Scenario Tree Reduction Methods Through Changing Node Values

Zhiping Chen*, Zhe Yan

Department of Computing Science, School of Mathematics and Statistics, Xi'an Jiaotong University, 710049, Xi'an, Shaanxi, P. R. China

Abstract To develop practical and efficient scenario tree reduction methods, we introduce a new methodology which allows for changing node values, and an easy-to-calculate distance function to measure the difference between two scenario trees. Based on minimizing the new distance, we first construct a primitive scenario tree reduction model which also minimizes the Wasserstein distance between the reduced tree and the initial tree. To make it appropriate for the reduction of general multi-stage scenario trees, we then construct an improved model which not only performs well for the multi-stage scenario tree reduction but also is supported theoretically by the stability results of stochastic programs. Two approximate algorithms are proposed to solve the primitive model. With them, we design a stage-wise scenario tree reduction algorithm which is superior to the simultaneous backward reduction method in terms of computational complexity, the corresponding reduction algorithm especially for fan-liked trees is also presented. We further design a multi-stage scenario tree reduction algorithm with a pre-specified distance by utilizing the stability results of stochastic programs. A series of numerical experiments with real trading data and the application to multi-stage portfolio selection demonstrate the practicality, efficiency and robustness of proposed reduction models and algorithms. **Keywords**: stochastic programs, scenario tree reduction, probabilistic metric, stability, computational efficiency, portfolio selection

1 Introduction

Stochastic programming is a very useful approach for coping with decision problems under uncertainty in areas such as production, energy, transportation and finance. Generally speaking, stochastic parameters in these decision problems are expressed via continuous probability distributions or stochastic processes, and the original forms of these decision problems are infinite-dimensional optimization problems. To make these problems tractable, one usually resorts to the scenario tree technique [1] to transform the original optimization problem into a finite-dimensional programming problem. In

*Corresponding author. Tel.: +86 29 8266 3741; fax: +86 29 8266 3938.
Email address: zchen@mail.xjtu.edu.cn (Zhiping Chen).

order to approximate the underlying random distribution as precisely as possible, the size of the scenario tree tends to be very large, and the deterministic transformation of the original decision problem is large-scale and hard to solve. Due to this, the scenario tree reduction method is proposed in [2], which can be used to decrease the dimension of the decision problem. How to appropriately reduce a scenario tree has become an active research area since then.

Scenario reduction methods aim at reducing the scenario number of a scenario tree. For example, the k-means clustering method is used in [3] to partition a given set of scenarios into a number of clusters. As a result of this partition, scenarios with similar features are assigned to the same cluster. Each cluster is replaced by a scenario, and the scenario number is reduced. However, when a scenario tree is reduced, the corresponding programming problem based on the tree is changed and so the corresponding optimal value. A good scenario reduction method should ensure that the reduced tree's scale is small enough while keeping the resulting optimal value close to the original optimal value as much as possible. For this reason, most scenario tree reduction algorithms are designed by utilizing the quantitative stability results of stochastic programs.

Based on an upper bound of the Fortet-Mourier probability metrics from the stability analysis of convex stochastic programs, backward reduction and forward selection methods are designed in [2] to reduce a scenario tree. These two methods are further developed to the simultaneous backward reduction method and the fast forward selection method in [4] in order to improve the computational performance. To enhance the accuracy of the reduction process, the preceding reduction methods are improved in [5] by relying directly on Fortet-Mourier metrics instead of their upper bounds. Some heuristics for forward and backward scenario tree reduction algorithms are developed in [6] by applying the new stability results in [7]. Scenario tree reduction methods have been extended to chance constrained and mix-integer two-stage stochastic programming models in [8] and [9], respectively. The sophisticated stability results for multistage stochastic programs in [7] tell us that the so-called filtration distance should be considered during the multistage scenario tree reduction process. Based on the new stability result, a single node reduction method is designed in [10], taking both the probability distribution distance and the filtration distance into account. All the above scenario tree reduction methods rely heavily on the solution of facility location problems which are NP-hard. A random search procedure that can quickly solve facility location problems is proposed in [11], and thus a more efficient scenario tree reduction method can be designed. Recently, the notions of the nested distribution and the nested distance are proposed for stochastic programs in [12, 13], with which the filtration distance can be avoided, these notions have been used for designing scenario tree generation methods [14] and can also be used as a potential way for designing new scenario tree reduction methods [15].

All the existing scenario tree reduction methods utilizing the quantitative stability results of stochastic programs transform the original tree into a small-scale tree by minimizing the distance between them. The Wasserstein distance is the one mostly used. It is proved in [16] and [17] that the difference between the optimal values of stochastic programs under different probability measures are bounded from above by the Wasserstein distance between the corresponding measures. The Wasserstein distance between the probability measure P and the probability measure $\widetilde{P}$ can be

formulated as

$$W_r(P,\widetilde{P}) := \inf\Bigg\{ \Big(\iint_{\Omega\times\widetilde{\Omega}} d(\omega,\tilde{\omega})^r \eta(d(\omega,\tilde{\omega}))\Big)^{1/r} : \eta \text{ is a probability measure on } \Omega\times\widetilde{\Omega} \text{ with marginal distributions } \Omega \text{ and } \widetilde{\Omega} \Bigg\}.$$

Here, $r \geq 1$, Ω and $\widetilde{\Omega}$ are sample spaces and $d(\omega,\tilde{\omega})$ is the distance between ω and $\tilde{\omega}$.

Most of the existing scenario tree reduction methods directly delete some scenarios from the initial tree to obtain a small-scale scenario tree. We illustrate this by an example. Consider a single-stage scenario tree which has two scenarios, 1.1 and 0.9, with the probabilities being 0.4 and 0.6, respectively. The simultaneous backward reduction method in [4] is used here to reduce the scenario tree to a tree with a single scenario. Suppose that the cost function ([2,4]) is the Euclidean distance. Then the Kantorovich distance ([2,4]) between two probability measures is equal to the square of the Wasserstein distance between them. By using the method in [4] under this setting, the scenario 1.1 should be deleted, and the reduced scenario tree is 0.9 with the probability one. Then the Wasserstein distance with order 2 between the original scenario tree and the reduced one is 0.126 and the Kantorovich distance is 0.016.

Can the above distance be further decreased if another reduction strategy is applied? The answer is yes. Consider a scenario tree which has only one scenario 1.0 with the probability one. The Wasserstein distance between this scenario tree and the above initial tree is 0.1 and the Kantorovich distance is 0.01. This example tells us that, if we not only reduce the scenario number but also change the values of the retained scenarios, the Wasserstein distance between the original tree and the reduced one can be further decreased. It means that the scenario tree reduction method in [4] can be improved. Instead of simply deleting some scenarios from the original scenario tree, we try to find a tree with the same scale as the reduced tree gotten with the method in [4] but perfectly minimizes the Wasserstein distance between it and the original one.

The Wasserstein distance corresponds to the optimal value of an optimization problem. If we want to find a fixed-scale scenario tree whose distance to the original tree is the smallest, we will come across a nested optimization problem which is hard to solve. So it is not easy to utilize the Wasserstein distance directly. To overcome this and to significantly improve the implementability and computational efficiency of current scenario tree reduction algorithms, we introduce a new methodology. The new technique accomplishes the tree reduction in two steps: in the first step, we keep the structure of the original scenario tree and just change node values; in the second step, we reduce the scenario tree by merging those nodes with the same value. In this way, we can adopt a new distance function with a rather simple structure to measure the difference between two scenario trees, derived from which are new scenario tree reduction methods easy to implement and computational efficient. By minimizing the introduced distance, a new scenario reduction model is constructed. It is proved that the reduced tree obtained by our model is the one which is closest to the original tree among the trees with the same scale in terms of the Wasserstein distance. The new model performs better than the scenario reduction model in [2]. Two methods are designed to solve the model: the recursive-type method and the r-cluster method. We demonstrate that both methods

consume less time than the simultaneous backward reduction method in [4], which is used to solve the scenario reduction model in [2].

However, the above model is not suitable for the reduction of general multi-stage scenario trees, because it doesn't consider the correlation among scenarios. So we propose an improved model which not only performs well for the reduction of general multi-stage scenario trees, but is supported theoretically by the stability result in [14]. By combining the recursive-type method and the r-cluster method, a stage-wise hybrid algorithm is designed to solve the improved model. The resulting algorithms can generate a reduced scenario tree with the pre-specified structure. Moreover, to find a smaller scenario tree which is the closest one to the original tree in terms of distance, we further design a multi-stage scenario tree reduction algorithm with a pre-specified distance by utilizing the stability result in [14]. Finally, all the proposed algorithms are tested through numerical experiments to demonstrate their efficiency. And the reduction algorithms are applied to solve a multi-stage portfolio selection problem. The results show that it is necessary and significant to use the reduction methods when we want to solve stochastic programming problems based on the scenario tree.

The rest of this paper is organized as follows. In Section 2, the new distance with a simple structure is proposed. Based on it, we construct a primitive model which is mainly for the reduction of single-stage scenario trees and an improved model for the reduction of general multi-stage scenario trees, respectively. The advantages of the two models are also examined. In Section 3, two methods for solving the primitive model are designed and an approximate stage-wise scenario tree reduction algorithm is then designed. We also analyse the features of the proposed algorithms. Furthermore, a multi-stage scenario tree reduction method with a pre-specified distance is designed in Section 4. Section 5 presents numerical results, and Section 6 concludes the paper.

2 Scenario tree reduction models

In this section, two scenario tree reduction models are designed. The first model which we call the primitive reduction model is significant but not appropriate for the general multi-stage scenario tree reduction. An improved reduction model is thus proposed which can handle the multi-stage case effectively. We first present the primitive reduction model.

2.1 The primitive reduction model

The T-stage ($T \geq 1$) scenario tree can be denoted by $\mathbb{P} = (\Omega, \{\mathscr{F}_t\}_{t=0}^T, P)$, where $\Omega \subset \mathbb{R}^d$ is the sample set and P is the probability measure on Ω. $\{\mathscr{F}_t\}_{t=0}^T$ is the filtration with $\mathscr{F}_0 = \{\Omega, \emptyset\}$, $\mathscr{F}_T = \sigma(\Omega)$ and $\mathscr{F}_0 \subset \mathscr{F}_1 \subset \mathscr{F}_2 \cdots \subset \mathscr{F}_T$. Let Ω_t denote the node set of the scenario tree at stage t and N_t be the corresponding node number, $t = 0, 1, 2, \cdots, T$. It is natural that $|\Omega_t| = N_t$ and $|\Omega_0| = N_0 = 1$. Here, $|\cdot|$ stands for the element number of a set. The ith node at stage t is denoted by $\omega_{t,i} \in \mathbb{R}^d$, then $\Omega_t = \{\omega_{t,1}, \omega_{t,2}, \cdots, \omega_{t,N_t}\}$, $t = 1, 2, \cdots, T$. Let $\Omega_0 = \{\omega_0\}$, where ω_0 denotes the root node. The scenario number of the multi-stage scenario tree is $N = N_T$ and the scenarios will be denoted by $\omega_1, \omega_2, \cdots, \omega_N$ with the probabilities being $p_1, p_2, \cdots, p_N$, respectively. The index of the father node at stage t, $t = 1, 2, \cdots, T-1$, of the leaf

node i is denoted by $\phi_t(i)$, then we have $\omega_i = (\omega_0, \omega_{1,\phi_1(i)}, \omega_{2,\phi_2(i)}, \cdots, \omega_{T-1,\phi_{T-1}(i)}, \omega_{T,i})$, $i = 1, 2, \cdots, N$.

The scenario tree can also briefly be denoted by $\mathbb{P} = \sum_{i=1}^{N} p_i \delta_{\omega_i}$. To reduce the scale of the tree, some scenarios of $\mathbb{P}$ should be deleted. Here, to derive easily implementable algorithms, we don't simply delete scenarios as usually do in the literature. Instead, we change the values of some scenarios in order to make their values be the same. The new scenarios are denoted by $y_1, y_2, \cdots, y_N$ with the probabilities still being $p_1, p_2, \cdots, p_N$, respectively, but some of $y_1, y_2, \cdots, y_N$ have the same value. For example, y_1, y_3, and y_4 may have the same value, and y_2 and y_6 may have another same value, etc. The resulting scenario tree is denoted by $\mathbb{Q}$, which is different from $\mathbb{P}$ only due to possibly different scenario values. Therefore, a simple distance between $\mathbb{P}$ and $\mathbb{Q}$ can be defined as

$$d_r(\mathbb{P}, \mathbb{Q}) = \{\sum_{i=1}^{N} p_i \parallel \omega_i - y_i \parallel^r\}^{1/r}. \tag{1}$$

Here, $r \geq 1$ is an integer and $\parallel \omega_i - y_i \parallel^r = \sum_{t=0}^{T} \parallel \omega_{t,i} - y_{t,i} \parallel^r$. This distance is called the d_r-distance in what follows.

For the scenario tree $\mathbb{Q}$, we can treat each group of those scenarios with the same value as a single new scenario, whose occurring probability is the sum of the probabilities of those included scenarios. In this way, we can derive a new scenario tree $\widetilde{\mathbb{P}}$ whose scenarios are denoted by $\tilde{\omega}_1, \tilde{\omega}_2, \cdots, \tilde{\omega}_{\widetilde{N}}$ with the probabilities being $\tilde{p}_1, \tilde{p}_2, \cdots, \tilde{p}_{\widetilde{N}}$. For example, y_1, y_3, and y_4 have the same value $\tilde{\omega}_1$ and $\tilde{p}_1$ is then the sum of p_1, p_3, and p_4, while y_2 and y_6 have the same value $\tilde{\omega}_5$ and $\tilde{p}_5$ is the sum of p_2 and p_6. Considering the relationship between $\widetilde{\mathbb{P}}$ and $\mathbb{Q}$, the distance between the original tree $\mathbb{P}$ and the reduced tree $\widetilde{\mathbb{P}}$ is defined as

$$D_r(\mathbb{P}, \widetilde{\mathbb{P}}) = d_r(\mathbb{P}, \mathbb{Q}). \tag{2}$$

According to this definition, to determine the optimal reduced tree $\widetilde{\mathbb{P}}$, we just need to find an optimal $\mathbb{Q}$ which minimizes $d_r(\mathbb{P}, \mathbb{Q})$. Since the reduced scenario tree $\widetilde{\mathbb{P}}$ has $\widetilde{N}$ scenarios, finding the optimal $\mathbb{Q}$ is equivalent to finding an optimal way to partition the original N scenarios into $\widetilde{N}$ groups and determining the optimal scenario value in each group simultaneously, this inspires us to establish the following model

$$\begin{aligned}
&\min_{\widetilde{I}_m, \tilde{\omega}_m} \; d_r(\mathbb{P}, \mathbb{Q}) = \{\sum_{i=1}^{N} p_i \parallel \omega_i - y_i \parallel^r\}^{1/r} \\
&\text{s.t.} \; \bigcup_{m=1}^{\widetilde{N}} \widetilde{I}_m = \{1, 2, \cdots, N\}, \\
&\quad \widetilde{I}_{m_1} \cap \widetilde{I}_{m_2} = \emptyset, \; \forall\, m_1 \neq m_2, \; m_1, m_2 = 1, 2, \cdots, \widetilde{N}, \\
&\quad y_i = \tilde{\omega}_m, \; \text{for}\, i \in \widetilde{I}_m, \; m = 1, 2, \cdots, \widetilde{N}.
\end{aligned} \tag{3}$$

Once we solve the problem (3), it will determine the optimal partition $\widetilde{I}_m$, $m = 1, 2, \cdots, \widetilde{N}$, and the corresponding scenario value $\tilde{\omega}_m$ for each subset $\widetilde{I}_m$. This specifies the scenario tree $\mathbb{Q}$ and the corresponding reduced tree $\widetilde{\mathbb{P}}$ in return.

The Wasserstein distance is often adopted when one reduces a scenario tree based on the quantitative stability results of stochastic programs. An interesting question is: Is the $\widetilde{\mathbb{P}}$ determined above also the one which minimizes the W_r-distance $W_r(\mathbb{P}, \widetilde{\mathbb{P}})$? To illustrate this, we need to introduce some basic conceptions about the quantization for probability distributions ([18]).

Let X denote a $\mathbb{R}^{(T+1)d}$-valued random variable with distribution $\mathbb{P}$. For $n \in \mathbb{N}$, let $\mathcal{F}_n$ be the set of all Borel measurable maps $f : \mathbb{R}^{(T+1)d} \to \mathbb{R}^{(T+1)d}$ with $|f(\mathbb{R}^{(T+1)d})| \leq n$. The elements of $\mathcal{F}_n$ are called n-quantizers. For each $f \in \mathcal{F}_n$, $f(X)$ gives a quantized version of X. Let $1 \leq r < \infty$ and assume $E \parallel X \parallel^r < \infty$. The n-th quantization error for $\mathbb{P}$ of order r is defined by

$$V_{n,r}(\mathbb{P}) = \inf_{f \in \mathcal{F}_n} E \parallel X - f(X) \parallel^r . \tag{4}$$

A quantizer $f \in \mathcal{F}_n$ is called n-optimal for $\mathbb{P}$ of order r if $V_{n,r}(\mathbb{P}) = E \parallel X - f(X) \parallel^r$. The following two lemmas are established in [18].

Lemma 1.

$$V_{n,r}(\mathbb{P}) = \inf_{\alpha \subset \mathbb{R}^{(T+1)d}, |\alpha| \leq n} E \min_{a \in \alpha} \parallel X - a \parallel^r .$$

It is obvious that if α is the optimal solution, then $|\alpha| = n$.

Lemma 2. *For the Wasserstein distance, we have*

$$V_{n,r}(\mathbb{P}) = \inf_{f \in \mathcal{F}_n} W_r^r(\mathbb{P}, \mathbb{P}^f) = \inf_{\widetilde{\mathbb{P}} \in \mathcal{P}_n} W_r^r(\mathbb{P}, \widetilde{\mathbb{P}})$$

if $d(\omega, \tilde{\omega}) = \parallel \omega - \tilde{\omega} \parallel$. *Here,* $\mathcal{P}_n$ *denotes the set of all discrete probability measures* $\mathbb{Q}$ *on* R^d *with* $|supp(\mathbb{Q})| \leq n$. $\mathbb{P}^f$ *denotes the image measure of* $\mathbb{P}$ *under* f.

For the probability measure $\mathbb{P} = \sum_{i=1}^N p_i \delta_{\omega_i}$, let $f(\omega_j) = \tilde{\omega}_m$ for $j \in \widetilde{I}_m$, where $\widetilde{I}_m$, $m = 1, 2, \cdots, n$, forms a partition of $\{1, 2, \cdots, N\}$, then the image measure $\mathbb{P}^f = \sum_{m=1}^n \tilde{p}_m \delta_{\tilde{\omega}_m}$, here $\tilde{p}_m = \sum_{j \in \widetilde{I}_m} p_j$ for $m = 1, 2, \cdots, n$. Actually, we have $\mathbb{P}^f = \mathbb{P} \circ f^{-1}$.

With the above preparations, we can derive the following result.

Theorem 1. *Let* $\mathbb{Q} = \sum_{i=1}^N p_i \delta_{y_i}$ *be the probability measure deduced from the optimal solution of the model* (3). *The probability measure* $\widetilde{\mathbb{P}}$ *obtained from* $\mathbb{Q}$ *is an optimal solution to* $\min_{\widetilde{\mathbb{P}} \in \mathcal{P}_{\widetilde{N}}} W_r(\mathbb{P}, \widetilde{\mathbb{P}})$ *where* $d(\omega, \tilde{\omega}) = \parallel \omega - \tilde{\omega} \parallel$. *Here* $\mathcal{P}_{\widetilde{N}}$ *denotes the set of all discrete probability measures* $\widetilde{\mathbb{P}}$ *with* $|supp(\widetilde{\mathbb{P}})| \leq \widetilde{N}$.

Proof. In the model (3), suppose that $\omega_i \in \mathbb{R}^{(T+1)d}$, $i = 1, 2, \cdots, N$, and let X be a random variable whose realizations are $\omega_1, \omega_2, \cdots, \omega_N$, with the probabilities being $p_1, p_2, \cdots, p_N$, respectively. Assume that $\tilde{\omega}_m$, $m = 1, 2, \cdots, \widetilde{N}$, are all determined, then in order to minimize $d_r(\mathbb{P}, \mathbb{Q})$, we must have

$$\widetilde{I}_m = \{j \in \{1, 2, \cdots, n\} | \parallel \omega_j - \tilde{\omega}_m \parallel^r = \min_{k \in \{1, 2, \cdots, \widetilde{N}\}} \parallel \omega_j - \tilde{\omega}_k \parallel^r\}.$$

The model (3) can thus be rewritten as

$$\inf_{\widetilde{I}_m,\tilde{\omega}_m} d_r^r(\mathbb{P},\mathbb{Q}) = \inf_{\{\tilde{\omega}_m, m=1,2,\cdots,\widetilde{N}\}\subset\mathbb{R}^d} \sum_{m=1,2,\cdots,\widetilde{N}} \sum_{j\in\widetilde{I}_m} p_j \parallel \omega_j - \tilde{\omega}_m \parallel^r . \tag{5}$$

Let $\alpha = \{\tilde{\omega}_m, m = 1, 2, \cdots, \widetilde{N}\}$. From the definition of $\widetilde{I}_m$, (5) can be transformed into

$$\inf_{\widetilde{I}_m,\tilde{\omega}_m} d_r^r(\mathbb{P},\mathbb{Q}) = \inf_{\alpha\subset\mathbb{R}^d,|\alpha|=\widetilde{N}} E \min_{\tilde{\omega}_m\in\alpha} \parallel X - \tilde{\omega}_m \parallel^r . \tag{6}$$

By Lemma 1, we have $\inf_{\widetilde{I}_m,\tilde{\omega}_m} d_r^r(\mathbb{P},\mathbb{Q}) = V_{\widetilde{N},r}(\mathbb{P})$.

Let $\tilde{\omega}_m$ and $\widetilde{I}_m$, $m = 1, 2, \cdots, \widetilde{N}$, be the optimal solution to the problem (3), and $\mathbb{Q}$ be the resulting probability measure. We define $f(\omega_j) = \tilde{\omega}_m$ for $j \in \widetilde{I}_m$, $m = 1, 2, \cdots, \widetilde{N}$. Then $\mathbb{P}^f$ is the reduced measure compared to $\mathbb{P}$. It is easy to show that

$$V_{\widetilde{N},r}(\mathbb{P}) = E \parallel X - f(X) \parallel^r .$$

This means that f is an n-optimal quantizer, and thus $\mathbb{P}^f \in \mathcal{P}_{\widetilde{N}}$ is an n-optimal quantizing measure for $\mathbb{P}$ of order r, so $V_{\widetilde{N},r}(\mathbb{P}) = W_r^r(\mathbb{P},\mathbb{P}^f)$ ([18]). By Lemma 2, we have

$$W_r^r(\mathbb{P},\mathbb{P}^f) = \inf_{\widetilde{\mathbb{P}}\in\mathcal{P}_{\widetilde{N}}} W_r^r(\mathbb{P},\widetilde{\mathbb{P}}).$$

□ □

Theorem 1 means that if the scale of the reduced scenario tree is pre-specified, then among all the scenario trees with the same scale, the tree obtained through the model (3) is the best one since it minimizes the Wasserstein distance to the original scenario tree.

To state it more concretely, We compare the reduction model (3) with the scenario reduction model in [2]. We first introduce the scenario reduction model in [2] briefly.

Corresponding to the original scenario tree $\mathbb{P} = \sum_{i=1}^N p_i\delta_{\omega_i}$, the reduced scenario tree $\widetilde{\mathbb{P}}$ is obtained in [2] by minimizing the Kantorovich distance $\hat{u}_c(\mathbb{P},\widetilde{\mathbb{P}})$:

$$\hat{u}_c(\mathbb{P},\widetilde{\mathbb{P}}) := \inf\left\{ \iint_{\Omega\times\widetilde{\Omega}} c(\omega,\tilde{\omega})\eta(d(\omega,\tilde{\omega})) : \eta \text{ is a probability measure on } \Omega\times\widetilde{\Omega} \right.$$
$$\left. \text{with marginal distributions } \Omega \text{ and } \widetilde{\Omega} \right\}.$$

Here, $c(\omega,\tilde{\omega})$ is the cost function. Readers can refer to [2] for the detailed definition of the cost function.

The probability measure corresponding to the reduced scenario tree can be denoted as $\widetilde{\mathbb{P}} = \sum_{j\notin J} q_j\delta_{\omega_j}$, here $J \subset \{1, 2, \cdots, N\}$. It means that $\widetilde{\mathbb{P}}$ is obtained by deleting all scenarios ω_j of $\mathbb{P}$, $j \in J$, and by assigning a new probabilistic weight q_j to each scenario ω_j, $j \notin J$. Let $D(J;q) := \hat{u}_c(\mathbb{P},\widetilde{\mathbb{P}})$. Then the reduced measure $\widetilde{\mathbb{P}}$ can be obtained by minimizing $D(J;q)$.

A method to minimize $D(J;q)$ was proposed in [2], which can be summarized as the following lemma.

Lemma 3. *Given $J \subset \{1,2,\cdots,N\}$, we have*

$$D_J := \min_q \{D(J;q) : \ q_j \geq 0, \sum_{j\notin J} q_j = 1\} = \sum_{i\in J} p_i \min_{j\notin J} c(\omega_i,\omega_j).$$

Moreover, the minimum is attained at $\overline{q}_j = p_j + \sum_{i\in J_j} p_i$, for each $j \notin J$, where $J_j = \{i \in J | j = j(i)\}$ and $j(i) \in arg\min_{j\notin J} c(\omega_i,\omega_j)$ for $i \in J$.

If J is the set which minimizes D_J, then the corresponding $\widetilde{\mathbb{P}}$ is the reduced measure which minimizes $\hat{u}_c(\mathbb{P},\widetilde{\mathbb{P}})$. Thus the scenario tree reduction model constructed in [2] can be described as follows.

$$\min\{D_J = \sum_{i\in J} p_i \min_{j\notin J} c(\omega_i,\omega_j) : \ J \subset \{1,2,\cdots,N\}, |J| = N - \widetilde{N}\}. \tag{7}$$

If we set $c(\omega,\tilde{\omega}) = d(\omega,\tilde{\omega})^r$, then $\hat{u}_c(\mathbb{P},\mathbb{Q}) = W_r^r(\mathbb{P},\mathbb{Q})$. Furthermore, if $d(\omega,\tilde{\omega}) = \| \omega - \tilde{\omega} \|$, we have:

Theorem 2. *Assume that $\mathbb{P}$ is the original probability measure, $\widetilde{\mathbb{P}}_1$ and $\widetilde{\mathbb{P}}_2$ are the reduced measures obtained through the model (3) and the model (7), respectively. If $\widetilde{\mathbb{P}}_1$ and $\widetilde{\mathbb{P}}_2$ have the same number of scenarios, then $W_r^r(\mathbb{P},\widetilde{\mathbb{P}}_1) \leq W_r^r(\mathbb{P},\widetilde{\mathbb{P}}_2)$, i.e., $\hat{u}_c(\mathbb{P},\widetilde{\mathbb{P}}_1) \leq \hat{u}_c(\mathbb{P},\widetilde{\mathbb{P}}_2)$.*

Proof. It's obvious that $W_r^r(\mathbb{P},\widetilde{\mathbb{P}}_2) = D_r^r(\mathbb{P},\widetilde{\mathbb{P}}_1)$ and

$$D_r^r(\mathbb{P},\widetilde{\mathbb{P}}_1) = \min_{\widetilde{I}_m,\tilde{\omega}_m} d_r^r(\mathbb{P},\mathbb{Q}) = \inf_{\{\tilde{\omega}_m, m=1,2,\cdots,\widetilde{N}\}\subset\mathbb{R}^d} \sum_{m=1,2,\cdots,\widetilde{N}} \sum_{i\in\widetilde{I}_m} p_i \parallel \omega_i - \tilde{\omega}_m \parallel^r .$$

Moreover,

$$\begin{aligned}
D_J = \sum_{i\in J} p_i \min_{j\notin J} c(\omega_i,\omega_j) &= \sum_{j\notin J}\sum_{i\in J_j} p_i c(\omega_i,\omega_j) \\
&= \sum_{j\notin J}\sum_{i\in J_j\cup\{j\}} p_i c(\omega_i,\omega_j) \\
&= \sum_{\substack{j\in N\setminus J \\ |N\setminus J| = \widetilde{N}}}\sum_{i\in J_j\cup\{j\}} p_i c(\omega_i,\omega_j) \\
&= \sum_{\substack{j\in N\setminus J \\ |N\setminus J| = \widetilde{N}}}\sum_{i\in J_j\cup\{j\}} p_i \parallel \omega_i - \tilde{\omega}_j \parallel^r \\
&\geq \inf_{\{\tilde{\omega}_j, j\in N\setminus J\}\subset\mathbb{R}^d} \sum_{\substack{j\in N\setminus J \\ |N\setminus J| = \widetilde{N}}}\sum_{i\in J_j\cup\{j\}} p_i \parallel \omega_i - \tilde{\omega}_j \parallel^r \\
&= D_r^r(\mathbb{P},\widetilde{\mathbb{P}}_1).
\end{aligned}$$

Therefore, $\min_J D_J \geq D_r^r(\mathbb{P},\widetilde{\mathbb{P}}_1)$. It follows that $D_r^r(\mathbb{P},\widetilde{\mathbb{P}}_1) \leq \hat{u}_c(\mathbb{P},\widetilde{\mathbb{P}}_2)$, which means $W_r^r(\mathbb{P},\widetilde{\mathbb{P}}_1) \leq W_r^r(\mathbb{P},\widetilde{\mathbb{P}}_2)$. □ □

We see from Theorem 2 that if the size of the reduced scenario tree is fixed beforehand, then the reduced tree obtained through the model (3) is closer to the initial tree than that through the model (7). And if the distance between the initial tree and the reduced one is set beforehand, then the size of the reduced tree obtained through the model (3) is smaller than that through the model (7). Therefore, the model (3) performs better than the model (7).

A scenario tree reduction method can then be designed based on the model (3). Suppose that an T-stage scenario tree $\widetilde{\mathbb{P}}$ with $\widetilde{N}$ scenarios, the reduced scenario tree corresponding to $\mathbb{P}$, is obtained through the model (3). Although $\widetilde{\mathbb{P}}$ has $\widetilde{N}$ scenarios, we have no idea about its structure, i.e., the number of the son nodes of each non-leaf node. So we propose a method to determine the structure of $\widetilde{\mathbb{P}}$. Assume the scenarios of $\widetilde{\mathbb{P}}$ are $\tilde{\omega}_1, \tilde{\omega}_2, \cdots, \tilde{\omega}_{\widetilde{N}}$ with the probabilities being $\tilde{p}_1, \tilde{p}_2, \cdots, \tilde{p}_{\widetilde{N}}$, respectively. Here, $\tilde{\omega}_i = (\tilde{\omega}_{0,i}, \tilde{\omega}_{1,i}, \tilde{\omega}_{2,i}, \cdots, \tilde{\omega}_{T,i})$, $i = 1, 2, \cdots, \widetilde{N}$, $\tilde{\omega}_{t,i} \in \mathbb{R}^d$, $t = 0, 1, 2, \cdots, T$, and $\tilde{\omega}_{0,1} = \tilde{\omega}_{0,2} = \cdots = \tilde{\omega}_{0,\widetilde{N}}$ corresponds to the root node. If the original scenario tree $\mathbb{P}$ is not a fan-liked scenario tree, $\widetilde{\mathbb{P}}$ should not be fan-liked either. In this case, there must exist some of $\tilde{\omega}_{t,1}, \tilde{\omega}_{t,2}, \cdots, \tilde{\omega}_{t,\widetilde{N}}$, $t = 1, 2, \cdots, T-1$, which have the same value. It is apparent that, at stage 1, $\{\tilde{\omega}_{1,1}, \tilde{\omega}_{1,2}, \cdots, \tilde{\omega}_{1,\widetilde{N}}\}$ is the son node set of the root node. Separate the elements of this set into several parts, the elements in each of these parts have the same value, while the elements of different parts have different values. We can treat each part as a new node, and then all the son nodes of the elements of that part would be the son nodes of the corresponding new node. At stage 2, for the nodes with the same father node, we separate them into several parts of which the values of the elements are the same. View each part as a new node and then the son nodes of the elements of that part should be the son nodes of the corresponding new node. We do this kind of operation stage by stage, the structure of $\widetilde{\mathbb{P}}$ can finally be determined.

The concrete reduction process of the multi-stage scenario tree $\mathbb{P}$ is shown in the following algorithm.

Algorithm 1: Reduction of the T-stage scenario tree $\mathbb{P}$ based on the model (3)

Step 1 Solve the problem (3) to obtain $\widetilde{\mathbb{P}}$ whose scenarios are $\tilde{\omega}_1, \tilde{\omega}_2, \cdots, \tilde{\omega}_{\widetilde{N}}$ with occurring probabilities being $\tilde{p}_1, \tilde{p}_2, \cdots, \tilde{p}_{\widetilde{N}}$, here $\tilde{\omega}_i = (\tilde{\omega}_{0,i}, \tilde{\omega}_{1,i}, \tilde{\omega}_{2,i}, \cdots, \tilde{\omega}_{T,i})$,$i = 1, 2, \cdots, \widetilde{N}$. Then $\{\tilde{\omega}_{1,i} : i = 1, \cdots, \widetilde{N}\}$ is the son nodes of the root node.

Step 2 Separate $\{\tilde{\omega}_{1,i} : i = 1, \cdots, \widetilde{N}\}$ into several parts, the elements in each of these parts have the same value. Treat each part as a new node, and all the son nodes of the elements of the part is chosen as the son nodes of the corresponding new node. The probability of the new node is the sum of probabilities of the elements in the corresponding part. $t = 1$.

Step 3 Let $t = t + 1$. If $t \leq T$, go to Step 4. Otherwise, stop.

Step 4 At stage t, for the nodes with the same father node, separate them into several parts of which the values of the elements are the same. Treat each part as a new node, and all the son nodes of the elements of that part is chosen as the son nodes of the corresponding new node. The probability of the new node is the sum of probabilities of the elements in the corresponding part. Go to Step 3.

Algorithm 1 relies on the solution of model (3), which will be investigated in the next section. We know that the number of leaf nodes, which is equal to the scenario number of the reduced scenario tree $\widetilde{\mathbb{P}}$ obtained by Algorithm 1, is smaller than that of

the original scenario tree $\mathbb{P}$. But we have no idea about the number of non-leaf nodes. Sometimes the node number at stage t of $\widetilde{\mathbb{P}}$, $t < T$, is not smaller than that of $\mathbb{P}$ or the two numbers are near to each other. These cases will be illustrated in numerical experiments. However, the scale of a scenario tree depends not only on the scenario number but also on the node number at each stage. So Algorithm 1 can not efficiently reduce the scale of the multi-stage scenario tree under some cases. What's more important, the scenarios of many multi-stage scenario trees are often correlated with each other, except the fan-liked scenario tree, but the model (3) essentially treats the scenarios as independent paths. This could lead to bad results when we use Algorithm 1 to reduce the multi-stage scenario tree, which will be shown in Section 5. Therefore, the model (3) should be improved in order to better reduce multi-stage scenario trees. It is worth mentioning here that, based on the model (7), the simultaneous backward reduction algorithm in [4] can not efficiently reduce the scale of multi-stage scenario trees either, because it also simply focuses on the reduction of scenarios as a whole, not on the reduction of non-leaf nodes of a scenario tree.

2.2 An improved reduction model

As pointed out in the last section, Algorithm 1, based on the model (3), can not efficiently reduce general multi-stage scenario trees. We will introduce an improved model for the reduction of multi-stage scenario tree in this section.

Similar to that in the last section, in order to derive an easily implementable algorithm, we do not delete the nodes of $\mathbb{P}$ at the beginning and the occurring probability of each node does not change either. Instead, the values of some nodes at the same stage will be set to a common value so that they can be treated as a single node afterwards. After the node values have been modified, the new scenario is denoted by $y_i = (\omega_0, y_{1,\phi_1(i)}, y_{2,\phi_2(i)}, \cdots, y_{T-1,\phi_{T-1}(i)}, y_{T,i})$, $i = 1, 2, \cdots, N$. The new scenario tree is denoted by $\mathbb{Q} = (\Omega^y, \{\mathscr{F}_t\}_{t=0}^T, P)$ where $\Omega^y = \{y_{T,i} | i = 1, 2, \cdots, N\}$. Like that in (1), the distance between $\mathbb{P}$ and $\mathbb{Q}$ can be defined as

$$d_r(\mathbb{P}, \mathbb{Q}) = \{\sum_{i=1}^{N} p_i \parallel \omega_i - y_i \parallel^r\}^{1/r}. \tag{8}$$

Here, $r \geq 1$ is an integer and $\parallel \omega_i - y_i \parallel^r = \sum_{t=1}^{T-1} \parallel \omega_{t,\phi(i)} - y_{t,\phi(i)} \parallel^r + \parallel \omega_{T,i} - y_{T,i} \parallel^r$. Furthermore, the formula (8) can be transformed into

$$d_r(\mathbb{P}, \mathbb{Q}) = \{\sum_{t=1}^{T} \sum_{i=1}^{N_t} p_{t,i} \parallel \omega_{t,i} - y_{t,i} \parallel^r\}^{1/r}, \tag{9}$$

where $p_{t,i}$ is the occurring probability of the node $\omega_{t,i}$, $i = 1, 2, \cdots, N_t$, $t = 1, 2, \cdots, T$.

$\mathbb{Q}$ is a T-stage scenario tree of which some nodes at a specific stage have the same value. Treating each group of this kind of nodes as a single new node and setting the probability of the new node to the sum of the probabilities of the original nodes in that group, we can then obtain a reduced multi-stage scenario tree corresponding to $\mathbb{P}$. Denote the reduced tree by $\widetilde{\mathbb{P}} = (\widetilde{\Omega}, \{\widetilde{\mathscr{F}_t}\}_{t=0}^T, \widetilde{P})$, and the distance between $\widetilde{\mathbb{P}}$ and $\mathbb{P}$

is defined as

$$D_r(\mathbb{P}, \widetilde{\mathbb{P}}) = d_r(\mathbb{P}, \mathbb{Q}). \tag{10}$$

Denote the node number at stage t of $\widetilde{\mathbb{P}}$ by $\widetilde{N}_t$, $t = 1, 2, \cdots, T$. From the definitions of the tree $\widetilde{\mathbb{P}}$ and the distance between $\widetilde{\mathbb{P}}$ and $\mathbb{P}$, we can establish the following scenario tree reduction model.

$$\begin{aligned}
\min_{\widetilde{I}_{t,m}, \tilde{\omega}_{t,m}, \tilde{N}_t^j} \; & d_r(\mathbb{P}, \mathbb{Q}) = \left\{ \sum_{t=1}^{T} \sum_{i=1}^{N_t} p_{t,i} \parallel \omega_{t,i} - y_{t,i} \parallel^r \right\}^{1/r} \\
\text{s.t.} \quad & \bigcup_{m_j = m_{j-1}+1}^{m_{j-1}+\widetilde{N}_t^j} \widetilde{I}_{t,m_j} = I_t^j, \\
& \widetilde{I}_{t,m_j+k_1} \cap \widetilde{I}_{t,m_j+k_2} = \emptyset, \, \forall \, k_1 \neq k_2, \; k_1, k_2 = 1, 2, \cdots, \widetilde{N}_t^j, \\
& m_0 = 0, \; j = 1, 2, \cdots, \widetilde{N}_{t-1}, \\
& \sum_{j=1}^{\widetilde{N}_{t-1}} \widetilde{N}_t^j = \widetilde{N}_t, \\
& y_{t,k} = \tilde{\omega}_{t,m}, \; for \, k \in \widetilde{I}_{t,m}, \; m = 1, 2, \cdots, \widetilde{N}_t, \\
& t = 1, 2, \cdots, T.
\end{aligned} \tag{11}$$

Here, I_t^j is the index set of the son nodes of the jth node at stage $t-1$. The first constraint means that the son nodes of the jth node at stage $t-1$ are divided into $\widetilde{N}_t^j$ groups. Each group will be represented by a new single node $\tilde{\omega}_{t,m}$ and then the jth node at stage $t-1$ will have only $\widetilde{N}_t^j$ son nodes. For example, when $t = 1$, the root node has N_1 son nodes, $\widetilde{N}_0 = 1$ and the index set of its son nodes is $I_1^1 = \{1, 2, \cdots, N_1\}$. According to the first constraint, these son nodes will be divided into $\widetilde{N}_1^1 = \widetilde{N}_1$ groups, and the root node will just have $\widetilde{N}_1$ son nodes. For the jth new node at stage 1, there is an index set $\widetilde{I}_{1,j}$ which consists of the indexes of the nodes in the group represented by the jth node. The index set of the son nodes of the jth node should be $I_2^j = \{i | \omega_{2,i} \in \Omega_2, \, \phi_1(i) \in \widetilde{I}_{1,j}\}$. It means that all the son nodes of those nodes in the group will be the son nodes of the new node that represents the group. The first constraint also enforces that only the nodes at the same stage with the same father node can possibly be treated as a single new node. This is important for ensuring the Markovian property, thus the proper structure of a scenario tree.

To solve the problem (11) is to determine the optimal $\tilde{N}_t^j$, $j = 1, 2, \cdots, \widetilde{N}_{t-1}$, and $\widetilde{I}_{t,m}, \tilde{\omega}_{t,m}$, $m = 1, 2, \cdots, \widetilde{N}_t$, $t = 1, 2, \cdots, T$. They specify the scenario tree $\mathbb{Q}$ and the resulting reduced scenario tree $\widetilde{\mathbb{P}}$ in return. The probability of the new node $\tilde{\omega}_{t,m}$ is the sum of the probabilities of the original nodes in $\widetilde{I}_{t,m}$. By the generation process of $\widetilde{\mathbb{P}}$, it's apparent that $\widetilde{\mathscr{F}_0} \subset \widetilde{\mathscr{F}_1} \subset \widetilde{\mathscr{F}_2} \cdots \subset \widetilde{\mathscr{F}_T}$ which demonstrates that $\widetilde{\mathbb{P}}$ is a multi-stage scenario tree.

When $T = 1$, the model (11) is the same as the model (3). When $T > 1$, the model (11) can help us to find an optimal scenario tree with the pre-specified structure, i.e., the specified node number at each stage, while the model (3) can only find an optimal scenario tree with a pre-specified number of scenarios.

Suppose that $\tilde{N}_t^j$, $j = 1, 2, \cdots, \widetilde{N}_{t-1}$, and $\widetilde{I}_{t,m}, \tilde{\omega}_{t,m}$, $m = 1, 2, \cdots, \widetilde{N}_t$, $t = 1, 2, \cdots, T$,

form a feasible solution to the problem (11). Then we can denote the scenario set of $\widetilde{\mathbb{P}}$ by

$$\widetilde{\mathscr{X}} = \{(\tilde{\omega}_0, \tilde{\omega}_{1,\phi_1(i)}, \tilde{\omega}_{2,\phi_2(i)}, \cdots, \tilde{\omega}_{T-1,\phi_{T-1}(i)}, \tilde{\omega}_{T,i}) | i = 1, 2, \cdots, \widetilde{N},\ \widetilde{N} = \widetilde{N}_T\},$$

while the scenario set of $\mathbb{P}$ can be denoted by

$$\mathscr{X} = \{(\omega_0, \omega_{1,\phi_1(i)}, \omega_{2,\phi_2(i)}, \cdots, \omega_{T-1,\phi_{T-1}(i)}, \omega_{T,i}) | i = 1, 2, \cdots, N\}.$$

Define the mapping $f : \mathscr{X} \to \widetilde{\mathscr{X}}$ as

$$f(\omega_0, \omega_{1,\phi_1(i)}, \omega_{2,\phi_2(i)}, \cdots, \omega_{T-1,\phi_{T-1}(i)}, \omega_{T,i})$$

$$= (\tilde{\omega}_0, \tilde{\omega}_{1,\phi_1(i)}, \tilde{\omega}_{2,\phi_2(i)}, \cdots, \tilde{\omega}_{T-1,\phi_{T-1}(i)}, \tilde{\omega}_{T,i}),$$

for all $i \in \widetilde{I}_{T,k}$. Since $\phi_t(i) \in \widetilde{I}_{t,\phi_t(k)}$ for $t = 1, 2, \cdots, T-1$ by the model (11), f is a transport map such that $\widetilde{\mathbb{P}} = \mathbb{P} \circ f^{-1}$. Furthermore, f is a non-anticipative transport map. That is, the tth component of $f(\omega^1, \omega^2, \cdots, \omega^T)$, denoted by $f(\omega^1, \omega^2, \cdots, \omega^T)_t$, is fully determined by $(\omega^1, \omega^2, \cdots, \omega^t)$ rather than $(\omega^1, \omega^2, \cdots, \omega^T)$ ([14]). Here, $\omega^1, \omega^2, \cdots, \omega^T$ denote the scenarios of the scenario tree $\mathbb{P}$. In this case, $\tilde{\omega}_{t,\phi_t(k)}$, the tth component of $f(\omega_0, \omega_{1,\phi_1(i)}, \omega_{2,\phi_2(i)}, \cdots, \omega_{T-1,\phi_{T-1}(i)}, \omega_{T,i})$, doesn't change if $(\omega_0, \omega_{1,\phi_1(i)}, \omega_{2,\phi_2(i)}, \cdots, \omega_{t,\phi_t(i)})$ doesn't change. This ensures that $\phi_\tau(i) \in \widetilde{I}_{t,\phi_\tau(k)}$ for $\tau = 1, 2, \cdots, t$, no matter how $(\omega_{t+1,\phi_{t+1}(i)}, \omega_{t+2,\phi_{t+2}(i)}, \cdots, \omega_{T-1,\phi_{T-1}}(i), \omega_{T,i})$ changes.

On the other hand, suppose that f defined by

$$f(\omega_0, \omega_{1,\phi_1(i)}, \omega_{2,\phi_2(i)}, \cdots, \omega_{T-1,\phi_{T-1}(i)}, \omega_{T,i}) = (\tilde{\omega}_0, \tilde{\omega}_{1,k_1}, \tilde{\omega}_{2,k_2}, \cdots, \tilde{\omega}_{T,k_T})$$

is a non-anticipative transport map such that $\widetilde{\mathbb{P}} = \mathbb{P} \circ f^{-1}$ is a multi-stage scenario tree. Denote the tth component of $f(\omega_0, \omega_{1,\phi_1(i)}, \omega_{2,\phi_2(i)}, \cdots, \omega_{T-1,\phi_{T-1}(i)},\ \omega_{T,i})$ by $f(\omega_0, \omega_{1,\phi_1(i)}, \cdots,\ \omega_{t,\phi_t(i)})_t$. Let

$$\widetilde{I}_{t,k} = \{l | f(\omega_0, \omega_{1,\phi_1(l)}, \cdots, \omega_{t-1,\phi_{t-1}(l)}, \omega_{t,l})_t = \tilde{\omega_{t,k}}\},$$

$$k = 1, 2, \cdots, \widetilde{N}_t,\ t = 1, 2, \cdots, T,$$

where $\widetilde{N}_t$ is the node number at stage t of the scenario tree $\widetilde{\mathbb{P}}$, $t = 1, 2, \cdots, T$. Because $\widetilde{\mathbb{P}}$ is a scenario tree, there are no different nodes at any stage which share the same son nodes. So $\widetilde{I}_{t,k}$ satisfies the first constraint of the model (11), and $\widetilde{N}_t^j$ can be easily determined from $\widetilde{I}_{t,k}$. Then, $\widetilde{N}_t^j$, $j = 1, 2, \cdots, \widetilde{N}_{t-1}$ and $\widetilde{I}_{t,m}, \tilde{\omega}_{t,m}$, $m = 1, 2, \cdots, \widetilde{N}_t$, $t = 1, 2, \cdots, T$, form a the feasible solution to the problem (11).

The above demonstration tells us that, there is a one to one correspondence between a non-anticipative transport map f such that $\widetilde{\mathbb{P}} = \mathbb{P} \circ f^{-1}$ is a multi-stage scenario tree and a feasible solution to problem (11). With this conclusion, we then know from Theorem 16 in [14] that under some conditions, the difference between the optimal value of a multi-stage stochastic programming problem based on $\mathbb{P}$ and that based on $\widetilde{\mathbb{P}}$ can be bounded from above by $Ec(\omega, f(\omega)) = \sum_{i=1}^{N} p_i c(\omega_i, f(\omega_i))$, here ω denotes a scenario of the tree $\mathbb{P}$, and $c(\cdot, \cdot)$ is the cost function. If $c(\cdot, \cdot) = \| \cdot - \cdot \|^r$, then $Ec(\omega, f(\omega))$ equals to the objective function of the problem (11) regardless of the

power $\frac{1}{r}$.

Summarizing the above conclusions, we obtain the following theorem.

Theorem 3. *The optimal solution to the problem* (11) *corresponds to a non-anticipative transport map f which makes $\widetilde{\mathbb{P}} = \mathbb{P} \circ f^{-1}$ a multi-stage scenario tree and meanwhile minimizes $Ec(\omega, f(\omega))$ with $c(\cdot,\cdot) = \| \cdot - \cdot \|^r$. The minimum is equal to the optimum value of the problem* (11).

Theorem 3 tells us that, when the node number at each stage of the reduced tree is pre-specified, the scenario tree $\widetilde{\mathbb{P}}$ obtained by the model (11) is the optimal one in the following sense: among all the scenario trees with the same pre-specified structure, the optimal value of the multi-stage stochastic programming problem based on $\widetilde{\mathbb{P}}$ is the one closest to that based on the original scenario tree $\mathbb{P}$.

3 Solution of the scenario reduction model (11)

Due to the combinatorial property of its constraints, the model (11) is an NP-hard problem. Observe that the determination of $\widetilde{I}_{t,m}$, $m = 1, 2, \cdots, \widetilde{N}_t$, relies on $\widetilde{I}_{t-1,m}$, $m = 1, 2, \cdots, \widetilde{N}_{t-1}$, and that the objective function of the problem (11) is separable with respect to t, if not considering the power $\frac{1}{r}$, we can divide the solution of problem (11) into T subproblems corresponding to different stages. The resulting approximate solution method can be described as follows.

We first solve the following problem corresponding to the first stage:

$$\begin{aligned} \min_{\widetilde{I}_{1,m}, \tilde{\omega}_{1,m}} \ & \sum_{i=1}^{N_1} p_{t,i} \parallel \omega_{t,i} - y_{t,i} \parallel^r \\ \text{s.t.} \ & \bigcup_{m=1}^{\widetilde{N}_1} \widetilde{I}_{1,m} = I_1^1, \\ & \widetilde{I}_{1,m_1} \cap \widetilde{I}_{1,m_2} = \emptyset, \, \forall\, m_1 \neq m_2, \, m_1, m_2 = 1, 2, \cdots, \widetilde{N}_1, \\ & y_{1,k} = \tilde{\omega}_{1,m}, \, \text{for}\, k \in \widetilde{I}_{1,m}, \, m = 1, 2, \cdots, \widetilde{N}_1. \end{aligned} \tag{12}$$

Actually, the problem (12) is the same as the problem (3). For $t = 2, 3, \cdots, T$, we solve the problem corresponding to the tth-stage based on the solution to the problem corresponding to the (t-1)th-stage. The tth problem, $t = 2, 3, \cdots, T$, can be described

as

$$
\begin{aligned}
\min_{\widetilde{I}_{t,m},\tilde{\omega}_{t,m},\widetilde{N}_t^j} \ & \sum_{i=1}^{N_t} p_{t,i} \parallel \omega_{t,i} - y_{t,i} \parallel^r \\
\text{s.t.} \ & \bigcup_{m_j=m_{j-1}+1}^{m_{j-1}+\widetilde{N}_t^j} \widetilde{I}_{t,m_j} = I_t^j, \\
& \widetilde{I}_{t,m_j+k_1} \cap \widetilde{I}_{t,m_j+k_2} = \emptyset, \, \forall\, k_1 \neq k_2, \, k_1, k_2 = 1, 2, \cdots, \widetilde{N}_t^j, \\
& m_0 = 0, \, j = 1, 2, \cdots, \widetilde{N}_{t-1}, \\
& \sum_{j=1}^{\widetilde{N}_{t-1}} \widetilde{N}_t^j = \widetilde{N}_t, \\
& y_{t,k} = \tilde{\omega}_{t,m}, \text{ for } k \in \widetilde{I}_{t,m}, \, m = 1, 2, \cdots, \widetilde{N}_t.
\end{aligned} \tag{13}
$$

The problem (13) is still very hard to solve because of the first and fourth constraints. If $\widetilde{N}_t^j$, $j = 1, 2, \cdots, \widetilde{N}_{t-1}$, are given and satisfy the fourth constraint, we just need to handle the first constraint. In practice, it is possible and/or reasonable to specify $\widetilde{N}_t^j$ beforehand. This means we specify the number of the son nodes for each non-leaf node, or equivalently, the structure of the reduced scenario tree. Sometimes people want to reduce a large-scale scenario tree to a specified-structure scenario tree, based on which the relevant multi-stage stochastic programming problem can be solved quickly. If $\widetilde{N}_t^j$, $j = 1, 2, \cdots, \widetilde{N}_{t-1}$, are fixed beforehand, the problem (13) can be equivalently separated into $\widetilde{N}_{t-1}$ problems. The jth problem, $j = 1, 2, \cdots, \widetilde{N}_{t-1}$, is

$$
\begin{aligned}
\min_{\widetilde{I}_{t,m_j},\tilde{\omega}_{t,m_j}} \ & \sum_{i \in I_t^j} p_{t,i} \parallel \omega_{t,i} - y_{t,i} \parallel^r \\
\text{s.t.} \ & \bigcup_{m_j=m_{j-1}+1}^{m_{j-1}+\widetilde{N}_t^j} \widetilde{I}_{t,m_j} = I_t^j, \\
& \widetilde{I}_{t,m_j+k_1} \cap \widetilde{I}_{t,m_j+k_2} = \emptyset, \, \forall\, k_1 \neq k_2, \, k_1, k_2 = 1, 2, \cdots, \widetilde{N}_t^j, \\
& y_{t,k} = \tilde{\omega}_{t,m_j}, \text{ for } k \in \widetilde{I}_{t,m_j}, \, m_j = m_{j-1}+1, m_{j-1}+2, \cdots, m_{j-1}+\widetilde{N}_t^j.
\end{aligned} \tag{14}
$$

The problem (14) is also the same as the problem (3). Therefore, to solve the problem (11), we need to design efficient methods to solve the problem (3).

3.1 Solution of the problem (3)

Solving the problem (3) is equivalent to determining an optimal partition $\{\widetilde{I}_m\}_{m=1}^{\widetilde{N}}$ of $\{1, 2, \cdots, N\}$ and the corresponding scenario value assignment $\tilde{\omega}_m$, $m = 1, 2, \cdots, \widetilde{N}$. We will devise two approximate methods to solve problem (3). A recursive-type method is proposed first.

To derive the recursive algorithm, we first consider the special case with $\widetilde{N} = N - 1$. In this case, the set $\{1, 2, \cdots, N\}$ should be divided into $N - 1$ sets, i.e., $\{\widetilde{I}_m^{(N-1)}\}_{m=1}^{N-1}$. There must be one and only one set among $\{\widetilde{I}_m^{(N-1)}\}_{m=1}^{N-1}$ which contains two elements,

e.g., j and k. In order to minimize the objective function under this situation, we should have $y_i^{(N-1)} = \omega_i$ for $i \neq j, k$ and $i \in \{1, 2, \cdots, N\}$. Suppose that $y_j^{(N-1)} = y_k^{(N-1)} = x$, then the objective function is equal to

$$\{p_j \parallel \omega_j - x \parallel^r + p_k \parallel \omega_k - x \parallel^r\}^{1/r}.$$

Assume that $\tilde{\omega}_{jk} \in \arg\min_x \{p_j \parallel \omega_j - x \parallel^r + p_k \parallel \omega_k - x \parallel^r\}^{1/r}$, we should then set $y_j^{(N-1)} = y_k^{(N-1)} = \tilde{\omega}_{jk}$.

Furthermore, to minimize the objective function, $\{j, k\}$ should satisfy

$$\begin{aligned} &\min_x \{p_j \parallel \omega_j - x \parallel^r + p_k \parallel \omega_k - x \parallel^r\}^{1/r} = \\ &\min_{h,g \in \{1,2,\cdots,N\}} \min_x \{p_h \parallel \omega_h - x \parallel^r + p_g \parallel \omega_g - x \parallel^r\}^{1/r}. \end{aligned} \tag{15}$$

When $r = 2$, we can easily derive that

$$\frac{p_g \omega_g + p_h \omega_h}{p_g + p_h} = \arg\min_x \{p_h \parallel \omega_h - x \parallel^2 + p_g \parallel \omega_g - x \parallel^2\}^{1/2}, \tag{16}$$

and

$$\min_x \{p_h \parallel \omega_h - x \parallel^2 + p_g \parallel \omega_g - x \parallel^2\}^{1/2} = \{\frac{p_g p_h}{p_g + p_h} \parallel \omega_g - \omega_h \parallel^2\}^{1/2}. \tag{17}$$

Without loss of generality, we assume that $\{j, k\}$ satisfies the equation (15). Then we obtain an optimal solution with $\tilde{\omega}_m^{(N-1)} = \omega_m$, $I_m^{(N-1)} = \{m\}$ for $m \neq j, k$ and $m = 1, 2, \cdots, N$, and $I_{jk}^{(N-1)} = \{j, k\}$. The probability of $\tilde{\omega}_m^{(N-1)}$ is $\tilde{p}_m^{(N-1)} = p_m$ for $m \neq j, k$ and $m = 1, 2, \cdots, N$, and that of $\tilde{\omega}_{jk}$ is $\tilde{p}_{jk} = p_{N-1} + p_N$. To make it convenient for the following presentation, we set $\tilde{\omega}_i^{(N-1)} = \tilde{\omega}_{m_i}^{(N-1)}$, $I_i^{(N-1)} = I_{m_i}^{(N-1)}$, $\tilde{p}_i^{(N-1)} = \tilde{p}_{m_i}^{(N-1)}$ where m_i is the ith minimum element of the set $\{m | m \neq j, k, m = 1, 2, \cdots, N\}$, $\widetilde{I}_{N-1}^{(N-1)} = \widetilde{I}_{jk}^{(N-1)}$, $\tilde{\omega}_{N-1}^{(N-1)} = \tilde{\omega}_{jk}$ and $\tilde{p}_{N-1}^{(N-1)} = \tilde{p}_{jk}$.

Next, we consider the case with $\widetilde{N} = N - 2$. The problem (3) under this case is solved in two steps: first, we apply the above method to the case with $\widetilde{N} = N - 1$; then, we set $\tilde{\omega}_m^{(N-1)}$ and $\tilde{p}_m^{(N-1)}$, $m = 1, 2, \cdots, N-1$, as the input to the model (3), and the right-hand side of the first constraint in (3) is $\{1, 2, \cdots, N-1\}$. With this transformation, it becomes the same as the case $\widetilde{N} = N - 1$. Thus, we can similarly find the optimal partition $\{\widetilde{I}_m^{(N-2)}\}_{m=1}^{N-2}$, the corresponding scenario value assignment $\tilde{\omega}_m^{(N-2)}$ and probabilities $\tilde{p}_m^{(N-2)}$, $m = 1, 2, \cdots, N-2$. Observe that $\{\widetilde{I}_m^{(N-2)}\}_{m=1}^{N-2}$ is a partition of the set $\{1, 2, \cdots, N-1\}$ and that $m \in \{1, 2, \cdots, N-1\}$ corresponds to $\widetilde{I}_m^{(N-1)}$, we can let $\widetilde{I}_m^{(N-2)*} = \bigcup_{i \in \widetilde{I}_m^{(N-2)}} \widetilde{I}_i^{(N-1)}$ for $m = 1, 2, \cdots, N-2$. Then $\{\widetilde{I}_m^{(N-2)*}\}_{m=1}^{N-2}$ forms a partition of the set $\{1, 2, \cdots, N\}$. Let

$$\tilde{\omega}_m^{(N-2)*} \in \arg\min_x \{\sum_{i \in \widetilde{I}_m^{(N-2)*}} p_i \parallel \omega_i - x \parallel^r\}^{1/r} \tag{18}$$

for $m = 1, 2, \cdots, N-2$. Because $\widetilde{I}_m^{(N-2)*}$ contains only one or two elements for most $m \in \{1, 2, \cdots, N-2\}$, $\tilde{\omega}_m^{(N-2)*} = \tilde{\omega}_m^{(N-2)}$ due to the way we get $\tilde{\omega}_m^{(N-2)}$. If $\widetilde{I}_m^{(N-2)*}$ contains more than two elements, $\tilde{\omega}_m^{(N-2)*}$ should be determined by the equation (18).

Replacing $\tilde{\omega}_m^{(N-2)}$ by $\tilde{\omega}_m^{(N-2)*}$ for $m = 1, 2, \cdots, N-2$, the minimum in (18) can be further reduced because of the equation (18) and the form of the objective function of the problem (3). For notational convenience, we use $\tilde{\omega}_m^{(N-2)}$ and $\widetilde{I}_m^{(N-2)}$ to represent $\tilde{\omega}_m^{(N-2)*}$ and $\widetilde{I}_m^{(N-2)*}$ for $m = 1, 2, \cdots, N-2$.

Reducing the number of scenarios by one each time, the above recursive solution technique can be applied to the general case with any value of $\widetilde{N}$. Thus we obtain a recursive algorithm to solve the problem (3). The following algorithm shows the concrete procedure to solve the problem (3).

Algorithm 2: The recursive-type method

Step 1 Initialization. The input of the problem (3) is ω_i and p_i, $i = 1, 2, \cdots, N$. Let $\widetilde{I}_m^{(N)} = \{m\}$, $\tilde{\omega}_m^{(N)} = \omega_m$ and $\tilde{p}_m^{(N)} = p_m$ for $m = 1, 2, \cdots, N$. Set $n = N - 1$.

Step 2 Find the set $\{j, k\} \subset \{1, 2, \cdots, n+1\}$ such that

$$\min_x \{\tilde{p}_j^{(n+1)} \parallel \tilde{\omega}_j^{(n+1)} - x \parallel^r + \tilde{p}_k^{(n+1)} \parallel \tilde{\omega}_k^{(n+1)} - x \parallel^r\}^{1/r} =$$

$$\min_{h,g \in \{1,2,\cdots,n+1\}} \min_x \{\tilde{p}_h^{(n+1)} \parallel \tilde{\omega}_h^{(n+1)} - x \parallel^r + \tilde{p}_g^{(n+1)} \parallel \tilde{\omega}_g^{(n+1)} - x \parallel^r\}^{1/r}.$$

Set $\tilde{\omega}_m^{(n)} = \tilde{\omega}_m^{(n+1)}$, $\widetilde{I}_m^{(n)} = \widetilde{I}_m^{(n+1)}$ for $m \neq j, k$ and $m = 1, 2, \cdots, n+1$, $\widetilde{I}_{jk}^{(n)} = \widetilde{I}_j^{(n+1)} \bigcup \widetilde{I}_k^{(n+1)}$ and

$$\tilde{\omega}_{jk}^{(n)} \in \arg\min_x \{\tilde{p}_j^{(n+1)} \parallel \tilde{\omega}_j^{(n+1)} - x \parallel^r + \tilde{p}_k^{(n+1)} \parallel \tilde{\omega}_k^{(n+1)} - x \parallel^r\}^{1/r}. \tag{19}$$

Let the probability of $\tilde{\omega}_m^{(n)}$ be $\tilde{p}_m^{(n)} = \tilde{p}_m^{(n+1)}$for $m \neq j, k$ and $m = 1, 2, \cdots, n+1$, and that of $\tilde{\omega}_{jk}^{(n)}$ be $\tilde{p}_{jk}^{(n)} = \tilde{p}_j^{(n+1)} + \tilde{p}_k^{(n+1)}$. Let $\tilde{\omega}_i^{(n)} = \tilde{\omega}_{m_i}^{(n)}$, $I_i^{(n)} = I_{m_i}^{(n)}$, $\tilde{p}_i^{(n)} = \tilde{p}_{m_i}^{(n)}$, here m_i is the ith minimum element of the set $\{m | m \neq j, k,\ m = 1, 2, \cdots, n+1\}$, $\widetilde{I}_n^{(n)} = \widetilde{I}_{jk}^{(n)}$, $\tilde{\omega}_n^{(n)} = \tilde{\omega}_{jk}^{(n)}$ and $\tilde{p}_k^{(n)} = \tilde{p}_{jk}^{(n)}$.

Step 3 If the element number of $\widetilde{I}_m^{(n)}$ is greater than two, then let

$$\tilde{\omega}_m^{(n)} \in \arg\min_x \{\sum_{i \in \widetilde{I}_m^{(n)}} p_i \parallel \omega_i - x \parallel^r\}^{1/r}, \tag{20}$$

for $m = 1, 2, \cdots, n$.

Step 4 Let $n = n - 1$. If $n < \widetilde{N}$, stop. Otherwise, go to Step 2.

In most cases, we have $r = 1$ or $r = 2$, then problems (19) and (20) can be solved in polynomial time and so is Algorithm 2. Especially, when $r = 2$, the number of multiplications in Algorithm 2 is $6N^2 + N - 2\widetilde{N}^2 - \widetilde{N}$. If $\widetilde{N}$ is much smaller than N, then the number of multiplications is $O(N^2)$. One can refer to the Appendix for detailed demonstration.

The simultaneous backward reduction method in [4] is an effective way to solve the model (7). Without considering the computational burden of evaluating the cost function $c(\omega_j, \omega_k)$ and that of the redistribution rule, the number of multiplications in the simultaneous backward reduction method is

$$\sum_{i=1}^{N-\widetilde{N}}(i(N+i-1)) = \frac{(N-\widetilde{N})(N-\widetilde{N}+1)(5N-2\widetilde{N}-2)}{6}.$$

If $\widetilde{N}$ is much smaller than N, then the number of multiplications is $O(N^3)$. Therefore, Algorithm 2 with $r=2$ is faster than the simultaneous backward reduction method.

The above recursive-type method is an approximation algorithm, it solves the model (3) through solving several similar but relatively simple problems. To solve the model (3) as a whole, we now introduce a kind of cluster method. As we know, to solve the model (3) is to determine the optimal $\widetilde{I}_m$ and $\tilde{\omega}_m$ for $m = 1, 2, \cdots, \widetilde{N}$. Once $\widetilde{I}_m$ is determined, we have

$$\tilde{\omega}_m \in \arg\min_x \{\sum_{j\in\widetilde{I}_m} p_j \parallel \omega_j - x \parallel^r\}^{1/r}. \tag{21}$$

On the other hand, once $\tilde{\omega}_m$ is determined, we have

$$\widetilde{I}_m = \{i \in \{1,2,\cdots,N\} |\ \parallel \omega_i - \tilde{\omega}_m \parallel^r = \min_{j\in\{1,2,\cdots,\widetilde{N}\}} \parallel \omega_i - \tilde{\omega}_j \parallel^r\}. \tag{22}$$

Inspired by the k-means method ([19]), we design the following r-cluster method.

Algorithm 3: the r-cluster method

Step 1 Initialization. Given the input parameters of the problem (3): ω_i and p_i, $i = 1, 2, \cdots, N$. Choose the accuracy parameter $\epsilon > 0$ and let $n = 1$.

Step 2 Randomly choose $\widetilde{N}$ elements from $\{\omega_1, \omega_2, \cdots, \omega_N\}$ and denote them as $\{\tilde{\omega}_1^{(n)}, \tilde{\omega}_2^{(n)}, \cdots, \tilde{\omega}_{\widetilde{N}}^{(n)}\}$. For $m = 1, 2, \cdots, \widetilde{N}$, determine

$$\widetilde{I}_m^{(n)} = \{i \in \{1,2,\cdots,N\} |\ \parallel \omega_i - \tilde{\omega}_m^{(n)} \parallel^r = \min_{j\in\{1,2,\cdots,\widetilde{N}\}} \parallel \omega_i - \tilde{\omega}_j^{(n)} \parallel^r\}.$$

Substitute them into the model (3) and obtain the value of the objective function v_n.

Step 3 For $m = 1, 2, \cdots, \widetilde{N}$, let

$$\tilde{\omega}_m^{(n+1)} \in \arg\min_x \{\sum_{i\in I_m^{(n)}} p_i \parallel \omega_i - x \parallel^r\}^{1/r},$$

and

$$\widetilde{I}_m^{(n+1)} = \{i \in \{1,2,\cdots,N\} |\ \parallel \omega_i - \tilde{\omega}_m^{(n+1)} \parallel^r = \min_{j\in\{1,2,\cdots,\widetilde{N}\}} \parallel \omega_i - \tilde{\omega}_j^{(n+1)} \parallel^r\}.$$

Substitute them into the model (3) and obtain the value of the objective function v_{n+1}.

Step 4 If $|v_n - v_{n+1}| > \epsilon$, let $n = n+1$ and go to Step 3. Otherwise, stop. Take $\widetilde{I}_m^{(n+1)}$ and $\tilde{\omega}_m^{(n+1)}$, $m = 1, 2, \cdots, \widetilde{N}$, as a solution to the model (3). The probability of node $\tilde{\omega}_m^{(n+1)}$ is $\tilde{p}_m^{(n+1)} = \sum_{j\in\widetilde{I}_m^{(n+1)}} p_j$.

Remark 1. *As a variant of the k-means method, Algorithm 3 is definitely convergent. When $r = 1$ or $r = 2$, problem (21) can be solved in polynomial time and so is Algorithm*

3. Especially, when $r = 2$, we have

$$\tilde{\omega}_m^{(n+1)} = \frac{\sum_{i \in \widetilde{I}_m^{(n)}} p_i \omega_i}{\sum_{i \in \widetilde{I}_m^{(n)}} p_i},$$

and the number of multiplications in Algorithm 3 is $(K\widetilde{N} + K - 1)N + (K-1)\widetilde{N}$, here K is the number of iterations. Indeed, Step 2 requires $\widetilde{N}N$ multiplications. At each iteration, the first equation of Step 3 requires $N + \widetilde{N}$ multiplications and the second one requires $\widetilde{N}N$ multiplications.

In most cases, N is much larger than K and $\widetilde{N}$. Then the number of multiplications in Algorithm 3 is $O(N)$ when $r = 2$. This means that Algorithm 3 consumes less time than both Algorithm 2 and the simultaneous backward reduction method.

What happens if $\widetilde{N}$ is close to N? We know that Algorithm 2 is a recursive method. If $\widetilde{N}$ is close to N, Algorithm 2 would stop after a small number of iterations. In this case, Algorithm 2 should take less time than Algorithm 3. To illustrate this, let

$$K(\widetilde{N} + 1)N + K > 6N^2 + N - 2\widetilde{N}^2 - \widetilde{N},$$

we get

$$\widetilde{N} > \frac{-(KN+1) + \sqrt{(KN+1)^2 - 8(KN + K - 6N^2 - N)}}{4}.$$

Then

$$\frac{\widetilde{N}}{N} > \frac{-(K + \frac{1}{N}) + \sqrt{(K + \frac{1}{N})^2 - 8(\frac{K}{N} + \frac{K}{N^2} - 6 - \frac{1}{N})}}{4}.$$

If K is much smaller than N, than the right-hand side of the above inequality is approximately equal to $\frac{1}{4}(-K + \sqrt{K^2 + 48})$. It means that if $\frac{\widetilde{N}}{N} > \frac{1}{4}(-K + \sqrt{K^2 + 48})$, then Algorithm 2 consumes less time than Algorithm 3 where $r = 2$. For instance, if $K = 10$, $\frac{1}{4}(-K + \sqrt{K^2 + 48}) = 0.54$. That is to say, if $\widetilde{N}$ is at least $0.54N$, then Algorithm 2 consumes less time than Algorithm 3. The above argument will be supported by the numerical experiments in Section 5.

Furthermore, when $\widetilde{N}$ is close to N, the optimal value of the model (3) obtained by Algorithm 2 should be smaller than that by Algorithm 3. We first examine the case when $\widetilde{N}$ is much smaller than N. As a recursive method, the computational error of Algorithm 2 accumulates each iteration it solves the corresponding subproblem to reduce the number of scenarios by one. The iteration number is large when $\widetilde{N}$ is far smaller than N, which results in a big accumulated error. However, Algorithm 3 tries to solve the model (3) as a whole, which may lead to a relatively small error. But when $\widetilde{N}$ is close to N, things will change. If $\widetilde{N} = N - 1$, Algorithm 2 returns a real optimal solution, while Algorithm 3 may not because it depends on the randomly selected initial points. If $\widetilde{N}$ is close to N, the accumulated error of Algorithm 2 should be relatively small and the error by Algorithm 3 should be large due to its dependence on the randomly selected initial points. This argument will also be supported by the numerical experiments in Section 5.

We use a simple example to illustrate the above demonstration. Suppose that $N = 3$ and the inputs of the model (3) are $\omega_1 = 1$, $\omega_2 = 2$, $\omega_3 = 3$, $p_1 = \dfrac{1}{2}$, $p_2 = \dfrac{1}{3}$

and $p_3 = \frac{1}{6}$. Let $\widetilde{N} = 2$ and assume $r = 2$. Algorithm 2 will return $\widetilde{\omega}_1 = 1$, $\widetilde{\omega}_2 = \frac{7}{3}$, $\widetilde{p}_1 = \frac{1}{2}$ and $\widetilde{p}_2 = \frac{1}{2}$, the optimal value is 0.333. If the selected initial points are ω_1 and ω_2, Algorithm 3 will return $\widetilde{\omega}_1 = 1$, $\widetilde{\omega}_2 = \frac{7}{3}$, $\widetilde{p}_1 = \frac{1}{2}$ and $\widetilde{p}_2 = \frac{1}{2}$, the optimal value is 0.333. However, if the selected initial points are ω_2 and ω_3, Algorithm 3 would then return $\widetilde{\omega}_1 = 3$, $\widetilde{\omega}_2 = \frac{7}{5}$, $\widetilde{p}_1 = \frac{1}{6}$ and $\widetilde{p}_2 = \frac{5}{6}$, the optimal value is 0.447.

3.2 Algorithms to solve the problem (11)

At the beginning of Section 3, we proposed a stage-wise approximate method to solve the scenario reduction model (11), which relies on the solution of the model (3). Based on the above two algorithms to solve the model (3), we design the following algorithm to solve the model (11), or equivalently a multi-stage scenario tree reduction algorithm.

Algorithm 4: stage-wise scenario tree reduction method

Step 1 Initialization. Choose $\widetilde{N}_t^j$, $j = 1, 2, \cdots, \widetilde{N}_{t-1}$, $t = 1, 2, \cdots, T$, such that $\sum_{j=1}^{\widetilde{N}_{t-1}} \widetilde{N}_t^j = \widetilde{N}_t$. $\widetilde{N}_0 = 1$ and let $\widetilde{N}_1^1 = \widetilde{N}_1$, $I_1^1 = \{1, 2, \cdots, N\}$, $t = 1$. Select $0 < \delta < 1$.

Step 2 If $\frac{\widetilde{N}_1}{N_1} > \delta$, call Algorithm 2 to solve the problem (12); otherwise, call Algorithm 3. Then $\widetilde{I}_{1,m}$ and $\tilde{\omega}_{1,m}$, $m = 1, 2, \cdots, \widetilde{N}_1$, are determined.

Step 3 Let $t = t + 1$. If $t > T$, stop. Otherwise, let

$$I_t^j = \{i | \omega_{t,i} \in \Omega_t,\ \phi_{t-1}(i) \in \widetilde{I}_{t-1,j}\},$$

for $j = 1, 2, \cdots, \widetilde{N}_{t-1}$.

Step 4 For $j = 1, 2, \cdots, \widetilde{N}_{t-1}$, if $\frac{\widetilde{N}_t^j}{|I_t^j|} > \delta$, call Algorithm 2 to solve problem (14); otherwise, call Algorithm 3. Then we obtain $\widetilde{I}_{t,m}$ and $\tilde{\omega}_{t,m}$, $m = 1, 2, \cdots, \widetilde{N}_t$. Go to Step 3.

Remark 2. *The δ in Algorithm 4 represents the threshold value for the ratio between the original number of the nodes and the node number after reduction in the problem* (12) *or* (14)*. As we have pointed out in Section 3.1, if these two numbers are close to each other, Algorithm 2 performs better, otherwise, Algorithm 3 performs better.*

The model (11) and the resulting Algorithm 4 are effective for the reduction of both fan-liked scenario trees and general multi-stage scenario trees. However, if $\mathbb{P}$ is fan-liked, the model (11) can be directly solved as a whole rather than stage-by-stage because of the independence of $\mathbb{P}$'s scenarios.

We first find an initial guess for the nodes of the reduced scenario tree $\widetilde{\mathbb{P}}$: $\tilde{\omega}_{t,m}$, $m = 1, 2, \cdots, \widetilde{N}_t$, $t = 1, 2, \cdots, T$, but we don't know the occurring probabilities of these nodes. Suppose the node $\tilde{\omega}_{t,m}$ has $\widetilde{N}_{t+1}^m$ son nodes, $m = 1, 2, \cdots, \widetilde{N}_t$, such that $\sum_{m=1}^{\widetilde{N}_t} \widetilde{N}_{t+1}^m = \widetilde{N}_{t+1}$, $t = 1, 2, \cdots, T-1$. The jth scenario of the reduced scenario tree $\widetilde{\mathbb{P}}$ is then

$$\tilde{\omega}_j = (\omega_0, \tilde{\omega}_{1,\tilde{\phi}_1(j)}, \tilde{\omega}_{2,\tilde{\phi}_2(j)}, \cdots, \tilde{\omega}_{T-1,\tilde{\phi}_{T-1}(j)}, \tilde{\omega}_{T,j}),\ j = 1, 2, \cdots, \widetilde{N},$$

where $\widetilde{N} = \widetilde{N}_T$ and $\tilde{\phi}_t(j)$ is the father node at stage t, $t = 1, 2, \cdots, T-1$, of the leaf node j of the scenario tree $\widetilde{\mathbb{P}}$. Based on the initial node values, we can get $\widetilde{I}_{t,m}$, $m = 1, 2, \cdots, \widetilde{N}_t$, $t = 1, 2, \cdots, T$, by minimizing the objective function of the model (11). Then we update the values of $\tilde{\omega}_{t,m}$, $m = 1, 2, \cdots, \widetilde{N}_t$, $t = 1, 2, \cdots, T$, again to further reduce the value of the objective function of problem (11). Through alternately updating the sets $\widetilde{I}_{t,m}$ and the values of $\tilde{\omega}_{t,m}$, $m = 1, 2, \cdots, \widetilde{N}_t$, $t = 1, 2, \cdots, T$, until they do not change anymore, we can find an optimal solution to the problem (11). The following algorithm shows the concrete procedure.

Algorithm 5: reduction of fan-liked multi-stage scenario trees.

Step 1 Initialization. Given the inputs of the model (11): $\omega_{t,i}$ and $p_{t,i}$, $i = 1, 2, \cdots, N$, $t = 1, 2, \cdots, T$. The ith scenario of $\mathbb{P}$ is $\omega_i = (\omega_0, \omega_{1,i}, \omega_{2,i}, \cdots, \omega_{T,i})$ and its probability is $p_i = p_{1,i} = p_{2,i} = \cdots = p_{T,i}$, $i = 1, 2, \cdots, N$. Choose the accuracy parameter $\epsilon > 0$, and a set of initial node values $\tilde{\omega}_{t,m}^{(n)}$, $m = 1, 2, \cdots, \widetilde{N}_t$, $t = 1, 2, \cdots, T$. Let $n = 1$, $\widetilde{N} = \widetilde{N}_T$ and

$$\tilde{\omega}_j^{(n)} = (\omega_0, \tilde{\omega}_{1,\tilde{\phi}_1(j)}^{(n)}, \tilde{\omega}_{2,\tilde{\phi}_2(j)}^{(n)}, \cdots, \tilde{\omega}_{T-1,\tilde{\phi}_{T-1}(j)}^{(n)}, \tilde{\omega}_{T,j}^{(n)}),\ j = 1, 2, \cdots, \widetilde{N}.$$

Step 2 Define

$$\widetilde{I}_j^{(n)} = \{i \in \{1, 2, \cdots, N\} | \parallel \omega_i - \tilde{\omega}_j^{(n)} \parallel^r = \min_{k \in \{1,2,\cdots,\widetilde{N}\}} \parallel \omega_i - \tilde{\omega}_k^{(n)} \parallel^r\} \tag{23}$$

for $j = 1, 2, \cdots, \widetilde{N}$,

$$A_{t,m}^{(n)} = \{j \in \{1, 2, \cdots, \widetilde{N}\} | \tilde{\phi}_t(j) = m\}, \tag{24}$$

and

$$\widetilde{I}_{t,m}^{(n)} = \bigcup_{j \in A_{t,m}^{(n)}} \widetilde{I}_j^{(n)}, \tag{25}$$

$m = 1, 2, \cdots, \widetilde{N}_t$, $t = 1, 2, \cdots, T$. Substitute them into the model (11) to obtain the value of the objective function, denote it as v_n.

Step 3 Let

$$\tilde{\omega}_{t,m}^{(n+1)} \in \arg\min_x \{\sum_{i \in \widetilde{I}_{t,m}^{(n)}} p_{t,i} \parallel \omega_{t,i} - x \parallel^r\} \tag{26}$$

and

$$\tilde{\omega}_j^{(n+1)} = (\omega_0, \tilde{\omega}_{1,\tilde{\phi}_1(j)}^{(n+1)}, \tilde{\omega}_{2,\tilde{\phi}_2(j)}^{(n+1)}, \cdots, \tilde{\omega}_{T-1,\tilde{\phi}_{T-1}(j)}^{(n+1)}, \tilde{\omega}_{T,j}^{(n+1)}),\ j = 1, 2, \cdots, \widetilde{N}.$$

And then define

$$\widetilde{I}_j^{(n+1)} = \{i \in \{1, 2, \cdots, N\} | \parallel \omega_i - \tilde{\omega}_j^{(n+1)} \parallel^r = \min_{k \in \{1,2,\cdots,\widetilde{N}\}} \parallel \omega_i - \tilde{\omega}_k^{(n+1)} \parallel^r\},$$

for $j = 1, 2, \cdots, \widetilde{N}$, $A_{t,m}^{(n+1)} = \{j \in \{1, 2, \cdots, \widetilde{N}\} | \tilde{\phi}_t(j) = m\}$, and $\widetilde{I}_{t,m}^{(n+1)} = \bigcup_{j \in A_{t,m}^{(n+1)}} \widetilde{I}_j^{(n+1)}$, $t = 1, 2, \cdots, T$. Substitute them into the model (11) to obtain the value of the objective function, denote it as v_{n+1}.

Step 4 If $|v_n - v_{n+1}| < \epsilon$, stop. $\tilde{\omega}_{t,m}^{(n+1)}$ and $\widetilde{I}_{t,m}^{(n+1)}$, $m = 1, 2, \cdots, \widetilde{N}_t$, $t = 1, 2, \cdots, T$,

constitute an optimal solution to the problem (11). The probability of the node $\tilde{\omega}_{t,m}^{(n+1)}$ is $\tilde{p}_{t,m}^{(n+1)} = \sum_{i \in \widetilde{I}_{t,m}^{(n+1)}} p_m$, $m = 1, 2, \cdots, \widetilde{N}_t$, $t = 1, 2, \cdots, T$. Otherwise, let $n = n + 1$ and go to Step 3.

Remark 3. *The equation* (23) *means that the scenario indexes of* $\mathbb{P}$ *which are the closest ones to the* j*th scenario among all the scenarios of* $\widetilde{\mathbb{P}}$ *are assigned to the set* $\widetilde{I}_j$. $A_{t,m}^{(n)}$ *stands for the index set of the leaf nodes whose father nodes at stage* t *are the* m*th node.*

The termination criterion $|v_n - v_{n+1}| < \epsilon$ relies on the convergence of $\{v_n\}_{n=1}^{\infty}$. The following theorem demonstrates the convergence of $\{v_n\}_{n=1}^{\infty}$, thus the convergence of Algorithm 5.

Theorem 4. $\{v_n\}_{n=1}^{\infty}$ *decreasingly converges to a positive number.*

Proof. From the expression of the objective function of the model (11) and the equations (23) and (26), we have

$$\begin{aligned}
v_n &= \{\sum_{j=1}^{\widetilde{N}} \sum_{i \in \widetilde{I}_j^{(n)}} p_i \parallel \omega_i - \tilde{\omega}_i^{(n)} \parallel^r\}^{\frac{1}{r}} \\
&= \{\sum_{t=1}^{T} \sum_{m=1}^{\widetilde{N}_t} \sum_{i \in \widetilde{I}_j^{(n)}} p_i \parallel \omega_{t,i} - \tilde{\omega}_{t,m}^{(n)} \parallel^r\}^{\frac{1}{r}} \\
&\geq \{\sum_{t=1}^{T} \sum_{m=1}^{\widetilde{N}_t} \sum_{i \in \widetilde{I}_j^{(n)}} p_i \parallel \omega_{t,i} - \tilde{\omega}_{t,m}^{(n+1)} \parallel^r\}^{\frac{1}{r}} \\
&= \{\sum_{j=1}^{\widetilde{N}} \sum_{i \in \widetilde{I}_j^{(n)}} p_i \parallel \omega_i - \tilde{\omega}_i^{(n+1)} \parallel^r\}^{\frac{1}{r}} \\
&\geq \{\sum_{j=1}^{\widetilde{N}} \sum_{i \in \widetilde{I}_j^{(n+1)}} p_i \parallel \omega_i - \tilde{\omega}_i^{(n+1)} \parallel^r\}^{\frac{1}{r}} = v_{n+1}.
\end{aligned}$$

So $\{v_n\}_{n=1}^{\infty}$ is a decreasing series. Since $v_n > 0$ always holds, $\{v_n\}_{n=1}^{\infty}$ must converge to a positive number. □ □

Because Algorithm 5 solves the model (11) as a whole when $\mathbb{P}$ is a fan-liked multistage scenario tree, it should return a smaller optimal value than that of Algorithm 4. However, it takes more time than Algorithm 4 when $r = 2$. Indeed, when $r = 2$, Step 2 of Algorithm 5 requires $TN\widetilde{N}$ multiplications and at any iteration, the first equation of Step 3 requires $N + \widetilde{N}_t$ multiplications and the third one requires $TN\widetilde{N}$. If Algorithm 5 terminates after K_0 iterations, the total number of multiplications is then

$$M_1 = [K_0 T\widetilde{N} + (K_0 - 1)T]N + (K_0 - 1)\sum_{t=1}^{T} \widetilde{N}_t.$$

On the other hand, if we use Algorithm 3 to solve the model (14), the multiplication number is

$$(K_t^j \widetilde{N}_t^j + K_t^j - 1)N_t^j + (K_t^j - 1)\widetilde{N}_t^j,$$

where K_t^j is the iteration number and N_t^j is the element number of I_t^j. Because $\mathbb{P}$ is a fan-liked multi-stage scenario tree, $\sum_{j=1}^{\widetilde{N}_{t-1}} N_t^j = N$ for $t = 1, 2, \cdots, T$. Let $K = \max\{K_t^j | j = 1, 2, \cdots, \widetilde{N}_{t-1}, t = 1, 2, \cdots, T\}$. Suppose that Algorithm 4 always uses Algorithm 3 to solve the model (14). Then the number of multiplications in Algorithm 4 is bounded from above by

$$M_2 = K \sum_{t=1}^{T} \sum_{j=1}^{\widetilde{N}_{t-1}} \widetilde{N}_t^j N_t^j + T(K-1)N + (K-1) \sum_{t=1}^{T} \widetilde{N}_t.$$

This upper bound can be further reduced when we also use Algorithm 2 to solve the model (14). If $K_0 \geq K$, which holds in most cases, then

$$M_1 - M_2 \geq K \sum_{t=1}^{T} (\widetilde{N}N - \sum_{j=1}^{\widetilde{N}_{t-1}} \widetilde{N}_t^j N_t^j) \geq K \sum_{t=1}^{T} (\widetilde{N}_t N - \sum_{j=1}^{\widetilde{N}_{t-1}} \widetilde{N}_t^j N_t^j),$$

because

$$\widetilde{N}_t N - \sum_{j=1}^{\widetilde{N}_{t-1}} \widetilde{N}_t^j N_t^j = \sum_{j=1}^{\widetilde{N}_{t-1}} \widetilde{N}_t^j \sum_{j=1}^{\widetilde{N}_{t-1}} N_t^j - \sum_{j=1}^{\widetilde{N}_{t-1}} \widetilde{N}_t^j N_t^j > 0,$$

we have $M_2 > M_1$. This means that Algorithm 5 consumes more time than Algorithm 4. This argument will be supported by the numerical experiments in Section 5.

4 Multi-stage scenario tree reduction with a pre-specified distance

The model (11) is suitable for the multi-stage scenario tree reduction with a pre-specified structure. However, sometimes we want to find a scenario tree with the smallest possible scale among all those scenario trees whose distances to the original tree are equal to some value set beforehand. For this reason, a new scenario tree reduction method will be designed in this section. We have argued in Section 2 that the model (11) is a scenario reduction model based on a stability result of multi-stage stochastic programming problems, i.e., Theorem 16 in [14]. Here, we also try to design a new scenario tree reduction method based on the same theorem. Concretely, we will find an optimal non-anticipative transport map f such that $d_f = Ec(\omega, f(\omega)) = \sum_{i=1}^{N} p_i c(\omega_i, f(\omega_i))$ is equal or closest to the pre-specified distance and meanwhile the new scenario tree $\widetilde{\mathbb{P}} = \mathbb{P} \circ f^{-1}$ has the smallest scale.

Suppose that we have obtained an optimal non-anticipative transport map $f = (f_1, f_2, \cdots, f_n)$ such that the scenario tree $\widetilde{\mathbb{P}} = \mathbb{P} \circ f^{-1}$ has one less node than the original scenario tree $\mathbb{P}$, there must be two nodes at some stage of $\mathbb{P}$ whose images of the corresponding component of f are the same. That is to say, for the ith and jth nodes at stage t, $\omega_{t,i}$ and $\omega_{t,j}$, we have $f_t(\omega_{t,i}) = f_t(\omega_{t,j})$; meanwhile, the two nodes

must have the same father node in order to make $\widetilde{\mathbb{P}}$ a scenario tree. Therefore, to find the optimal f and $\widetilde{\mathbb{P}}$ which has one less node than $\mathbb{P}$, we should try to find an optimal couple of nodes such that d_f is minimized. If the couple of nodes are $\omega_{t,i}$ and $\omega_{t,j}$, then

$$d_f = p_{t,i}c(\omega_{t,i}, f(\omega_{t,i})) + p_{t,j}c(\omega_{t,j}, f(\omega_{t,j})).$$

Because $f_t(\omega_{t,i}) = f_t(\omega_{t,j})$, denoted by z, we have

$$d_f = p_{t,i}c(\omega_{t,i}, z) + p_{t,j}c(\omega_{t,j}, z).$$

To make sure that d_f is minimized, we should have

$$z \in \arg\min_x p_{t,i}c(\omega_{t,i}, x) + p_{t,j}c(\omega_{t,j}, x).$$

For any couple of nodes with the same father node, we can calculate the above z and d_f, find the smallest d_f and the corresponding z, replace the corresponding couple of nodes by z and set all the son nodes of this couple of nodes to be the son nodes of z, and finally, we obtain $\widetilde{\mathbb{P}}$. Here, the probability of the node z is the sum of the probabilities of the corresponding couple of nodes. Let $\epsilon_1 = d_f$. Treating $\widetilde{\mathbb{P}}$ as the original scenario tree, we can use the above method to further reduce the number of nodes by one, and let $\epsilon_2 = d_f$. We can repeat this process by reducing the number of nodes of the newly obtained scenario tree by one each time, until $\sum_i \epsilon_i$ is equal or closest to the pre-specified distance. The following algorithm shows the concrete procedure.

Algorithm 6: reduction of the multi-stage scenario tree with a pre-specified distance.

Step 1 Initialization. Given the original scenario tree $\mathbb{P}$ whose nodes and corresponding probabilities are $\omega_{t,i}$ and $p_{t,i}$ respectively, $i = 1, 2, \cdots, N_t$, $t = 1, 2, \cdots, T$. Choose the pre-specified distance $pred > 0$ and the accuracy parameter $\epsilon > 0$. Let $d = 0$.

Step 2 At stage t, $t = 2, 3, \cdots, T$, for all the couples of nodes $(\omega_{t,i}, \omega_{t,j})$, $1 \leq i < j \leq N_t$, which have the same father node, calculate

$$d_f = p_{t,i}c(\omega_{t,i}, z) + p_{t,j}c(\omega_{t,j}, z)$$

where

$$z \in \arg\min_x p_{t,i}c(\omega_{t,i}, x) + p_{t,j}c(\omega_{t,j}, x).$$

Step 3 Among all the examined couples of nodes, find the smallest d_f and the corresponding z. Replace the corresponding couple of nodes by a single node z and set the son nodes of this couple of nodes to be the son nodes of z, obtain or update $\widetilde{\mathbb{P}}$. The probability of the node z is the sum of the probabilities of the corresponding couple of nodes.

Step 4 Let $d = d + d_f$. If $|d - pred| < \epsilon$, stop, $\widetilde{\mathbb{P}}$ is the reduced scenario tree; otherwise, denote the node number at stage t of $\widetilde{\mathbb{P}}$ by N_t, and the nodes and corresponding probabilities by $\omega_{t,i}$ and $p_{t,i}$, respectively, $i = 1, 2, \cdots, N_t$, $t = 1, 2, \cdots, T$. Go to Step 2.

If $c(\cdot, \cdot) = \| \cdot - \cdot \|^r$ and r equals to 1 or 2, Algorithm 6 terminates in polynomial time.

5 Numerical results

In this section, we test the proposed scenario tree reduction algorithms by using real financial trading data. Twelve stocks from different industries in both Dow Jones Industrial Average and S&P 500 Indexes are selected as risky assets in the following experiments. They are AAPL, MSFT, JPM, MCD, BA, EL, WMT, HD, FDX, AXP, AMZN and SAM. We use the adjusted weekly close-prices of these stocks to compute their weekly return rates in the period from July 30, 1997 to November 2, 2015. The original data are downloaded from http://finance.yahoo.com/.

The initial large-scale single-stage scenario trees and fan-liked multi-stage scenario trees are generated by the VAR-MGARCH model in [20], and the original large-scale general multi-stage scenario trees are generated by the Algorithm 1 in [21]. By the way, it doesn't matter which scenario generation method we choose because we are focusing on scenario tree reduction methods.

5.1 Features of the proposed models and algorithms

First, we examine the performance of single-stage scenario tree reduction methods: Algorithm 2 and Algorithm 3, which corresponds to the recursive-type method and the r-cluster method, respectively, to solve the model (3). We compare them with the simultaneous backward reduction method (SBRM) in [4] to solve the model (7).

A single-stage scenario tree with 650 scenarios is generated by the VAR-MGARCH model and it is reduced to smaller trees with respectively 10, 20, 30, 40, 50, 90, 130, 170, 210, 250, 290, 330, 370, 410, 450, 490, 530, 570 and 610 scenarios. For each of three algorithms, Table 1 shows the corresponding distance between the reduced scenario tree and the original tree, and the time consumed by the corresponding method to obtain the reduced scenario tree. Here, we choose $r = 2$, and SNRT stands for the scenario number of the reduced tree, and Time(s) is the solution time in seconds.

From Table 1, it's clear that all the distances obtained by SBRM is larger than the corresponding distances obtained by both Algorithm 2 and Algorithm 3, which supports Theorem 2. Meanwhile, the time consumed by SBRM is much longer than the corresponding times consumed by Algorithms 2 and 3. Therefore, both Algorithm 2 and Algorithm 3 perform much better than SBRM and the model (3) is better than the model (7) when the original scenario tree is single-stage.

As we can see from Table 1, the distance decreases as the scenario number of the reduced scenario tree increases, no matter which method we choose. This is reasonable since the reduced scenario tree gets closer to the original tree as its scenario number increases. Except for a few abnormal cases, the time consumed by Algorithm 3 increases as the scenario number increases; conversely, the time consumed by Algorithm 2 decreases. These results support our arguments in Section 3.1. In terms of the distance, when the scenario number of the reduced scenario tree is no more than 40, Algorithm 3 is better than Algorithm 2; when the scenario number is no less than 290, Algorithm 2 is better than Algorithm 3. When the scenario number is between 50 and 250, the distances obtained by Algorithm 2 are all smaller than those obtained by Algorithm 3 and the average relative error is 6.4%, while the time consumed by Algorithm 2 is larger than that consumed by Algorithm 3 and the average relative error is 80.6%. So in this case, we think Algorithm 3 performs better than Algorithm 2. Therefore, one

should properly choose Algorithm 2 or Algorithm 3 according to the desired scale of the reduced tree.

As we can see from Table 2, the conclusions are similar to what we observed from Table 1 when we choose $r = 1$. Nevertheless, the time consumed by Algorithm 2 is always larger than that by Algorithm 3, because Algorithm 2 has to solve the problem (20) at each iteration which consumes more time than that when $r = 2$. Another difference is that, when the scenario number of the reduced tree is larger than 250, the distance obtained by SBRM is a little smaller than that by Algorithm 3 but still larger than that by Algorithm 2, while SBRM always takes much more time than Algorithms 2 and 3. These results further support the effectiveness of the model (3), rather than the model (7), and effectiveness of Algorithm 2 and Algorithm 3, rather than SBRM.

SNRT	Algorithm 3		Algorithm 2		SBRM	
	Distance	Time(s)	Distance	Time(s)	Distance	Time(s)
10	0.0923	0.31	0.0946	3.79	0.0979	723.5
20	0.0852	0.64	0.0869	3.78	0.0914	720.3
30	0.0802	1.16	0.0819	3.79	0.0877	719.5
40	0.0773	1.05	0.0780	3.79	0.0850	718.1
50	0.0751	1.42	0.0748	3.79	0.0826	717.2
90	0.0679	1.73	0.0655	3.79	0.0755	705.0
130	0.0626	2.11	0.0586	3.74	0.0699	684.3
170	0.0571	2.26	0.0532	3.70	0.0648	687.0
210	0.0536	2.79	0.0483	3.67	0.0601	627.0
250	0.0496	3.31	0.0439	3.62	0.0557	575.8
290	0.0459	3.82	0.0339	3.51	0.0514	529.6
330	0.0426	3.46	0.0361	3.42	0.0472	463.5
370	0.0387	3.88	0.0325	3.28	0.0430	395.6
410	0.0350	4.31	0.0289	3.09	0.0387	326.8
450	0.0311	3.56	0.0254	3.01	0.0343	248.0
490	0.0277	5.11	0.0218	2.61	0.0297	172.3
530	0.0225	4.15	0.0181	2.23	0.0248	107.7
570	0.0186	4.48	0.0140	1.84	0.0194	51.6
610	0.0126	4.76	0.0093	1.40	0.0129	14.6

Table 1: Results of single-stage scenario tree reduction by three different methods where the original scenario tree has 650 scenarios and $r = 2$

We have pointed out in Section 2.1 that both the model (3) and the model (7) are not effective for the reduction of multi-stage scenario trees because they just focus on the reduction of the scenario number and don't consider the reduction of non-leaf nodes. Here, we will test this argument by numerical experiments. In all the following experiments, we set $r = 2$. Five multi-stage scenario trees are generated by the Algorithm 1 in [21] and they are reduced to smaller scenario trees with different

SNRT	Algorithm 3		Algorithm 2		SBRM	
	Distance	Time(s)	Distance	Time(s)	Distance	Time(s)
10	0.2406	1.03	0.2555	13.54	0.2583	740.7
20	0.2240	1.41	0.2372	13.68	0.2392	733.4
30	0.2115	2.11	0.2229	13.74	0.2275	709.9
40	0.2016	2.15	0.2099	13.42	0.2185	710.9
50	0.1945	2.67	0.1999	13.35	0.2107	708.7
90	0.1728	3.04	0.1705	13.20	0.1861	706.4
130	0.1546	4.32	0.1501	13.07	1.1662	687.9
170	0.1403	4.41	0.1337	12.89	0.1487	662.8
210	0.1274	3.98	0.1184	12.64	0.1324	634.9
250	0.1167	4.63	0.1047	12.32	0.1172	583.8
290	0.1042	5.37	0.0915	12.00	0.1028	535.1
330	0.0907	4.04	0.0787	11.61	0.0891	467.0
370	0.0806	5.68	0.0675	11.19	0.0759	402.8
410	0.0682	4.99	0.0563	10.62	0.0634	325.9
450	0.0576	5.49	0.0461	10.00	0.0514	248.1
490	0.0454	6.02	0.0359	9.31	0.0398	174.8
530	0.0342	6.54	0.0260	8.50	0.0288	106.4
570	0.0234	5.23	0.0169	7.64	0.0184	51.3
610	0.0114	5.62	0.0082	6.71	0.0085	14.7

Table 2: Results of single-stage scenario tree reduction by three different methods where the original scenario tree has 650 scenarios and $r = 1$

numbers of scenarios by Algorithm 1 and SBRM, respectively. Table 3 shows the detailed results, where SOST stands for the branching structure of the original scenario tree. If a T-stage scenario tree's branching structure is $[b_1, b_2, \cdots, b_T]$, it means that each node at the stage t has b_{t+1} son nodes, $t = 0, 1, \cdots, T-1$. This kind of scenario trees are called symmetric scenario trees. SNN stands for the stage-wise node number of a scenario tree. If a T-stage scenario tree's stage-wise node number is $[n_1, n_2, \cdots, n_T]$, it means that it has n_t nodes at stage t, $t = 1, 2, \cdots, T$. For example, the two-stage scenario tree with the branching structure [30,25] is reduced to scenario trees with 90 scenarios by Algorithm 1 and SBRM, respectively. The reduced tree obtained by Algorithm 1 has 74 nodes at the first stage, which is much larger than the original scenario tree. The reason is that Algorithm 1 treats the scenarios of the original tree as independent paths and tries to separate them into 90 groups, which makes the scenario tree have 90 nodes at the first stage if the nodes with the same values are not viewed as a single node. So it's not strange that the reduced tree has so many nodes at the first stage. The same phenomenon occurs for the reduction of other scenario trees as Table 3 shows. Compared to Algorithm 1, SBRM does better in this aspect, but it is not good at controlling the non-leaf node number of the reduced tree. Even worse, SBRM always takes extremely longer time than Algorithm 1 to reduce the original tree to the tree with the same number of scenarios, and the distance obtained by SBRM is always larger than the corresponding distance obtained by Algorithm 1. Therefore, the performance of Algorithm 1 is better than that of SBRM.

SOST	SNRT	Algorithm 1			SBRM		
		SNN	Distance	Time(s)	SNN	Distance	Time(s)
[30,25]	90	[74,90]	0.0752	6.9	[30,90]	0.0818	1369
[35,30]	100	[86,100]	0.0797	15.6	[35,100]	0.0859	6415
[10,9,9]	100	[51,95,100]	0.0816	9.1	[10,51,100]	0.0896	2214
[11,10,9]	160	[62,147,160]	0.0719	18.0	[11,69,160]	0.0802	5965
[12,11,10]	180	[75,167,180]	0.0735	29.6	[12,83,180]	0.0816	19093

Table 3: Results of multi-stage scenario tree reduction by Algorithm 1 and SBRM

To check the performance of the improved reduction model and for comparison purpose, we adopt the tree structures of the reduced scenario trees obtained by SBRM in Table 3 as the inputs to Algorithm 4. In this way, we obtain the pre-specified structure for each reduced scenario tree, i.e., we know the son node number of each non-leaf node $\widetilde{N}_t^j$, $j = 1, 2, \cdots, \widetilde{N}_{t-1}$, $t = 1, 2, \cdots, T$, and use them as the input to Algorithm 4. The obtained results are presented in Table 4, where 'p-SNN' stands for the pre-specified structure and 'No.' the mark number of the original scenario tree. We can see from Table 4 that the distance obtained by Algorithm 4 is a little larger than that by SBRM because Algorithm 4 runs stage-wisely. Nevertheless, the differences of the pairing distances are always small, independent of the structure of the original scenario tree, the maximum relative difference is only 20.4%. This size of difference can be neglected in most applications because, more importantly, the time consumed by SBRM is almost always, except for one case, 10000 times bigger than that consumed by Algorithm 4. Furthermore, with the increase of the scale of the original scenario tree, the time consumed by SBRM increases rapidly, while the time needed by Algorithm 4

changes hardly any. The above results demonstrate, compared with SBRM, Algorithm 4 is very useful and efficient for the reduction of large-scale multi-stage scenario trees.

No.	SOST	SBRM			Algorithm 4		
		SNN	Distance	Time(s)	p-SNN	Distance	Time(s)
1	[30,25]	[30,90]	0.0818	1369	[30,90]	0.0986	0.12
2	[35,30]	[35,100]	0.0859	6415	[35,100]	0.1060	0.15
3	[10,9,9]	[10,51,100]	0.0896	2214	[10,51,100]	0.1123	0.25
4	[11,10,9]	[11,69,160]	0.0802	5965	[11,69,160]	0.1007	0.19
5	[12,11,10]	[12,83,180]	0.0816	19093	[12,83,180]	0.0982	0.18

Table 4: Results of multi-stage scenario tree reduction by Algorithm 4 and SBRM

To test the actual performance of Algorithm 5 we especially designed for the reduction of fan-liked multi-stage scenario trees, we compare Algorithm 4 with Algorithm 5. The fan-liked multi-stage scenario trees are generated by the VAR-MGARCH model in [20], which will be reduced to scenario trees with different structures. The results are given in Table 5, where SNFS stands for the scenario number of the fan-liked scenario tree and SRST the branching structure of the reduced scenario tree. We can see from Table 5 that the distance, the optimal value of the model (11), obtained by Algorithm 5 is always a little smaller than that obtained by Algorithm 4. The relative difference is between 2.8% and 24.4 %, which is neglectable. This is due to that Algorithm 5 is a global method while Algorithm 4 runs stage-wisely. Unfortunately, Algorithm 5 takes much longer time than Algorithm 4, especially when the scale of the fan-liked scenario tree is large. Moreover, the time increase speed of Algorithm 5 is faster than that of Algorithm 4. Therefore, Algorithm 5 is a good choice when the size of the fan-liked scenario tree is small; otherwise, we should use Algorithm 4.

Stage	SNFS	SRST	Algorithm 4		Algorithm 5	
			Distance	Time(s)	Distance	Time(s)
2	1000	[15,5]	0.3401	1.5	0.2571	6.2
2	2000	[20,6]	0.3242	3.5	0.2477	25.5
3	6000	[10,5,2]	1.0336	8.8	0.8249	128.2
3	8000	[12,6,3]	0.9672	15.5	0.7782	398.3
4	10000	[10,6,3,2]	1.5425	21.7	1.4989	750.8
4	13000	[11,6,3,2]	1.5376	27.2	1.2981	1042.2

Table 5: Results of fan-liked multi-stage scenario tree reduction by Algorithms 4 and 5

Finally, we consider the scenario tree reduction with a pre-specified distance. In this case, we can use Algorithm 6. To compare it with Algorithm 4, we consider the same scenario trees in Table 4 which have been reduced by Algorithm 4. Table 6 shows the results, the p-distance stands for the pre-specified distance, which is set to the distance between the original scenario tree and the reduced scenario tree obtained by Algorithm 4. We can see that Algorithm 6 generates a smaller scenario tree than that of Algorithm 4 with the same distance. The reason is that Algorithm 4 is a stage-wise

reduction method while Algorithm 6 is not. Compared with Algorithm 4, Algorithm 6 costs more time because it reduces the scenario tree node by node. The larger the original scenario tree, the longer the time Algorithm 6 will need. However, the time is not very long. Therefore, we can say that Algorithm 6 is practical and efficient.

SOST	Algorithm 4			Algorithm 6		
	SNN	Distance	Time(s)	SNN	p-distance	Time(s)
[30,25]	[30,90]	0.0986	0.12	[30,79]	0.0986	0.71
[35,30]	[35,100]	0.1060	0.15	[35,78]	0.1060	1.26
[10,9,9]	[10,51,100]	0.1123	0.25	[10,50,70]	0.1123	0.92
[11,10,9]	[11,69,160]	0.1007	0.19	[11,75,124]	0.1047	1.25
[12,11,10]	[12,83,180]	0.0982	0.18	[12,80,152]	0.0982	1.94

Table 6: Results of multi-stage scenario tree reduction by Algorithms 4 and 6

To further investigate the efficiency of Algorithms 4 and 6 for the reduction of large-scale multi-stage scenario trees, we generate a 3-stage scenario tree with the branching structure $[40, 30, 20]$ by Algorithm 1 in [21]. This scenario tree is reduced by Algorithm 4 to scenario trees whose branching structures are $[35, 27, 18]$, $[30, 24, 16]$, $[25, 21, 14]$, $[20, 18, 13]$ and $[15, 14, 13]$, respectively. Like before, we set the pre-specified distance, p-distance, to the distance between the original tree and the reduced tree got with Algorithm 4, and then reduce the original scenario tree by Algorithm 6. The results are shown in Table 7, where TNN stands for the total number of nodes in the reduced scenario tree. We can see that Algorithm 4 is more efficient for the reduction of large-scale multi-stage scenario trees than Algorithm 6 in terms of time. On the other hand, when we reduce the original 3-stage scenario tree with 24000 scenarios to a scenario tree with 2199 scenarios, Algorithm 6 only takes 511.9 seconds. Moreover, the number of scenarios and the total number of nodes in the reduced scenario tree got with Algorithm 6 are always smaller than those obtained with Algorithm 4. Therefore, we can still say that Algorithm 6 is practical and efficient for reduction of large-scale multi-stage scenario trees.

SRST	Algorithm 4				Algorithm 6			
	SNN	TNN	Distance	Time(s)	SNN	TNN	p-distance	Time(s)
[35,27,18]	[35,945,17010]	17990	0.0433	7.0	[40,1182,9767]	10989	0.0433	386.7
[30,24,16]	[30,720,11520]	12270	0.0615	8.0	[40,1134,6181]	7355	0.0615	448.1
[25,21,14]	[25,525,7350]	7900	0.0796	8.4	[40,1093,4814]	5947	0.0796	485.6
[20,18,13]	[20,360,4680]	5060	0.0968	8.4	[40,952,3129]	4121	0.0968	503.9
[15,14,13]	[15,210,2730]	2955	0.1106	8.1	[40,762,2119]	2921	0.1106	511.9

Table 7: Results of the 3-stage scenario tree [40,30,20] reduction by Algorithms 4 and 6

5.2 Application to multi-stage portfolio selection problems

The scenario tree is an efficient tool to solve multi-stage stochastic programming problems. The scenario tree reduction method is useful for reducing the size of the

resulting deterministic programming problem, and thus for quickly solving them. In this part, we test the proposed scenario tree reduction algorithms by applying them to solve the following multi-stage portfolio selection problem.

$$
\begin{aligned}
\min\ & \sum_{t=1}^{T} \lambda_t^- \frac{\sum_{k_t=1}^{K_t} p_t^{k_t} d_t^{-k_t}}{(R_0)^t} - \sum_{t=1}^{T} \lambda_t^- \frac{\sum_{k_t=1}^{K_t} p_t^{k_t} (x_t^{k_t} + \sum_{i=1}^{m} y_{i,t}^{k_t})}{(R_0)^t} \\
\text{s.t. } & x_0 + \sum_{i=1}^{m} y_{i,0} = W_0, \\
& y_{i,t}^{k_t} = (y_{i,t-1}^{a_{t-1}(k_t)} + v_{i,t-1}^{b,a_{t-1}(k_t)} - v_{i,t-1}^{s,a_{t-1}(k_t)}) R_{i,t}^{k_t},\ i = 1, 2, \cdots, m, \\
& x_t^{k_t} = (x_{t-1}^{a_{t-1}(k_t)} - \sum_{i=1}^{m} (1+\theta_{bi}) v_{i,t-1}^{b,a_{t-1}(k_t)} + \sum_{i=1}^{m} (1-\theta_{si}) v_{i,t-1}^{s,a_{t-1}(k_t)}) R_0, \\
& \sum_{i=1}^{m} y_{i,t}^{k_t} + x_t^{k_t} + d_t^{-k_t} - d_t^{+k_t} = G_t, \\
& v_{i,0}^{s,k_0} \le y_{i,0},\ v_{i,t-1}^{b,a_{t-1}(k_t)} \ge 0,\ v_{i,t-1}^{s,a_{t-1}(k_t)} \ge 0,\ x_t^{k_t} \ge 0,\ y_t^{k_t} \ge 0, \\
& d_t^{+k_t} \ge 0,\ d_t^{-k_t} \ge 0,\ k_t = 1, 2, \cdots, K_t,\ t = 1, 2, \cdots, T.
\end{aligned}
\tag{27}
$$

Here, x_t and $y_{i,t}$, $i = 1, 2, \cdots, m$, denote the investments in the risk-less asset and m risky assets at stage t, respectively, R_0 and $R_{i,t}$, $i = 1, 2, \cdots, m$, are their return rates. The superscript k_t means the k_tth node of the scenario tree at stage t, $a_{t-1}(k_t)$ stands for the ancestor node of the node k_t at stage $t-1$. $v_{i,t}^{b,k_t}$ and $v_{i,t}^{s,k_t}$ are the amounts of buying and selling the ith risky asset at node k_t, respectively, and θ_{bi} and θ_{si} denote the unit transaction cost for buying and selling the ith risky asset, respectively. G_t is the investor's target wealth at the beginning of period t, $d_t^{-k_t}$ and $d_t^{+k_t}$ denote the negative deviation and positive deviation of the wealth at stage t relative to the target G_t at node k_t. The first part of the objective function is the sum of the down-sided deviations in different periods and the second one is the sum of the wealths in different periods. The coefficient vector $\lambda^- = (\lambda_1^-, \lambda_2^-, \cdots, \lambda_T^-)$ is the weighting vector put onto the down-sided deviations and wealths in different periods. The model (27) is similar to the model in [20] except for the mean-risk form of the objective function.

We first solve the problem (27) based on the original scenario tree chosen in Section 5.1 and then solve it based on the corresponding reduced scenario trees obtained by different scenario tree reduction algorithms. By analyzing the results, the pros and cons of the proposed scenario tree reduction algorithms will be clear.

For comparison purpose, we solve the multi-stage portfolio selection problem with respect to the risk-less asset and 12 stocks chosen at the beginning of this section. Here, for simplicity, the initial budget 1000 dollar is fully invested in the risk-less asset. For the 2-stage portfolio selection problem, the stage-wise target wealth vector is selected as $G = (1002.5, 1005.0)$, and the weighting vector is $\lambda^- = (1, 1)$. And for the 3-stage portfolio selection problem, the stage-wise target wealth vector is selected as $G = (1002.5, 1005.0, 1007.5)$, and the weighting vector is $\lambda^- = (1, 1, 1)$. The unit transaction costs for buying and selling risky assets are 0.001 and 0.002, respectively. The weekly risk-less return rate is 1.0005.

Based on the numerical results in subsection 5.1, we consider here the application

of Algorithm 4, Algorithm 6 and SBRM to the solution of problem (27). Except for the 5 scenario trees examined in Table 4, we also consider a large-scale scenario tree [40,30,20], which is numbered as 6. Unfortunately, SBRM failed to reduce this big scenario tree. In what follows, we use i-S, $1 \leq i \leq 5$, to denote the reduced scenario tree corresponding to the ith scenario tree in Table 4 obtained by SBRM, i-4 (i-6), $1 \leq i \leq 5$, to denote the reduced tree obtained from the ith tree by Algorithm 4 (6). And we use 6-j-4 (6-j-6), $1 \leq j \leq 5$, to denote the 5 reduced scenario trees corresponding to the scenario tree 6 obtained by Algorithm 4 (6), respectively. For comparison purpose, we solve the resulting portfolio selection problems based on 6 original trees and all the above reduced trees, the results are shown in Table 8. Here, RID stands for the optimal root investment decision, OV the optimum vlaue, V-error the difference between the optimal value of the problem based on the reduced scenario tree and that based on the corresponding original scenario tree, FW the final wealth at the last stage, W-error the difference between the final wealth based on the reduced scenario tree and that based on the corresponding original scenario tree, and 'Percent' is the proportion of the time needed to solve the problem based on the reduced scenario tree to the time needed to solve the problem based on the original scenario tree.

In Table 8, the first component of RID is the money invested in the risk-less asset, the second component is that invested in the first stock and the third component is that invested in the second stock, and so on. We omitted those zero components due to the space limitation. From Table 8, it's clear that SBRM is not an appropriate method for the reduction of multi-stage scenario trees compared to Algorithms 4 and 6. When we solve the portfolio selection problem based on the reduced scenario tree obtained by SBRM, the RID, the optimal value and the final wealth deviate far away from those obtained under the original tree, the results are worse than those gotten with either Algorithm 4 or Algorithm 6. Especially, the W-errors generated by Algorithm 4 and Algorithm 6 are constantly no more than 1%, while that generated by SBRM is always larger than 1% and the maximum is even 6.9%. What's more, SBRM takes a lot time to reduce the scenario tree even if the scale of the original scenario tree is not large. In most cases, the time consumed by SBRM (see Table 4) is much longer than that consumed by solving the portfolio selection problem based on the original scenario tree, which makes it meaningless to reduce the original scenario tree by SBRM. Furthermore, when the scale of the original scenario tree is large, say for the sixth scenario tree [40,30,20], SBRM doesn't work anymore. On the other hand, Algorithms 4 and 6 perform well when they are applied to solve the portfolio selection problem, especially for the large-scale scenario tree case as we can see from the last two blocks in Table 8. The optimal value and the final wealth obtained by solving the portfolio selection problem based on the reduced scenario tree gotten with Algorithm 4 or Algorithm 6 are quite close to those values obtained under the original tree, the corresponding deviations are always less than 1%. Moreover, the solution time decreases a lot compared to the solution time under the original tree. These results demonstrate that both Algorithm 4 and Algorithm 6 are practical and efficient methods for the reduction of multi-stage scenario trees and the solution of large-scale multi-stage portfolio selection problems, of course, dynamic stochastic programming problems based on the scenario tree, we believe.

ST	RID	OV	V-error	FW	W-error	Time(s)	Percent
1	(891,57,0,$\cdots$,0,14,38)	-3005.0	0	1002.4	0	77.7	100%
1-4	(895,65,0,$\cdots$,0,11,30)	-3007.7	0.09%	1004.8	0.24%	2.5	3.2%
1-6	(895,65,0,$\cdots$,0,11,30)	-3006.5	0.05%	1003.3	0.09%	1.1	1.4%
1-S	(644,41,0,$\cdots$,0,49,266)	-3018.1	0.44%	1017.4	1.50%	2.5	3.2%
2	(886,47,0,$\cdots$,0,15,51)	-3005.0	0	1002.4	0	124.7	100%
2-4	(884,44,0,$\cdots$,0,18,55)	-3008.7	0.12%	1004.9	0.25%	3.2	2.6%
2-6	(875,51,0,$\cdots$,0,16,59)	-3007.3	0.08%	1004.4	0.20%	1.7	1.4%
2-S	(750,103,0,$\cdots$,0,14,135)	-3016.9	0.40%	1017.7	1.52%	2.9	2.3%
3	(447,180,0,$\cdots$,0,46,328)	-2999.3	0	983.8	0	18.8	100%
3-4	(524,178,0,$\cdots$,0,20,279)	-3005.5	0.21%	983.8	0%	1.7	9.3%
3-6	(449,250,0,$\cdots$,0,24,277)	-3004.2	0.17%	989.0	0.53%	1.8	9.4%
3-S	(767,34,0,$\cdots$,0,83,0,0,116,0)	-3018.1	0.63%	1051.8	6.90%	1.8	9.7%
4	(594,210,0,$\cdots$,0,27,170)	-2998.6	0	1010.3	0	22.4	100%
4-4	(656,172,0,$\cdots$,0,3,169)	-3002.5	0.13%	1010.6	0.03%	2.6	11.2%
4-6	(616,179,0,$\cdots$,0,39,166)	-3001.4	0.09%	1011.5	0.12%	2.2	9.9%
4-S	(920,35,0,$\cdots$,0,44,0)	-3022.9	0.81%	1026.8	1.63%	2.3	10.4%
5	(550,153,0,$\cdots$,0,59,238)	-2998.5	0	992.4	0	28.3	100%
5-4	(680,120,0,$\cdots$,0,37,164)	-3002.2	0.12%	999.9	0.76%	4.0	14.2%
5-6	(625,136,$\cdots$,0,34,206)	-3001.2	0.09%	993.7	0.13%	3.5	12.4%
5-S	(887,67,0,$\cdots$,0,73,0)	-3016.9	0.61%	1028.4	3.62%	4.8	16.9%
6	(799,77,0,$\cdots$,0,36,88)	-2996.7	0	1006.0	0	12625	100%
6-1-4	(779,85,0,$\cdots$,0,39,97)	-2997.1	0.01%	1006.5	0.05%	6455	51.1%
6-2-4	(765,92,0,$\cdots$,0,41,102)	-2997.3	0.02%	1007.0	0.10%	3048	24.1%
6-3-4	(717,111,0,$\cdots$,0,48,124)	-2997.7	0.04%	1008.0	0.20%	1348	10.7%
6-4-4	(553,221,$\cdots$,0,51,175)	-2998.7	0.07%	1011.0	0.50%	589	4.7%
6-5-4	(331,318,0,$\cdots$,0,22,329)	-3000.0	0.11%	1013.4	0.74%	246	1.9%
6-1-6	(786,81,0,$\cdots$,0,39,94)	-2997.1	0.01%	1006.5	0.05%	2845	22.5%
6-2-6	(784,82,0,$\cdots$,0,40,95)	-2997.5	0.03%	1006.8	0.08%	1396	11.1%
6-3-6	(779,85,0,$\cdots$,0,40,97)	-2998.2	0.05%	1007.3	0.13%	661	5.2%
6-4-6	(759,93,0,$\cdots$,0,44,104)	-2999.0	0.08%	1008.0	0.20%	310	2.5%
6-5-6	(769,88,0,$\cdots$,0,43,100)	-3000.0	0.11%	1008.4	0.24%	163	1.3%

Table 8: Solution results of the multi-stage portfolio selection problem based on different scenario trees

6 Conclusion

To improve the implementability and computational efficiency of current scenario tree reduction algorithms, we introduce a new methodology which allows for changing node values, and a simple and easy-to-calculate distance function to measure the difference between two scenario trees. A primitive scenario tree reduction model is constructed based on minimizing the new distance. This model also minimizes the Wasserstein distance between the reduced tree and the initial tree. To make it appropriate for the reduction of general multi-stage scenario trees, we construct an improved model based on the primitive model. This advanced model not only performs well for the multi-stage scenario tree reduction, but also is supported theoretically by the stability results of stochastic programs. Then we design a stage-wise algorithm to solve the improved model, thus to reduce multi-stage scenario trees, and the algorithm is superior to the simultaneous backward reduction method in terms of computational complexity. We further design a multi-stage scenario tree reduction algorithm with a pre-specified distance by utilizing the stability results of stochastic programs.

The proposed algorithms are tested through a series of numerical experiments with real financial data as well as the solution of multi-stage portfolio selection problems. The results demonstrate that both the stage-wise algorithm, Algorithm 4, and the multi-stage scenario tree reduction algorithm with a pre-specified distance, Algorithm 6, perform much better than SBRM, especially when they are used for the reduction of general multi-stage scenario trees and the solution of large-scale multi-stage portfolio selection problems.

In this paper, we develop Algorithm 4 through designing two approximate methods to solve the model (11) and it runs stage-wisely. This might be the reason that Algorithm 6 can generate a smaller scenario tree than Algorithm 4 with the same distance. Therefore, an interesting topic is how to design a "global" or more efficient algorithm to solve the model (11), which is important for further improving the overall performance of Algorithm 4. This is left for future research.

Acknowledgements

This research was supported by the National Natural Science Foundation of China (Grant Numbers 71371152 and 11571270).

References

[1] Høyland, K., Wallace S.W.: Generating scenario trees for multistage decision problems. Manage. Sci. 47(2), 295-307 (2001)

[2] Dupačová J., Gröwe-Kuska N., Römisch W.: Scenario reduction in stochastic programming: an approach using probability metrics. Math. Program. 95(3), 493-511 (2003)

[3] Arthur D., Vassilvitskii S.: K-means++: the advantages of careful seeding. Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms, New Orleans, Louisiana, USA, January 7-9, 1027-1035 (2007)

[4] Heitsch H., Römisch W.: Scenario reduction algorithms in stochastic programming. Comput. Optim. Appl. 24(2), 187-206 (2003)

[5] Heitsch H., Römisch W.: A note on scenario reduction for two-stage stochastic programs. Oper. Res. Lett. 35(6), 731-738 (2007)

[6] Heitsch H., Römisch W.: Scenario tree modeling for multistage stochastic programs. Math. Program. 118(2), 371-406 (2009)

[7] Heitsch H., Römisch W., Strugarek C.: Stability of multistage stochastic programs. SIAM J. Optim. 17(2), 511-525 (2006)

[8] Henrion R., Küchler C., Römisch W.: Scenario reduction in stochastic programming with respect to discrepancy distances. Computational Comput. Optim. Appl. 43(1), 67-93 (2009)

[9] Henrion R., Küchler C., Römisch W.: Discrepancy distances and scenario reduction in two-stage stochastic mixed-integer programming. J. Ind. Manag. Optim. 4(2), 363-384 (2008)

[10] Heitsch H., Römisch W.: Scenario tree reduction for multistage stochastic programs. Comput. Manag. Sci. 6(2), 117-133 (2009)

[11] Armstrong M., Ndiaye A., Razanatsimba R., Galli A.: Scenario reduction applied to geostatistical simulations. Math. Geosci. 45(2), 165-182 (2013)

[12] Pflug G. Ch.: Version-independence and nested distributions in multistage stochastic optimization. SIAM J. Optim. 20(3), 1406-1420 (2009)

[13] Pflug G. Ch., Pichler A.: A distance for multistage stochastic optimization models. SIAM J. Optim. 22(1), 1-23 (2012)

[14] Pflug G. Ch., Pichler A.: Dynamic generation of scenario trees. Comput. Optim. Appl. 62(3), 641-668 (2015)

[15] Kovacevic R. M., Pichler A.: Tree approximation for discrete time stochastic processes - a process distance approach. Ann. Oper. Res. 1, 1-27 (2015)

[16] Houda M.: Probability metrics and the stability of stochastic programs with recourse. Bull. Czech Econom. Soc. 9(17), 65-77 (2002)

[17] Römisch W., Schultz R.: Stability analysis for stochastic programs. Ann. Oper. Res. 30(1), 241-266 (1991)

[18] Graf S., Luschgy H.: Foundations of Quantization for Probability Distributions. Volume 1730 of Lecture Notes in Math., Springer-Verlag Berlin Heidelberg (2000)

[19] MacQueen J.: Some methods for classification and analysis of multivariate observations. Proceedings of the fifth Berkeley symposium on mathematical statistics and probability, 1(14), 281-297 (1967)

[20] Xu D., Chen Z., Yang L. Scenario tree generation approaches using K-means and LP moment matching methods. J. Comput. Appl. Math. 236(17), 4561-4579 (2012)

[21] Chen Z., Xu D.: Knowledge-based scenario tree generation methods and application in multiperiod portfolio selection problem. Appl. Stoch. Model. Bus. 30(3), 240-257 (2014)

Appendix

In Algorithm 2, when $r = 2$, we have

$$\tilde{\omega}_m^{(n+1)} = \frac{\sum_{j\in\widetilde{I}_m^{(n)}} p_j\omega_j}{\sum_{j\in\widetilde{I}_m^{(n)}} p_j}. \tag{28}$$

It holds that

$$\frac{p_i\omega_i + p_j\omega_j + p_k\omega_k}{p_i + p_j + p_k} = \frac{p_{ij}\omega_{ij} + p_k\omega_k}{p_{ij} + p_k}, \tag{29}$$

where $p_{ij} = p_i + p_j$ and $\omega_{ij} = \frac{p_i\omega_i+p_j\omega_j}{p_i+p_j}$.

According to (16), (17), (19) and (20), an implementation of Algorithm 2 when $r = 2$ is as follows.

Algorithm 2(a): Implementation of Algorithm 2 when $r = 2$

Step 1 Initialization. Given the input of the problem (3): ω_i and p_i, $i = 1, 2, \cdots, N$. Set $n = N$. Let $\widetilde{I}_m^{(n)} = \{m\}$, $\tilde{\omega}_m^{(n)} = \omega_m$ and $\tilde{p}_m^{(n)} = p_m$ for $m = 1, 2, \cdots, n$. Calculate the matrix $D^{(n)} = (D_{ij})_{n\times n}$ where

$$D_{ij}^{(n)} = \frac{\tilde{p}_i^{(n)}\tilde{p}_j^{(n)}}{\tilde{p}_i^{(n)} + \tilde{p}_j^{(n)}} \parallel \tilde{\omega}_i^{(n)} - \tilde{\omega}_j^{(n)} \parallel^2 .$$

Step 2 Find the smallest non-diagonal element of $D^{(n)}$, denoted as D_{ij}. Let

$$(\tilde{\omega}_1^{(n-1)}, \cdots, \tilde{\omega}_{n-2}^{(n-1)}) = (\tilde{\omega}_1^{(n)}, \cdots, \tilde{\omega}_{i-1}^{(n)}, \tilde{\omega}_{i+1}^{(n)}, \cdots, \tilde{\omega}_{j-1}^{(n)}, \tilde{\omega}_{j+1}^{(n)}, \cdots, \tilde{\omega}_n^{(n)}),$$

$$(\tilde{p}_1^{(n-1)}, \cdots, \tilde{p}_{n-2}^{(n-1)}) = (\tilde{p}_1^{(n)}, \cdots, \tilde{p}_{i-1}^{(n)}, \tilde{p}_{i+1}^{(n)}, \cdots, \tilde{p}_{j-1}^{(n)}, \tilde{p}_{j+1}^{(n)}, \cdots, \tilde{p}_n^{(n)}),$$

$$\tilde{p}_{n-1}^{(n-1)} = \tilde{p}_i^{(n)} + \tilde{p}_j^{(n)}$$

and

$$\tilde{\omega}_{n-1}^{(n-1)} = \frac{\tilde{p}_i^{(n)}\tilde{\omega}_i^{(n)} + \tilde{p}_j^{(n)}\tilde{\omega}_j^{(n)}}{\tilde{p}_i^{(n)} + \tilde{p}_j^{(n)}}.$$

Delete the ith and jth rows and ith and jth columns of $D^{(n)}$ to obtain the matrix $D_{(n-1)}$, here

$$D_{n-1,i}^{(n-1)} = D_{i,n-1}^{(n-1)} = \frac{\tilde{p}_{n-1}^{(n-1)}\tilde{p}_i^{(n-1)}}{\tilde{p}_{n-1}^{(n-1)} + \tilde{p}_i^{(n-1)}} \parallel \tilde{\omega}_{n-1}^{(n-1)} - \tilde{\omega}_i^{(n-1)} \parallel^2, \ i = 1, 2, \cdots, n-1.$$

Step 3 Let $n = n - 1$. If $n > \widetilde{N}$, go to Step 2. Otherwise, stop.

With the above detailed descriptions, we can easily see that the number of multiplications in Algorithm 2(a) is

$$4N^2 + \sum_{k=\widetilde{N}+1}^{N} (3 + 4(k-1)) = 6N^2 + N - 2n^2 - n.$$