investments for electricity generation: A stochastic model and a test problem. *Numerical Techniques for Stochastic Optimization*, pages 33–64, 1988. doi:10.1007/978-3-642-61370-8_24.

Wai-Kei Mak, David P Morton, and R Kevin Wood. Monte Carlo bounding techniques for determining solution quality in stochastic programs. *Operations Research Letters*, 24(1):47–56, 1999. doi:10.1016/s0167-6377(98)00054-6.

Oden Maron and Andrew W Moore. The racing algorithm: Model selection for lazy learners. *Artificial Intelligence Review*, 11(1-5):193–225, 1997. doi:10.1007/978-94-017-2053-3_8.

Naomi Miller and Andrzej Ruszczyński. Risk-averse two-stage stochastic linear programming: Modeling and decomposition. *Operations Research*, 59(1):125–132, 2011. doi:10.1287/opre.1100.0847.

Peyman Mohajerin Esfahani and Daniel Kuhn. Data-driven distributionally robust optimization using the Wasserstein metric: performance guarantees and tractable reformulations. *Mathematical Programming*, 2017. doi:10.1007/s10107-017-1172-1.

Bernard Munos. Lessons from 60 years of pharmaceutical innovation. *Nature Reviews Drug Discovery*, 8(12):959, 2009. doi:10.1038/nrd2961.

Barry L Nelson, Julie Swann, David Goldsman, and Wheyming Song. Simple procedures for selecting the best simulated system when the number of alternatives is large. *Operations Research*, 49(6):950–963, 2001. doi:10.1287/opre.49.6.950.10019.

Yew-Soon Ong, Prasanth B Nair, and Kai Yew Lum. Max-min surrogate-assisted evolutionary algorithm for robust design. *IEEE Transactions on Evolutionary Computation*, 10(4):392–404, 2006. ISSN 1089-778X. doi:10.1109/TEVC.2005.859464.

Ingo Paenke, Jürgen Branke, and Yaochu Jin. Efficient search for robust solutions by means of evolutionary algorithms and fitness approximation. *IEEE Transactions on Evolutionary Computation*, 10(4):405–420, 2006. ISSN 1089-778X. doi:10.1109/TEVC.2005.859465.

Christos H Papadimitriou and John N Tsitsiklis. The complexity of optimal queuing network control. *Mathematics of Operations Research*, 24(2):293–305, 1999. doi:10.1287/moor.24.2.293.

Panos Parpas, Berk Ustun, Mort Webster, and Quang Kha Tran. Importance sampling in stochastic programming: A Markov chain Monte Carlo approach. *INFORMS Journal on Computing*, 27(2):358–377, 2015. doi:10.1287/ijoc.2014.0630.

Steven M Paul, Daniel S Mytelka, Christopher T Dunwiddie, Charles C Persinger, Bernard H Munos, Stacy R Lindborg, and Aaron L Schacht. How to improve R&D productivity: the pharmaceutical industry's grand challenge. *Nature reviews Drug discovery*, 9(3):203, 2010. doi:10.1038/nrd3078.

Yijie Peng, Chun-Hung Chen, Michael C Fu, and Jian-Qiang Hu. Dynamic sampling allocation and design selection. *INFORMS Journal on Computing*, 28(2):195–208, 2016.

G Ch Pflug. Scenario tree generation for multiperiod financial optimization by optimal discretization. *Mathematical Programming*, 89(2):251–271, 2001. doi:10.1007/pl00011398.

Warren B Powell and Ilya O Ryzhov. *Optimal learning*, volume 841. John Wiley & Sons, 2012. doi:10.1002/9781118309858.

Ragheb Rahmaniani, Teodor Gabriel Crainic, Michel Gendreau, and Walter Rei. The Benders decomposition algorithm: A literature review. *European Journal of Operational Research*, 259(3):801–817, 2017. doi:10.1016/j.ejor.2016.12.005.

Magnus Rattray and Jon Shapiro. Noisy fitness evaluation in genetic algorithms and the dynamics of learning. In Richard Belew and Michael Vose, editors, *Foundations of Genetic Algorithms: 4th Workshop*, volume 4, pages 117–139. Morgan Kaufmann, San Francisco, 1998.

Andrzej Ruszczyński and Artur Świetanowski. Accelerating the regularized decomposition method for two stage stochastic linear problems. *European Journal of Operational Research*, 101(2):328–342, 1997. doi:10.1016/s0377-2217(96)00401-8.

Ilya O Ryzhov and Warren B Powell. Information collection on a graph. *Operations Research*, 59(1):188–201, 2011. doi:10.1287/opre.1100.0873.

Ilya O. Ryzhov and Warren B. Powell. Information collection for linear programs with uncertain objective coefficients. *SIAM Journal on Optimization*, 22(4):1344–1368, 2012. doi:10.1137/12086279x.

Amit Saha, Tapabrata Ray, and Smith. Towards practical evolutionary robust multiobjective optimization. In *2011 IEEE Congress of Evolutionary Computation*, pages 2123–2130, 2011. doi:10.1109/CEC.2011.5949877.

Shaul Salomon, Gideon Avigad, Robin C Purshouse, and Peter J Fleming. Gearbox design for uncertain load requirements using active robust optimization. *Engineering Optimization*, 48(4):652–671, 2016.

Tjendera Santoso, Shabbir Ahmed, Marc Goetschalckx, and Alexander Shapiro. A stochastic programming approach for supply chain network design under uncertainty. *European Journal of Operational Research*, 167(1):96–115, 2005. doi:10.1016/j.ejor.2004.01.046.

Craig W Schmidt and Ignacio E Grossmann. Optimization models for the scheduling of testing tasks in new product development. *Industrial & Engineering Chemistry Research*, 35(10):3498–3510, 1996. doi:10.1021/ie9601099.

Alexander Schuhmacher, Oliver Gassmann, and Markus Hinder. Changing R&D models in research-based pharmaceutical companies. *Journal of translational medicine*, 14(1):105, 2016. doi:10.1186/s12967-016-0838-4.

Warren Scott, Peter Frazier, and Warren Powell. The correlated knowledge gradient for simulation optimization of continuous parameters using gaussian process regression. *SIAM Journal on Optimization*, 21(3):996–1026, 2011. doi:10.1137/100801275.

Nicola Secomandi. A rollout policy for the vehicle routing problem with stochastic demands. *Operations Research*, 49(5):796–802, 2001. doi:10.1287/opre.49.5.796.10608.

Nicola Secomandi. An analysis of the control-algorithm re-solving issue in inventory and revenue management. *Manufacturing & Service Operations Management*, 10 (3):468–483, 2008. doi:10.1287/msom.1070.0184.

Suvrajeet Sen, Robert D Doverspike, and Steve Cosares. Network planning with random demand. *Telecommunication Systems*, 3(1):11–30, 1994. doi:10.1007/bf02110042.

Suresh Sethi and Gerhard Sorger. A theory of rolling horizon decision making. *Annals of Operations Research*, 29(1):387–415, 1991. doi:10.1007/bf02283607.

Alexander Shapiro and Tito Homem-de Mello. A simulation-based approach to two-stage stochastic programming with recourse. *Mathematical Programming*, 81(3): 301–325, 1998. doi:10.1007/bf01580086.

Alexander Shapiro, Tito Homem-de Mello, and Joocheol Kim. Conditioning of convex piecewise linear stochastic programs. *Mathematical Programming*, 94(1): 1–19, 2002. doi:10.1007/s10107-002-0313-2.

Alexander Shapiro, Darinka Dentcheva, and Andrzej Ruszczyński. *Lectures on stochastic programming: modeling and theory*. SIAM, 2009. doi:10.1137/1.9780898718751.

Leyuan Shi and Sigurdur Ólafsson. Nested partitions method for global optimization. *Operations research*, 48(3):390–407, 2000. doi:10.1287/opre.48.3.390.12436.

David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *Nature*, 550(7676): 354, 2017. doi:10.1038/nature24270.

Peter Stagge. Averaging efficiently in the presence of noise. In *International Conference on Parallel Problem Solving from Nature*, pages 188–197. Springer, 1998. doi:10.1007/bfb0056862.

Rebecca Stockbridge and Güzin Bayraksan. A probability metrics approach for reducing the bias of optimality gap estimators in two-stage stochastic linear programming. *Mathematical Programming*, 142(1-2):107–131, 2013. doi:10.1007/s10107-012-0563-6.

Csaba Szepesvári. Algorithms for reinforcement learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 4(1):1–103, 2010. doi:10.2200/s00268ed1v01y201005aim009.

Colin Terry and Neil Lesser. A new future for R&D? Measuring the return from pharmaceutical innovation 2017. Technical report, The Deloitte Centre for Health Solutions, 2018.

Peter F Thall. Bayesian adaptive dose-finding based on efficacy and toxicity. *J Statistical Research*, 14:187–202, 2012.

Shigeyoshi Tsutsui and Ashish Ghosh. Genetic algorithms with a robust solution searching scheme. *IEEE Transactions on Evolutionary Computation*, 1(3):201–208, 1997. ISSN 1089-778X. doi:10.1109/4235.661550.

Wim van Ackooij, Welington de Oliveira, and Yongjia Song. Adaptive partition-based level decomposition methods for solving two-stage stochastic programs with fixed recourse. *Informs Journal on Computing*, 30(1):57–70, 2017. doi:10.1287/ijoc.2017.0765.

Richard M Van Slyke and Roger Wets. L-shaped linear programs with applications to optimal control and stochastic programming. *SIAM Journal on Applied Mathematics*, 17(4):638–663, 1969. doi:10.1137/0117061.

Peter M Verderame, Josephine A Elia, Jie Li, and Christodoulos A Floudas. Planning and scheduling under uncertainty: a review across multiple sectors. *Industrial & engineering chemistry research*, 49(9):3993–4017, 2010. doi:10.1021/ie902009k.

C. Villani. *Optimal transport: Old and new*. Grundlehren der mathematischen Wissenschaften. Springer Berlin Heidelberg, 2008.

Sofía S Villar. Indexability and optimal index policies for a class of reinitialising restless bandits. *Probability in the Engineering and Informational sciences*, 30(1): 1–23, 2016.

Roger Wets. Programming under uncertainty: The complete problem. *Probability Theory and Related Fields*, 4(4):316–339, 1966. doi:10.1007/bf00539117.

Peter Whittle. Restless bandits: Activity allocation in a changing world. *Journal of Applied Probability*, 25(A):287–298, 1988. doi:10.1017/s0021900200040420.

Christian Wolf, Csaba I Fábián, Achim Koberstein, and Leena Suhl. Applying oracles of on-demand accuracy in two-stage stochastic programming–A computational study. *European Journal of Operational Research*, 239(2):437–448, 2014. doi:10.1016/j.ejor.2014.05.010.

Jie Xu, Si Zhang, Edward Huang, Chun-Hung Chen, Loo Hay Lee, and Nurcin Celik. Mo2tos: Multi-fidelity optimization with ordinal transformation and optimal sampling. *Asia-Pacific Journal of Operational Research*, 33(03):3940–3951, 2016. doi:10.1142/s0217595916500172.

Juan Camilo Zapata, Vishal A Varma, and Gintaras V Reklaitis. Impact of tactical and operational policies in the selection of a new product portfolio. *Computers & Chemical Engineering*, 32(1-2):307–319, 2008. doi:10.1016/j.compchemeng.2007.03.024.

Zhe Zhou, Jianyun Zhang, Pei Liu, Zheng Li, Michael C Georgiadis, and Efstratios N Pistikopoulos. A two-stage stochastic programming model for the optimal design of distributed energy systems. *Applied Energy*, 103:135–144, 2013. doi:10.1016/j.apenergy.2012.09.019.