Markov Chain-based Policies for Multi-stage Stochastic Integer Linear Programming with an Application to Disaster Relief Logistics

Margarita Castro, Merve Bodur, Yongjia Song

Appendix A: Examples

EXAMPLE 1. Consider a simple MC $\mathcal{M}$ with two MC states representing the intensity of a hurricane: low-intensity ○ and high-intensity ●. Assume the transition probability between any pair of states is strictly positive. Consider an MSILP (P) for the hurricane disaster relief problem defined over $\mathcal{M}$, where the integer state variable z_n represents the option of activating an emergency alert in each node of the scenario tree. The first drawing in Figure 3 depicts a scenario tree for $\mathcal{M}$ starting at a low-intensity state and representing all possible MC states for $T = 4$ stages. The remaining drawings illustrate the effect of node aggregation for transformations `HN`, `MA`, and `MM` (we omit `PM` because in this example the MC state has only one attribute: intensity). The `HN` tree shows that all nodes in the same stage will share the same set of (aggregated) integer state variables z^A (i.e., they have the same color and shape), meaning that the emergency alert decision is the same for these nodes regardless of their corresponding hurricane intensity. In contrast, `MA` aggregates integer state variables in nodes with the same stage and hurricane intensity (i.e., the emergency alert decision depends on the current hurricane intensity), resulting in nodes of two different colors in each stage. For instance, the transformation for node $n \equiv (t, m^t) = (3, ○●○) \in \mathcal{N}_3$ only considers the low-intensity state in the third stage (i.e., $\Phi_3 m^3(○●○) = m_3(○) =$ ⬟), thus, $z^A_{3,⬟}$ represents its associated aggregated variables. Finally, nodes in each stage of the `MM` tree have up to four different colors, resulting from an aggregation based on the MC states in the current and previous stage (i.e., the model considers the current and previous hurricane intensity for the emergency alert decision). For instance, $z^A_{3,⬟}$ is associated to node (3,○●●) because the transformation considers the last two high-intensity states (i.e., $\Phi_3 m^3(○●●) = (m_2(●), m_3(●)) =$ ⬟). Overall, `HN` contains four sets of z^A variables (i.e., one for each stage), `MA` contains seven, `MM` contains eleven, and `FH` variant contains 15 sets of z^A variables (i.e., one for each node of the scenario tree). □

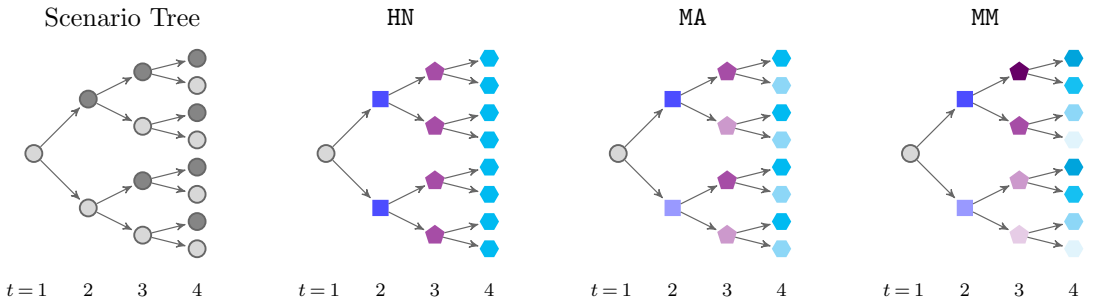

Figure 3 **Example showing nodes with the same aggregated integer state variables for transformations `HN`, `MA`, and `MM`. Nodes with the same color and shape share the same set of integer state variables.**

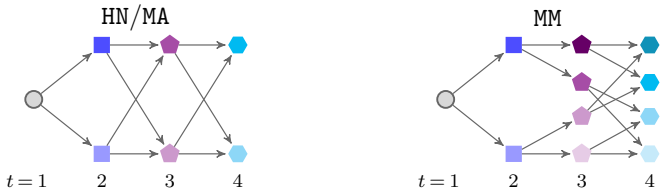

Figure 4 **SDDP policy graphs and subproblems for various types of transformations, `HN`, `MA`, and `MM`.**

EXAMPLE 2. Recall Example 1 where we define a MC $\mathcal{M}$ with two MC states: low-intensity state ○ and high-intensity state ●. Figure 4 illustrates the policy graphs for the `HN`, `MA`, and `MM` transformations based on the aggregations depicted in Figure 3. Since, at a minimum, we should have one SDDP subproblem for each MC state, we have at least two SDDP subproblems defined in each stage $t \geq 2$. In the case of `MA`, since nodes with the same MC state share the same set of aggregated variables, we have exactly two SDDP subproblems per stage (i.e., one per MC state) and the policy graph only depends on the MC transitions (i.e., a Markovian policy graph). The same applies for `HN`, although we only have one set of aggregated variables per stage. Since their corresponding policy graphs are identical, we use `HN/MA` to denote the first policy graph in Figure 4. On the other hand, the SDDP subproblem structure for `MM` is quite different: nodes with the same MC state can be associated with different aggregated variables, because the `MM` transformation depends not only on the MC state in the current node but also its parent node. For example, nodes (3,○●●) and (3,○○●) are associated with the same stage and MC state (m_3(○●●) = m_3(○○●) = ●, see Figure 3), but since the MC states of their parent nodes differ, they have different sets of aggregated variables (i.e., $z^A_{3,\bullet}$ and $z^A_{3,\bigcirc}$, respectively). Therefore, `MM` has one SDDP subproblem for each pair of MC states, corresponding to both the current node and the parent node for stages $t \geq 3$. The policy graph for `MM` can be seen as an aggregated version of its scenario tree (see Figure 3) with one copy for each type of nodes. □

EXAMPLE 3. Figure 5 illustrates the main components of our SDDP integrated B&C algorithm for the `MA` transformation for Examples 1 and 2. Here, an SDDP subproblem is defined for each MC state in each stage $t > 1$. The two illustrations show the decomposition structure over the policy graph: the root node ○ corresponds to the master problem while other nodes correspond to the SDDP subproblems. The left drawing depicts the forward pass of Algorithm 1 for scenario ○ → ● → ● → ●. Note that the solution of the SDDP master problem $(\widehat{x}_{\bigcirc}, \widehat{z}^A)$ is used as part of the input parameters to ■ and, similarly $(\widehat{x}_{\blacksquare}, \widehat{\zeta}_{\blacksquare})$ is an input to ⬟. Recall that the ζ variables are just the copies of z^A, so $\widehat{z}^A = \widehat{\zeta}_{\blacksquare} = \widehat{\zeta}_{\bullet}$. The drawing on the right illustrates the backward pass where the dual optimal solution of an SDDP subproblem is used to generate cuts to improve the outer-approximation of the cost-to-go function defined at its parent subproblems. For example, the optimal dual solution of ⬟ is used to generate cuts for subproblems ■ and ■. □

Figure 5 **SDDP integrated B&C algorithm: decomposition structure and two subroutines.**

Appendix B: B&C Integrated with the SDDP algorithm

Algorithm 2 describes the B&C procedure. The procedure receives as input a few parameters used by an SDDP sub-routine to be explained in detail later. Note that lines 4-7 correspond to the standard B&B algorithm of commercial solvers.

There are a few important points to mention about this procedure. First, we call the SDDP sub-routine for each child of the root node r, in which we perform the SDDP forward pass only considering sample paths that contain that specific child. Also, we look for cuts for all children nodes of the root node r before re-solving the current B&B node relaxation. An alternative would be to re-solve the B&B node relaxation whenever we find a violated cut, but this option showed worse performance in our preliminary experiments. Lastly, all cuts added to enhance the outer approximation of the cost-to-go functions associated with the SDDP subproblems in the previous iterations are automatically carried over to the current iteration.

We also note that this B&C procedure iterates multiple times (i.e., lines 9-15) for a given master solution, mainly due to the nature of SDDP. Since we consider a multi-stage problem, a first iteration of SDDP can add cuts in sub-problems at later stages without adding any cuts in the earlier stages, in particular, the master problem. To guarantee convergence and the propagation of these later-stage cuts, we need multiple iterations of SDDP. By doing so, we are guaranteed that the information propagates to the earlier stages and a cut is indeed created for the master problem.

Appendix C: Benders Decomposition for the 2SLDR Model

In this section we present a Benders decomposition approach to solve (P^L). We consider a Benders master problem that handles the first-stage decisions and decompose the second-stage problem into one Benders subproblem per node. Because the number of subproblems can be exponentially many (e.g., when `LDR-TH` is used), we consider an aggregated approach where we have one variable $\theta^L_{t,m}$ that approximates all the cost-to-go functions associated with nodes in stage t and with MC state $m \in \mathcal{M}_t$. This variant can be seen

Algorithm 2 A B&C framework integrated with the SDDP algorithm

1: **procedure** `SDDP-B&C`($\texttt{exact}, K, \epsilon$)
2: Initialize B&B search for $\overline{Q}^A_{\mathsf{r}}$ (master problem) and the `SDDP` environment.
3: Initialize $\mathcal{B}_n := \emptyset$ for all $n \in \mathcal{C}(\mathsf{r})$ and `cutAdded` := **false**.
4: **repeat**
5: Choose a B&B node from its list of open nodes. Solve the associated node relaxation problem.
6: **if** the node is infeasible or gives a relaxation bound worse than the incumbent solution **then**
7: Prune this B&B node.
8: **if** $\widehat{z}^A$ of the current solution $(\widehat{x}_{\mathsf{r}}, \widehat{z}^A, \widehat{\theta})$ is an integer vector **then**
9: **repeat** %% Cutting plane subroutine
10: `cutAdded` := **false**.
11: **for** $n' \in \mathcal{C}(\mathsf{r})$ **do** %% Find cuts parameters for each child subproblem
12: Get cut coefficients $(\alpha_{n'}, \beta_{n'}, \gamma_{n'})$:= `SDDP`$((\widehat{x}_{\mathsf{r}}, \widehat{z}^A, \widehat{\theta}), n', \epsilon, K, \texttt{exact})$.
13: **if** Cut with coefficients $(\alpha_{n'}, \beta_{n'}, \gamma_{n'})$ is violated by $(\widehat{x}_{\mathsf{r}}, \widehat{z}^A, \widehat{\theta})$ **then**
14: $\mathcal{B}_{n'} := \mathcal{B}_{n'} \cup (\alpha_{n'}, \beta_{n'}, \gamma_{n'})$, `cutAdded` := **true**.
15: **until** `cutAdded` = **false**.
16: **else**
17: Branch on the current B&B node and update the list of open nodes.
18: **until** The list of open B&B nodes is empty.

as a hybrid between the single-cut version (i.e., using a single θ variable) and the multi-cut version (i.e., using one $\theta^L_{t,n}$ variable for each subproblem) of Benders decomposition (Van Slyke and Wets 1969, Birge and Louveaux 1988). Our preliminary experiments show that our Benders variant performs the best on our test instances, however, other variants of Benders decomposition and computational enhancements can also be considered (see, e.g., Zverovich et al. (2012) and Bodur et al. (2017)).

In what follows we present the details of our Benders decomposition for `LDR-TH` (the other two cases, `LDR-T` and `LDR-M` are similar). The Benders master problem is given by

$$
\begin{aligned}
\overline{Q}^L_{\mathsf{r}} = \min \; & \sum_{n\in\mathcal{N}} p_n \left(c_n^\top z^A_{\phi_t(n)} + d_n^\top(\mu_{t(n)}^\top \xi_n^{t(n)})\right) + h_{\mathsf{r}}^\top y_{\mathsf{r}} + \sum_{t=2}^{T}\sum_{m\in\mathcal{M}_t} \theta^L_{t,m} \\
& \text{s.t. } (6a)-(6c), \\
& \theta^L_{t,m} \geq {\alpha^L_{t,m}}^\top \mu + {\beta^L_{t,m}}^\top z^A + \gamma^L_{t,m}, \quad \forall (\alpha^L_{t,m},\beta^L_{t,m},\gamma^L_{t,m}) \in \mathcal{B}^L_{t,m},\ m\in\mathcal{M}_t,\ t\in\{2,...,T\}, \qquad (9) \\
& (y_{\mathsf{r}},\mu,z^A) \in \mathbb{R}^r \times \mathbb{R}^{k\times\sum_{t\in[T]} l^t} \times \mathbb{Z}^{\ell\cdot\sum_{t\in[T]} q_t},
\end{aligned}
$$

where $\theta^L \in \mathbb{R}^{\sum_{t=2}^T |\mathcal{M}_t|}$ is introduced to represent the cost-to-go function approximation for each Markovian state in all remaining stages (except for the first stage) via constraints (9). Here $\mathcal{B}^L_{t,m}$ stores the coefficients associated with the Benders cuts for each MC state $m \in \mathcal{M}_t$.

Algorithm 3 describes our Benders decomposition for (P^L). The procedure is a B&C algorithm similar to the one described in Algorithm 2. Once an integer solution is found during search, we enter the cutting plane subroutine. The algorithm iterates over each stage and MC state and its corresponding nodes, where $\mathcal{N}_t(m)$ is the set of all nodes in stage t associated with MC state $m \in \mathcal{M}_t$. It then evaluates the Benders subproblems for all the nodes associated with a particular m and t and checks if the current approximation $\widehat{\theta}^L_{t,m}$ is close enough to the actual value $\widehat{Q}^L_{t,m}$ for a tolerance ϵ. If the approximation is not good enough, then it returns a cut and stops evaluating other subproblems. Similar to our SDDP variant, we add cuts to each candidate integer solution until no more cuts are found. The complete procedure ends when there are no more nodes to explore in the B&B search tree, i.e., we have found an optimal solution or proven infeasibility.

Algorithm 3 Benders Decomposition

procedure BENDERS(ϵ)
  Initialize B&B search for $\overline{Q}^L_{\mathsf{r}}$ (master problem)
  repeat
    Choose a node B&B from its node list. Solve the associated node relaxation problem.
    if the node is infeasible or gives a relaxation bound worse than the incumbent solution **then**
      Prune this B&B node.
    if $\widehat{z}^A$ of the current solution $(\widehat{\mu},\widehat{z}^A,\widehat{\theta})$ is an integer vector **then**
      for $m\in\mathcal{M}_t$ and $t\in\{2,...,T\}$ **do**
        for $n\in\mathcal{N}_t(m)$ **do**
          Solve $\widehat{Q}^L_n := Q^L_n(\widehat{\mu},\widehat{z}^A)$ and save dual solutions π^n.
        Save value $\widehat{Q}^L_{t,m} := \sum_{n\in\mathcal{N}_t(m)} p_n \widehat{Q}^L_n$.
        if $|\widehat{Q}^L_{t,m} - \widehat{\theta}^L_{t,m}| \geq \epsilon |\widehat{Q}^L_{t,m}|$ **then** %% Add cut if needed
          Add inequality (9) and exit the for loop.
    else
      Branch on the current B&B node and update list of nodes.
  until No more nodes in the B&B node list.

The cuts added to the master problem (9) are Benders cuts over all the nodes associated to a given stage and MC state. Specifically, the cut coefficients for $t \in \{2, ..., T\}$ and $m \in \mathcal{M}_t$ are

$$\alpha_m^L = \sum_{n \in \mathcal{N}_t(m)} p_n \pi^{n\top} A_n, \qquad \beta_m^L = \sum_{n \in \mathcal{N}_t(m)} p_n \pi^{n\top} B_n,$$
$$\gamma_m^L = \sum_{n \in \mathcal{N}_t(m)} p_n \left(\widehat{Q}_n^L - (\pi^{n\top} A_n)^\top \widehat{\mu}_{t(n)-1} - (\pi^{n\top} B_n)^\top \widehat{z}^A_{\phi_t(a(n))} \right),$$

where π^n are the dual solutions associated with the cost-to-go function $Q_n^L(\cdot, \cdot)$. Each coefficient corresponds to a weighted sum of the associated coefficients in the subproblems. Also, the algorithm generates Benders feasibility cuts when necessary, which is common in 2SLDR approximations (Bodur and Luedtke 2022).

Appendix D: Aggregated Model for HDR Applicaton

We now present how to transform the MSILP (*HDR*) with integer state variables at each node of the scenario tree to an MSILP with integer variables only at the root node, following the ideas presented in Section 3 and Section 4. The general idea is to consider all integer state variables as first-stage variables and apply a suitable transformation Φ_t to aggregate the number of variables to a manageable size. In what follows, we present the resulting aggregated model (*HDR*^{A}) for a generic transformation and then show four alternatives given the specific structure of (*HDR*).

Following the notation introduced in Section 3, we consider z^A to be the set of aggregated integer state variables for each node. There are two aspects to consider when applying this transformation. First, constraints (8e) and (8f) are now first-stage constraints. Second, we can omit state variables x_n^{c} because these variables are only used to model the capacity dependency from one stage to the next, which is fully dictated by the first-stage variables z^A. The resulting aggregated model is:

$$\min \sum_{j \in \mathcal{J}} \left(g_j x^{\mathrm{I}}_{\mathsf{r}j} + q_{\mathsf{r}j} v_j + \sum_{i \in \mathcal{I}} f_{\mathsf{r}ij} y_{ij} \right) + \sum_{i \in \mathcal{I}} b_i w_i + \sum_{n \in \mathcal{N}} \sum_{\ell \in \mathcal{L}} z^A_{\phi_t(n)\ell} c_\ell + \sum_{n \in \mathcal{C}(\mathsf{r})} \bar{p}_{\mathsf{r}n} Q_n^A(x^I_{\mathsf{r}}, z^A) \qquad (HDR^A)$$
$$\text{s.t. } (7a) - (7b)$$
$$v_j \le C_j, \qquad \forall j \in \mathcal{J}, \qquad (10a)$$
$$\sum_{\ell \in \mathcal{L}_l} z^A_{\phi_t(n)\ell} \le 1, \qquad \forall l \in [L],\, n \in \mathcal{N}, \qquad (10b)$$
$$z^A_{\phi_{t-1}(a(n))\ell} \le z^A_{\phi_t(n)\ell}, \qquad \forall \ell \in \mathcal{L}, n \in \mathcal{N} \setminus \{\mathsf{r}\}, \qquad (10c)$$
$$x^{\mathrm{I}}_{\mathsf{r}j}, v_j, w_i, y_{ij} \ge 0,\ z^A_{\phi_t(n)\ell} \in \{0,1\}, \qquad \forall j \in \mathcal{J},\, i \in \mathcal{I}, \ell \in \mathcal{L}, n \in \mathcal{N} \setminus \{\mathsf{r}\},$$

where the cost-to-go function for a node $n \in \mathcal{N} \setminus \{\mathsf{r}\}$ is given by:

$$Q_n^A(x^I_{a(n)}, z^A) = \min \sum_{j \in \mathcal{J}} \left(g_j x^{\mathrm{I}}_{nj} + q_{nj} v_j + \sum_{i \in \mathcal{I}} f_{nij} y_{ij} \right) + \sum_{i \in \mathcal{I}} b_i w_i + \sum_{n' \in \mathcal{C}(n)} \bar{p}_{nn'} Q^A_{n'}(x^I_n, z^A)$$
$$\text{s.t. } (8a) - (8b)$$
$$v_j \le C_j + \sum_{n' \in \mathcal{P}(n)} \sum_{\ell \in \mathcal{L}} K_{j\ell} z^A_{\phi_t(n')\ell}, \qquad \forall j \in \mathcal{J}, \qquad (11a)$$
$$x^{\mathrm{I}}_{nj}, v_j, w_i, y_{ij} \ge 0, \qquad \forall j \in \mathcal{J},\, i \in \mathcal{I}.$$

Note that the right-hand side of constraint (11a) represents the current capacity of a DC by considering the modality activation and capacity increases at previous stages. Also, although constraints (10b) and (10c) are imposed for each node, the fact is that some of them might be identical because they share the same problem parameters and aggregated variables. For example, transformation `HN` only needs one set of constraints (10b) per stage because of the stage-based variable aggregation.

Appendix E: Problem Description and Instance Generation for HDR

E.1. Details on Synthetic Instance Generation

We now describe the main components of our HDR instance generation. We focus on the grid representation for the problem, the MC, and three main parameters (i.e., demand, modalities, and capacity). All other details about instances generation (e.g., cost parameters) can be found in our instances generation code. We will make the instance generation code and the set of instances publicly available upon publication.

Grid Representation and Locations of Shelters and DCs. Similar to previous works in the literature, we use a grid network to represent the potential locations of shelters and DCs, as well as possible locations of the hurricane (see Figure 6). The top row on the grid represents the land, where each cell contains a subset of shelters and DCs. The remaining cells in the grid correspond to the sea and are used to represent possible locations of the hurricane at any stage during the planning horizon. For simplicity, we assume that any shelter can be supplied by any DC in the network.

Land and sea cells have pre-defined dimensions: land cells have a size of 50×100, while sea cells have a size of 20×100. We randomly generate 3 to 7 shelters and 2 to 4 DCs in each land cell. Shelters and DCs are positioned uniformly at random inside their corresponding cells. The coordinates of the hurricane located in a cell are assumed to be the coordinates of the center of the cell.

MC Description. As described in the main text, each MC state corresponds to two attributes of the hurricane: location and intensity, that is, $m = (m^x, m^y, m^{int}) \in \mathcal{M}$ where m^x and m^y are the x- and y-coordinates of the hurricane's location, respectively, and m^{int} represents the hurricane's intensity level. The MC is given by two independent transition probability matrices: one for the hurricane intensity level and one for its location. The intensity level transition matrix is identical to the one used by Pacheco and Batta (2016), which considers six levels of intensity $\{0, 1, \ldots, 5\}$ where level 5 corresponds to the maximum hurricane

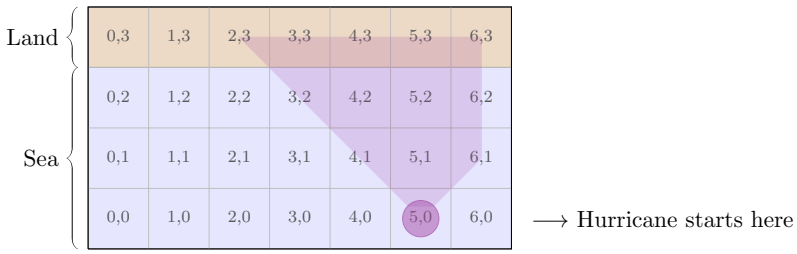

Figure 6 **Grid with hurricane's initial location $(m^x, m^y) = (5, 0)$ and possible future locations (i.e., shaded region).**

intensity and level 0 corresponds to the case when the hurricane dissipates. The following matrix shows the transition probabilities where P_{ij} represents the probability of transitioning from intensity level i to j.

$$P = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0.11 & 0.83 & 0.06 & 0 & 0.6 & 0 \\ 0.15 & 0.6 & 0.25 & 0 & 0 & \\ 0 & 0 & 0.04 & 0.68 & 0.28 & 0 \\ 0 & 1 & 0 & 0.18 & 0.79 & 0.03 \\ 0 & 0 & 0 & 0 & 0.5 & 0.5 \end{pmatrix}$$

For the hurricane movement, we assume that it originates at the bottom row of the grid (i.e., $m^y = 0$ for the initial MC state) and that m^y increases by one in each period (i.e., the hurricane advances upwards by one step in each period) and reaches land in exactly T periods. The hurricane can also move on the x-axis by either staying in the same x-coordinate, or moving one cell to the left or the right. To determine the transition probabilities of each movement, we assign random weights to each movement and normalize them. We assign a uniform weight between 30 and 40 for staying in the same x-coordinate and a weight between 20 and 40 for moving to the right and left cells. If the current x-coordinate is at one of the grid borders (e.g., $m^x = 0$), then the probability of stepping out of the grid is zero. As an example, consider a 4×5 grid, current hurricane location $(m^x, m^y) = (1, 1)$, and weights 25, 36, and 22, for moving to locations $(0, 2)$, $(1, 2)$, $(2, 2)$, respectively. Then, the transition probability of moving from location $(1, 1)$ to $(0, 2)$ is $\frac{25}{83} \approx 0.3$.

The hurricane movement transition probability matrix is generated at random for each instance. The initial x-coordinate is chosen uniformly at random considering all possible options. The initial intensity level is also generated with a uniform random distribution considering hurricane intensity levels of two or higher.

Demand Generation. The demand for each shelter depends on the current MC state of the hurricane (i.e., location and intensity). We consider a maximum demand parameter $d^{\max}$ and a maximum distance parameter $\delta^{\max}$ to model this dependency. The maximum demand $d^{\max}$ is chosen at random between values 1000 and 1500 for each instance and represents the maximum demand for all shelters in a cell. The maximum demand at a cell is randomly distributed across all shelters in that cell. Therefore, each shelter $j \in \mathcal{I}$ has a maximum demand $d_j^{\max}$ and the total maximum demand of all shelters in the same cell is equal to $d^{\max}$.

The maximum distance $\delta^{\max} = 150$ represents the distance of influence of the hurricane, i.e., all shelters which are farther than $\delta^{\max}$ from the hurricane have zero demand. If the distance between a shelter $i \in \mathcal{I}$ and the hurricane is less than $\delta^{\max}$, then the shelters demand for MC state $m = (m^x, m^y, m^{int}) \in \mathcal{M}$ is:

$$d_{i,m} = d_i^{\max} \left(1 - \frac{\delta_{m,i}}{\delta^{\max}}\right) \cdot \left(\frac{m^{int}}{5}\right)^2, \tag{12}$$

where $\delta_{m,i}$ is the Euclidean distance between the current location of the hurricane and the shelter. Thus, the demand generation is linear with respect to the hurricane location and quadratic with respect to the hurricane intensity. The demand is zero if the intensity level is 0 and that the maximum demand is achieved when the hurricane is on top of the shelter and has the highest intensity level (i.e., $\delta_{m,i} = 0$ and $m^{int} = 5$).

Initial Capacity, Contingency Modality Options, and Capacity Expansion. The initial capacity of a DC depends on both $d^{\max}$ and the initial capacity percentage defined in Section 6. The total initial capacity over all DCs in a cell is given by $C^{\text{ini}} = d^{\max} \cdot p$, where $p \in \{0.2, 0.25, 0.3\}$ is the initial capacity percentage. The total initial capacity of a cell C^{ini} is then distributed uniformly at random across all DCs in that cell.

Table 9 **Average number of variables and constraints for different aggregations over small-size instances.**

Small-size instances	HN	MA	PM	MM	FH
Continuous Variables	566,370	566,370	566,370	566,370	566,370
Integer Variables	80	1,120	6,720	16,528	41,760
Total Variables	566,450	567,490	573,090	582,898	608,130
Constraints	120,129	124,834	129,373	137,171	164,414
Large-size instances	**HN**	**MA**	**PM**	**MM**	**FH**
Continuous Variables	12,281,770	12,281,770	12,281,770	12,281,770	12,281,770
Integer Variables	120	2,300	13,800	46,800	733,240
Total Variables	12,281,890	12,284,070	12,295,570	12,328,570	13,015,010
Constraints	2,163,164	2,175,033	2,188,428	2,224,132	2,932,940

Thus, the initial capacity C_j of a particular DC $j \in \mathcal{J}$ is given by a portion of C^{ini}. Also, we consider the inventory capacity of a DC $j \in \mathcal{J}$ as 20% of the initial capacity, that is, $C_j^{\text{I}} = 0.2C_j$.

We generate different contingency modality options depending on the grid size (i.e., number of land cells) and the modality Type (i.e., either Type-1 or Type-2, see Section 6). Our modality set $\mathcal{L}$ considers options that increase DCs capacity in adjacent cells and options that increase the capacity for all DCs. For example, in a 4×5 grid we consider modality options that increase capacity for DCs in cells 0 and 1, 1 and 2, 2 and 3, and options that increase the capacity in all DCs, that is, a total of four different contingency modality options based on location. Since a center can be affected by more than one contingency option, we do not partition the modality set, that is, $L = 1$ and $\mathcal{L} = \mathcal{L}_L$.

In addition, we consider capacity increments of Type-1 or Type-2, each one with four different options: Type-1 has options $\{10\%, 20\%, 30\%, 40\%\}$ and Type-2 has options $\{15\%, 30\%, 45\%, 60\%\}$. Then each modality option in $\mathcal{L}$ corresponds to a combination of the affected DCs based on location and the capacity increase. Considering a 4×5 grid example with Type-1 modalities, a possible modality option is to increase the capacity of DCs in cells 2 and 3 by 20% increments. In this example, we have a total of 16 modality options (i.e., four location options and four increments options).

The capacity increments for each modality are based on the initial capacity of the DCs. For example, consider the 4×5 grid with Type-1 modalities and active modality $\ell \in \mathcal{L}$ affecting cells 2 and 3 by 20% increments. Then, a DC $j \in \mathcal{J}$ in cell 0 has no capacity increments given the current active modality (i.e., $K_{j\ell} = 0$), but a DC $j' \in \mathcal{J}$ in cell 2 with initial capacity $C_{j'} = 100$ has capacity increments $K_{j'\ell} = 20$.

Model Size for HDR Instances. Table 9 shows the average number of variables and constraints for both small-size (4×5 grid) and large-size (5×6 grid) instances.

E.2. Details on Case-study Instance Generation

In this section, we briefly describe the logistics network related input data for the case-study instance. Table 10, 11, and 12 provide the data for the DCs, shelters, and the set of eligible DCs to serve shelters, respectively, which are collected from (SCEMD 2023) as well as the authors' personal communication with officials at the SCEMD.

Demand. We adapt the demand function $d_{i,m}$ following equation (12) defined for the synthetic instances. Specifically, we set $\delta_{\max} = 250$, and set $d_i^{\max}$ to be 20% of the population of the corresponding county served by shelter $i \in \mathcal{I}$, which is obtained from the US Census 2020 data (see Table 11). Additionally, at each time t, the associated parameters $\delta_{m,i}$ are calculated based on the track forecast error associated with the hurricane's state at time t and the point forecast of the hurricane's location at its landfall time.

Table 10 Data for DCs in the case-study instance.

Facility	Address	Max. Capacity	Init. Capacity	Group
Florence Airport	2100 Terminal Dr, Florence, SC 29501	3.26×10^5	4.08×10^3	1
North AAF	270 Slab Landing Rd, North, SC 29112	6.80×10^5	8.50×10^3	2
Charleston Convention Center	5001 Coliseum Dr, North Charleston, SC 29418	1.70×10^5	2.13×10^3	3
I-26E (Harleyville)	495 Judge St, Harleyville, SC 29448	1.36×10^5	1.70×10^3	3
I-95S (Hendersonville)	2410 State Rd S-15-28, Walterboro, SC 29488	2.13×10^5	2.66×10^3	3
Farmers Market	3483 Charleston Hwy, West Columbia, SC 29172	1.70×10^5	2.13×10^3	2
SCEMD Warehouse	100 Tidewater Dr, Winnsboro, SC 29180	1.70×10^4	4.25×10^3	2

Table 11 Data for shelters used in the case-study instance.

Facility	County	Address	Demand
Andrews Elementary School	Georgetown	13072 County Line Road, Andrews, SC 29510	12681
Loris High School	Horry	301 Loris Lion Road, Loris, SC 29569	70206
Cane Bay High School	Berkeley	1624 State Rd. Summerville, SC 29483	45972
Stall High School	Charleston	3625 Ashley Phosphate Road, North Charleston, SC 29418	80635
Fort Dorchester High School	Dorchester	8500 Patriot Boulevard, N Charleston, SC 29420	31457
Colleton County High School	Colleton	150 Cougar Nation Dr. Walterboro, SC 29488	7682
Ridgeland School Complex	Jasper	250 Jaguar Trail, Ridgeland, SC 29936	6020

Table 12 Data for the service eligibility from DCs to shelters in the case-study instance.

Facility	Counties Served
Florence Airport	Georgetown, Horry, Berkeley, Charleston, Dorchester
North AAF	Berkeley, Charleston, Dorchester, Colleton, Jasper
Charleston Convention Center	Colleton, Jasper
I-26E (Harleyville)	Berkeley, Charleston, Dorchester, Colleton, Jasper
I-95S (Hendersonville)	Berkeley, Charleston, Dorchester, Colleton, Jasper
Farmers Market	Berkeley, Charleston, Dorchester
SCEMD Warehouse	All

Logistics costs. The logistics costs are specified as follows:

- The unit cost of transporting relief commodities from DC $j \in \mathcal{J}$ to shelter $i \in \mathcal{I}$ for each MC node $n \in \mathcal{N}$ is defined as $f_{nij} = \omega \times dist(\ell_i, \ell_j)$, where the base transportation cost $\omega = 0.0038$ (e.g., fuel cost per mile), and $dist(\ell_i, \ell_j)$ is a distance function that measures the distance between the two locations ℓ_i and ℓ_j, which we obtained from Google Map API.
- We consider four options for the contingency modalities, corresponding to a 10%, 20%, 40%, and 80% increase per period in the production capacity, with a cost factor of 1, 2, 3, and 4, respectively. In addition, we consider three contingency modality options based on groups of DC locations, where the group assignment is specified in Table 10. Then each modality option in $\mathcal{L}$ corresponds to a combination of the affected DCs based on location and the level of capacity increment per period, resulting in a total of $|\mathcal{L}| = 12$ modality options (three location options and four increment options). The contingency modality cost for each option $\ell \in \mathcal{L}$ is given by the sum of the initial DC capacities for each group (see Table 10), multiplied by the cost factor associated with the modality option, and then finally scaled by a constant 0.1.
- The unit production cost q_{nj} is set to be the base cost $\beta = 22.4$ at all DCs $j \in \mathcal{J}$ in each MC node $n \in \mathcal{N}$. The base cost is estimated based on the market price for commodities obtained in a typical disaster relief kit (Horry County Emergency Management Division 2024).
- The unit inventory cost g_j is set to be a constant for each DC $j \in \mathcal{J}$: $g_j = 0.01 \times \beta$, $\forall j \in \mathcal{J}$.
- The unit penalty cost for demand shortage is set to be a constant for each shelter $i \in \mathcal{I}$: $b_i = 50 \times \beta$, $\forall i \in \mathcal{I}$.

Markov Chain. We now briefly describe how we model the evolution of the hurricane as an MC is presented in (Bhattarai and Song 2024). We define each period as 12 hours, which is aligned with the intervals between forecast lead times in the National Hurricane Center's (NHC) forecast error database (National

Hurricane Center 2012). The initial forecast is chosen to be the one given at H-120, i.e., 120 hours before Hurricane Florence's landfall time. At this time, the track and intensity forecasts are based on the NHC's best available predictions at H-120. To account for forecast uncertainty, we apply a time series model trained based on historical forecast error data, covering both track and intensity errors over the past 20 years. Using this model, we generate sample paths of forecast errors, which, when combined with the initial forecast made at H-120, yield possible trajectories for the hurricane's path and intensity over the planning horizon. In constructing the MC, we assume that the hurricane's landfall time is deterministic, with the hurricane's forward velocity (along-track direction) remaining constant, while forecast errors are applied in the cross-track direction only, leading to a 11-period planning horizon (where landfall occurs at $t = 11$). The track error and intensity error are assumed to be independent in our numerical experiments, and this assumption is supported by the statistical analysis using real-world forecast error data from the NHC database (National Hurricane Center 2022). To construct the MC model, we apply an optimization-based discretization technique to identify the MC states and their associated transition probability matrix. This process is designed to align with the sample paths, using a predefined discretization threshold (as described in (Löhndorf and Shapiro 2019) and related works). Table 13 provides the number of MC states in each period for the MC used for the case study. We note, however, that the initial states for the instances used for the case study correspond to period 7 and 8 due to the computational challenge to handle the large MC state space.

Appendix F: Computational Effort for Exact Methods

F.1. Extensive Form vs SDDP integrated B&C

We now compare two solution methods for the aggregated models with four types of transformations, that is, solving the extensive form model directly with CPLEX (`Ex`) versus the B&C procedure integrated with the SDDP algorithm proposed in Section 4 (`S`). Table 14 shows the average solution time (for instances where optimal solutions are obtained within the time limit) and the average optimality gaps (for unsolved instances) for the small-size instances. We also include the results of solving `FH` with `Ex` as a point of reference. We note that `Ex` solves to optimality all the instances for all transformations except for 33 when solving `FH`, while `S` fails to solve one instance for `PM` and solves only one instance to optimality for `MM`.

From Table 14 we see a clear advantage of `Ex` over `S` in both computational time and number of instances solved within the time limit across all transformations and instance configurations. These results are mostly explained by the fact that approximately 90% of the computational time is spent inside the SDDP subroutine. One main factor is the number of SDDP subproblems for each transformation: `HN` and `MA` have on average 69 SDDP subproblems, `PM` has 144, and `MM` has 294, which explains the relatively poor `MM` performance.

The previous results may suggest that `Ex` requires less computational effort for small-size instances than `S`. However, large-size instances cannot even be loaded into the solver due to the large number of variables and constraints in the formulation, that is, approximately 16GB of memory after the presolved phase (see Appendix E.1 for further model size information), so `S` is our only resort for these instances. Table 15 shows

Table 13 Number of Markovian states at each period for the Hurricane Florence case study.

Periods	1	2	3	4	5	6	7	8	9	10	11
# of location states	1	2	2	3	4	4	5	6	7	9	10
# of intensity states	1	6	6	6	6	6	6	6	6	6	6
Total # of states	1	12	12	18	24	24	30	36	42	54	60

Table 14 Performance comparison between S and Ex for small-size instances (4×5 grid size).

		Average Time (sec)								Gap (%)		
		HN		MA		PM		MM		FH	PM	MM	FH
Modality	Cap.	Ex	S	Ex	S	Ex	S	Ex	S	Ex	S	S	Ex
Type-1	20%	**260**	1,221	**712**	5,969	**961**	4,553	**1,853**	-	1,402	-	14.9	-
	25%	**132**	195	**361**	1,313	**487**	2,760	**626**	-	3,974	-	3.5	0.8
	30%	**73**	102	**109**	272	**222**	3,380	**367**	-	-	-	2.1	0.4
Type-2	20%	**253**	1,138	**919**	5,880	**2,047**	13,224	**2,951**	20,699	2,546	19.9	17.8	0.9
	25%	**167**	140	**393**	1,324	**651**	3,635	**817**	-	-	-	6.2	0.8
	30%	**82**	102	**119**	205	**271**	4,347	**556**	-	-	-	2.3	0.6
Average		**161**	483	**436**	2,494	**773**	5,316	**1,195**	20,699	2,641	19.9	7.8	0.7

the number of instances solved to optimality, the number of instances where the algorithm found a feasible solution (but could not prove optimality), and the average optimality gap over instances with a feasible solution for S over large-size instances. We see that S performs quite well for transformations HN and MA solving several instances to optimality; however, it fails to even find any feasible solution for MM. As before, these results are explained mainly by a large number of SDDP subproblems, and for these large-size instances, HN and MA have on average 114 subproblems, PM has 249, and MM has 593.

F.2. SDDiP vs SDDP integrated B&C

Next, we compare the SDDP integrated B&C algorithm, S, with the SDDiP algorithm (Zou et al. 2019), denoted by SDDiP. We implement SDDiP using the state-of-the-art `sddp.jl` package (Dowson and Kapelevich 2021), where we use all the default algorithm parameters except the time limit. We note that we conducted preliminary experiments to test various other parameter settings, such as the multi-cut version and the exclusive use of the Lagrangian Duality handler, however, we have not observed performance improvements, hence opted for the default setting. Based on the arguments in Dowson and Kapelevich (2021) and as in Bodur and Luedtke (2022), we provide SDDiP, more specifically to its training phase, 50% more time than the time taken by S, to compensate for the fact that `sddp.jl` is implemented in Julia whereas S is implemented in C++. It is important to note that the simulation of the obtained candidate SDDiP policy can be time-consuming, nevertheless, we exclude this process from the time limit and instead report and discuss the simulation times separately. Also, we remark that while S is an exact method, SDDiP is not guaranteed to converge to a globally optimal policy due to the existence of continuous state variables, which need to be discretized into a set of binary state variables (if one wishes to achieve a globally optimal policy by SDDiP).

We consider the HN and MA policies, for which S could obtain optimal solutions for all of our small-size instances. To implement the HN policy in `sddp.jl`, we define all the binary here-and-now modality selection

Table 15 Performance of the SDDP integrated B&C algorithm in large-size instances (5×6 grid size).

		# Optimal				# Feasible				Opt. Gaps (%)			
Modality	Cap.	HN	MA	PM	MM	HN	MA	PM	MM	HN	MA	PM	MM
Type-1	20%	2	0	0	0	8	10	7	1	53.4	51.8	56.1	82.0
	25%	5	2	0	0	5	8	6	0	35.7	26.2	27.9	-
	30%	10	8	0	0	0	1	4	0	-	15.4	4.3	-
Type-2	20%	3	0	0	0	7	10	7	1	43.5	52.9	61.4	86.0
	25%	6	3	0	0	4	7	6	0	37.2	24.9	32.8	-
	30%	9	8	0	0	1	1	6	0	12.0	13.3	6.4	-
Total/Average		35	21	0	0	25	37	36	2	36.4	30.7	31.5	84.0

variables, namely $\{z_{t\ell}\}_{t\in[T],\ell\in\mathcal{L}}$, along with the constraints only involving them, namely $\sum_{\ell\in\mathcal{L}_l} z_{t\ell} \leq 1\ \forall t \in [T], l \in [L]$ and $z_{t-1,\ell} \leq z_{t\ell}\ \forall t \in [T]\setminus\{1\}, \ell \in \mathcal{L}$, as a part of the problem associated with the root node of the scenario tree. Moreover, we define their *continuous* copies for each of the remaining nodes of the tree as state variables, namely $\{z_{nt\ell}\}_{t\in[T],\ell\in\mathcal{L}}$ for node n, along with constraints $z_{nt\ell} = z_{\alpha(n),t\ell}$ for all $t \in [T], \ell \in \mathcal{L}$ to be able to pass them to the subproblems. To implement the `MA` policy, we employ a similar approach, defining binary $\{z_{t\ell m}\}_{t\in[T],\ell\in\mathcal{L},m\in\mathcal{M}_t}$ variables at the root-node problem along with their associated constraints and creating their *continuous* copies for the other nodes along with the state-copying constraints linking these variables to those from the parent node, again to have the required parametrization for the subproblems.

We experiment with the small-size instances. For each instance subclass, we test the first half of the randomly generated instances, i.e., five instances per subclass, leading to 30 instances in total. For simulating the policy obtained at the end of the SDDiP training, we use 10,000 and 5,000 replications for the `HN` and `MA` policies, respectively. In Table 16, we report the followings in order: the time in seconds spent for the policy generation (column "Train. time") and the evaluation of the obtained policy (column "Sim. time"), the total number of subproblems solved in the training phase (column "Total solves"), the percentage gap between the deterministic lower bound (LB) obtained at the end of the training phase and the true optimal value (column "LB gap"), and the percentage gap between the higher end of the upper bound confidence interval (UB) obtained at the end of the simulation phase and the true optimal value (column "UB gap"). More specifically, the gaps are calculated taking the optimal value (Opt) as the reference, as 100(Opt-LB)/Opt and 100(UB-Opt)/Opt. (We note that the obtained UB confidence intervals are quite tight, their half-width being 2.6% away from the mean of the interval, on average.)

We observe that for the `HN` policy, `SDDiP` can find exact lower bounds and near-optimal solutions, in a computational effort similar to `S`. However, for the `MA` policy, when the initial capacity level is higher, `SDDiP` lower (upper) bound is around 11% (27%) away from the true optimal value found by `S`. In the `MA` case, compared to the `HN` case, due to the increased size of the subproblems, fewer subproblems are solved in the training phase and the simulation times are significantly increased.

Appendix G: Impact of MC-based Aggregation and Propagation Effect

We now analyze the impacts of the MC-based aggregation on the resulting HDR solution policies. In particular, we study why the performance of `MA` policy is similar to `HN` in our test instances. First, recall that binary variables $z_{n\ell}$ represent whether contingency modality $\ell \in \mathcal{L}$ is active in node $n \in \mathcal{T}$, only one modality

Table 16 Performance of the SDDiP algorithm in small-size instances (4×5 grid size).

		HN					MA				
Modality	Cap.	Train. time	Sim. time	Total solves	LB gap	UB gap	Train. time	Sim. time	Total solves	LB gap	UB gap
Type-1	20%	1782	294	391716	0.00	2.34	7636	15329	74844	0.00	2.40
	25%	138	127	99824	0.00	4.06	674	5758	11372	13.06	13.89
	30%	158	46	108475	0.00	2.22	328	2630	6124	6.82	29.29
Type-2	20%	1269	143	342241	0.00	1.21	7852	15097	72900	0.00	2.71
	25%	140	97	100796	0.00	2.36	1204	3432	16816	16.12	20.31
	30%	157	48	104393	0.00	3.30	333	2393	5249	8.56	44.95
Average		607	126	191241	0.00	2.58	3002	7416	31166	7.43	18.93

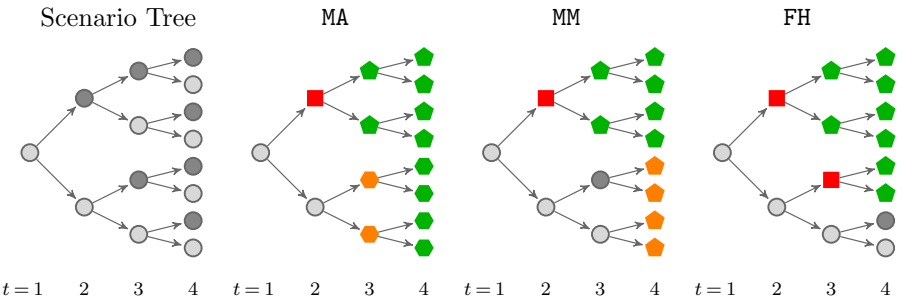

Figure 7 **Activation solutions over an abstract scenario tree for different MC-based transformations.**

can be active in each state (8e) and every active modality remains active in the following stages (8f). Thus, these constraints restrict the number of different modalities that can be active in any solution.

To understand the implications of an MC-based aggregation over nodes in the scenario tree with active modalities, Figure 7 illustrates solutions of different transformations in an abstract scenario tree with two states (from Example 1). Specifically, ■ are nodes where a modality is *initially activated* (as opposed to passively activated due to propagation, as explained below), ⬟ are nodes that have active modalities because of prior activation at their parent nodes, and ⬣ are nodes where the modality is *passively activated* (i.e., the activation is a consequence of the propagation caused by the transformation that forces certain nodes to have the same z values). For example, we can see that the `MA` policy initially activates a modality in $t = 2$ at the dark node. However, this causes the two ⬣ nodes to be passively activated because nodes with the same MC state must have the same z values according to transformation `MA`. This propagation, as a result of the MC-based aggregation, may cause an initial activation to have "unintended consequences" of passively activating nodes that may not necessarily benefit from the modality activation. This explains the similarity between `HN` and `MA` that we observe in Section 6.3. We can also observe such propagation behaviors with `PM` and `MM` but they occur less often since the aggregation depends on the MC state of the current stage and the previous one. In contrast, the activation when using `FH` does not have any such propagation effect since no aggregation is imposed (i.e., no ⬣ can be observed from `FH`).

Figure 8 illustrates the resulting policies for an instance in a series of images showing the grid of the problem where the hurricane originates at the bottom row and lands at the top row. The top grids show the percentage of nodes in the scenario tree with an active contingency modality for each cell. The bottom grids are heat-maps showing the proportion of nodes in a cell where a contingency modality is first activated (either initially or passively activated), and the color scheme illustrates the hurricane intensity (i.e., darker colors for higher intensity). Each row has a symbol ■ or ⬣, which distinguishes if the nodes are initially or passively activated, respectively. Note that `MA`, `PM`, and `MM` lead to a wait-and-activate type behavior by activating modalities in high-intensity nodes at later stages to reduce the number of active modalities in the last stage, a similar strategy to `FH`. In contrast, `HN` activates a modality in the first stage even though it may not be necessary for many nodes in later stages. We also find the intensity levels of ⬣ nodes tend to be much smaller than the ones associated with ■ nodes. This reflects the "unintended consequence" of forcing nodes of low intensity to be passively activated due to the restrictions imposed by the transformations.

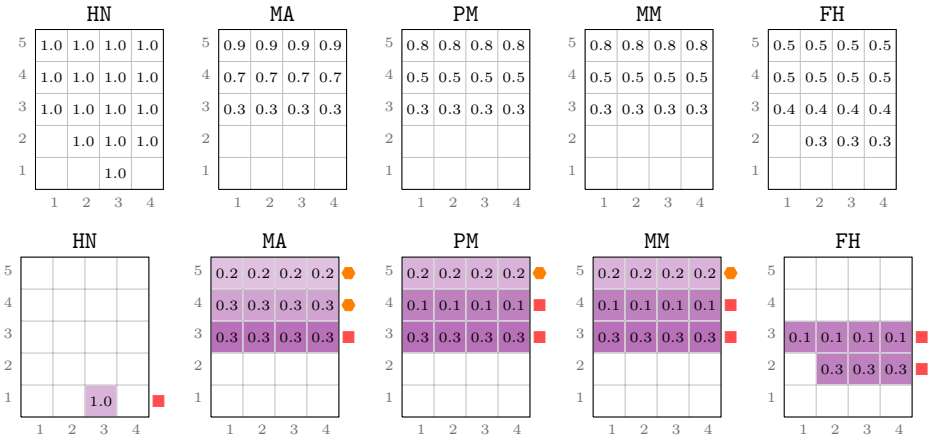

Figure 8 **Graphical representation of the solutions of an instance of Type-1 with 20% initial capacity.**

Appendix H: Additional Results for Approximation Methods

Tables 17 and 18 show the performance of the approximated methods based on 2SLDR and SDDP for small-size and large-size synthetic instances, and transformations HN, MA, and MM.

Table 17 **Solution time and quality of 2SLDR and SDDP bounds over small-size synthetic instances.**

			Average Time (sec)				Relative Difference (%)				
Trans.	Modality	Cap.	LDR-TH	LDR-T	LDR-M	S-LB	LDR-TH	LDR-T	LDR-M	S-UB	S-LB
HN	Type-1	20%	193.7	103.6	**71.7**	76.5	0.04	0.05	**0.00**	0.06	0.44
		25%	132.2	**31.3**	33.1	15.2	0.05	0.07	**0.00**	**0.00**	0.63
		30%	82.5	20.0	**19.0**	9.9	0.10	0.21	**0.00**	**0.00**	1.36
	Type-2	20%	262.8	**64.3**	76.0	61.6	0.03	0.05	**0.00**	0.03	0.46
		25%	131.7	**29.3**	35.8	12.4	0.05	0.07	**0.00**	0.01	0.62
		30%	80.4	19.7	**19.0**	10.0	0.10	0.21	**0.00**	**0.00**	1.36
	Average		147.2	44.7	**42.4**	30.9	0.06	0.11	**0.00**	0.02	0.81
MA	Type-1	20%	431.1	104.2	**95.2**	367.3	0.26	0.29	0.23	**0.08**	0.19
		25%	368.3	58.4	**49.2**	167.4	0.14	0.18	**0.11**	0.22	1.11
		30%	119.3	25.7	**25.4**	17.1	0.11	0.22	0.02	**0.00**	1.56
	Type-2	20%	630.6	119.4	**117.1**	439.4	0.12	0.14	0.10	**0.09**	0.26
		25%	464.3	49.3	**46.9**	111.2	0.15	0.18	**0.11**	0.18	1.15
		30%	113.8	**22.7**	23.1	16.4	0.10	0.21	0.01	**0.00**	1.58
	Average		354.5	63.3	**59.5**	186.5	0.15	0.21	**0.09**	0.10	0.97
MM	Type-1	20%	2766.2	**262.2**	277.0	3521.5	0.12	0.26	0.11	**0.00**	0.39
		25%	1272.7	176.6	**154.9**	12447.8	0.09	0.13	**0.04**	0.25	0.98
		30%	4049.5	**235.3**	310.3	14746.8	0.13	0.25	**0.00**	1.37	1.77
	Type-2	20%	3916.3	504.6	**396.0**	11703.3	0.13	0.26	0.13	**0.02**	0.22
		25%	1486.7	226.3	**164.9**	21226.9	0.09	0.12	**0.01**	0.38	2.30
		30%	8042.7	**431.0**	720.4	15323.7	0.13	0.24	**0.01**	0.95	1.80
Average			3589.0	**306.0**	337.2	13161.7	0.11	0.21	**0.05**	0.50	1.24

Table 18 Solution time and quality of 2SLDR and SDDP bounds over large-sizes synthetic instances.

			Average Time (sec)				Opt. Gap (%)		
Trans.	Modality	Cap.	LDR-T	LDR-M	S-LB	(opt)	LDR-T	LDR-M	S-UB
HN	Type-1	20%	2,100.9	1,961.8	**449.3**	(10)	0.51	0.33	**0.13**
		25%	1,679.4	1,628.6	**288.0**	(10)	0.78	0.39	**0.15**
		30%	965.1	864.2	**56.6**	(10)	0.61	0.48	**0.18**
	Type-2	20%	2,501.3	1,794.3	**730.9**	(10)	0.56	0.31	**0.12**
		25%	1,660.5	1,764.6	**203.7**	(10)	0.78	0.40	**0.13**
		30%	950.1	1,035.3	**57.9**	(10)	0.62	0.49	**0.18**
	Average (Total)		1,642.9	1,508.1	**297.7**	(60)	0.64	0.40	**0.15**
MA	Type-1	20%	2,266.5	2,855.8	**1,913.0**	(10)	0.50	0.32	**0.13**
		25%	3,067.1	2,458.6	**1,295.8**	(10)	0.84	0.46	**0.19**
		30%	1,051.1	1,028.4	**133.1**	(10)	0.62	0.49	**0.19**
	Type-2	20%	4,065.5	**3,488.9**	3,500.0	(10)	0.59	0.34	**0.14**
		25%	4,295.9	4,107.4	**1,960.6**	(10)	0.89	0.40	**0.18**
		30%	1,272.7	1,252.9	**150.3**	(10)	0.65	0.52	**0.20**
	Average (Total)		2,669.8	2,532.0	**1,492.1**	(60)	0.68	0.42	**0.17**
MM	Type-1	20%	8,288.1	**4,776.9**	20,125.0	(1)	39.71	**33.00**	39.39
		25%	14,398.1	**12,203.6**	21,601.1	(0)	15.73	**15.08**	17.53
		30%	**1,911.4**	2,076.6	21,601.6	(0)	2.41	**0.67**	2.09
	Type-2	20%	14,668.7	**13,135.3**	21,600.7	(0)	39.55	**37.42**	43.68
		25%	16,280.7	**16,051.0**	21,601.6	(0)	23.98	**23.22**	26.70
		30%	2,391.2	**1,925.4**	21,601.4	(0)	3.63	**1.51**	3.50
	Average (Total)		9,656.4	**8,361.5**	21,355.2	(1)	20.84	**18.48**	22.15

Appendix I: Additional Results for Policy Structures and Managerial Insights

We analyze the underlying policies of different transformations for synthetic instances. We characterize these policies by defining some key metrics, reported in Table 19. Columns "Nodes (%)" show the average percentage of nodes in the scenario tree with an active contingency modality, and "# of Contingencies" provide the average number of contingency modalities used by a policy among all instances. Columns "Aggressiveness (%)" refer to the percentage of capacity increase (between 10-40%) resulting from the contingency modality activation, and "Intensity" to the hurricane's intensity when a contingency modality is first activated. Average values in "# of Contingencies" and "Aggressiveness (%)" are computed over nodes with active modalities. For example, PM only activates one aggressive modality (i.e., 30%) for instances with 20% initial capacity, which is activated when the hurricane's intensity is three or higher.

Table 19 Solution structures for different policies and initial capacities for Type-1 instances.

		Nodes (%)					# of Contingencies					Aggressiveness (%)					Intensity				
Modality	Cap.	HN	MA	PM	MM	FH	HN	MA	PM	MM	FH	HN	MA	PM	MM	FH	HN	MA	PM	MM	FH
Type-1	20%	100	93	72	72	53	1.0	1.0	1.0	1.0	3.5	16	38	30	30	35	2.3	2.5	2.9	2.9	2.9
	25%	10	50	22	22	47	1.0	1.0	1.5	2.1	4.4	10	15	35	36	21	2.3	2.9	4.0	4.0	3.0
	30%	0	7	16	15	19	-	1.0	2.0	4.3	4.6	-	9	17	18	15	-	3.0	4.1	4.1	4.0
Type-2	20%	100	92	72	72	53	1.0	1.0	1.0	1.0	4.1	14	39	52	52	46	2.3	2.5	2.9	2.9	2.9
	25%	10	38	22	16	31	1.0	1.0	1.9	3.8	5.3	15	20	51	55	33	2.3	2.9	4.0	4.1	3.5
	30%	0	7	15	14	17	-	1.0	1.9	4.8	5.7	-	14	29	27	23	-	3.0	4.2	4.1	4.0

We analyze the results for Type-1 instances since similar insights can be concluded for Type-2 instances. We see that a lower initial capacity typically leads to more contingency modality activation and modality choices that are more aggressive. Comparing HN with other transformations, it has more active nodes when the initial capacity is lower, and has fewer activated nodes when the initial capacity is higher. This is attributed to the structure of the policy under HN which resembles an "all-or-nothing" feature due to the

lack of adaptability. The behaviors of HN and MA are quite similar, although the activation timing of MA is later, resulting in a different number of nodes with an active modality. We attribute this similarity to the "propagation effect" in MC-based aggregation policies as we discussed in Appendix G. In contrast, PM and MM result in more adaptive policies that resemble a "wait-and-activate" type of behavior, similar to FH. In these more adaptive policies, contingency modalities tend to be activated when the hurricane reaches a higher intensity state. In addition, the more adaptive policies lead to a more diverse set of modality choices and a better balance between aggressiveness and the number of nodes activated depending on the initial capacity. Comparing the MC-based transformations PM and MM with FH, which has full adaptability, we see the value of additional adaptability in activating less nodes throughout the scenario tree with more diverse contingency modalities with appropriate aggressiveness.

Table 20 presents the same metrics shown in Table 8 but for the larger MC case-study instances. Lastly, Figures 9 present the heat map on modality choices for the larger MC case-study instances.

Table 20 Solution structures for different policies and initial capacities for the larger MC case-study instance.

	Nodes (%)					# of Contingencies					Aggressiveness (%)				
Penalty factor	HN	MA	PM	MM	FH	HN	MA	PM	MM	FH	HN	MA	PM	MM	FH
0.25	33	33	33	54	5	1	1	1	2	3	27	27	27	53	70
1	33	33	33	56	68	1	1	1	2	3	27	27	27	44	54
4	67	67	67	67	68	2	2	2	2	3	53	53	53	53	55

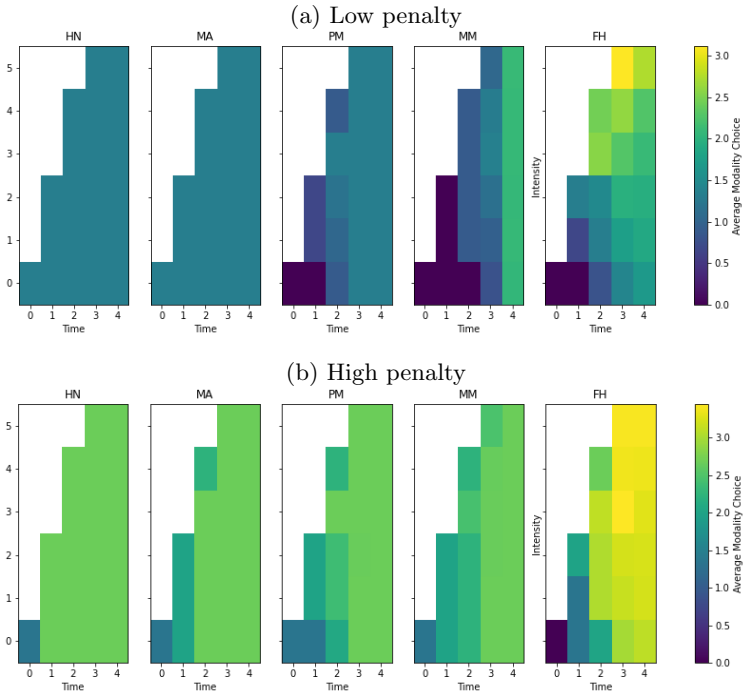

Figure 9 Modality choice heat map over time at different intensity states, for the larger MC with various MC-based transformations for low and high penalty cases.