$$\sum_{\{(i,p):aa_{k,p}=1,dd_{i,p}=1\}} y_{i,p} \leq x_k r_k \qquad \forall k \in K \tag{28}$$

$$\{y, z\} \in \mathbb{R}^+$$

The objective (24) minimizes the expected shortage over all scenarios. Constraint (25) enforces the budget. Constraint (27) ensures assigned demand plus shortage add up to point-to-point demand.
Constraint (28) enforces capacity along each arc.

PS4-2b
Calculate EV.

Run a Python script in a Python shell to solve the problem. The associated deterministic model and scenario data can be found in the folder "PS4/4-2b". The optimal value is 6.1 and the optimal solution is as follows.

x_{rp}	
$x(1)$	7.07143
$x(2)$	0
$x(3)$	5.07143
$x(4)$	0
$x(5)$	2.92857
$x(6)$	4.92857
$x(7)$	0

PS4-2c
We cannot calculate WS exactly because it would involve solving $4.1 * 10^7$ variables. Instead, form a sample mean estimate of WS based on a sample size of 100. State the point estimate and the associated approximate 95% confidence interval.

Run a Python script to solve the problem. The associated files can be found in the folder "PS4/4-2c". Using 50 samples, a point estimate of $WS =$ 6.0 and 95% CI = [5.11 6.89].

PS4-2d
Use the capacity expansion decision from part (b), x_{ev}, to form a sample mean estimate of EEV with a sample size of 100. State the point estimate and the associated approximate 95% confidence interval.

Run a Python script to solve the problem. The associated files can be found in the folder "PS4/4-2d". In the deterministic model, a constraint is added fixing the first stage variable x to x_{ws}. Using 50 samples, a point of $EEV =$ 8.3 and 95% CI = [7.29 9.31].

PS4-2e
Based on your calculations in part (c), form a capacity expansion decision, x_{ws}. Form a point estimate and approximate 95% confidence interval for $EWS = cx_{ws} + Eh(x_{ws}, \xi)$. Again, use a sample size of 100.

Run a Python script to solve the problem. The associated files can be found in the folder "PS4/4-2e". In the deterministic model, a constraint is added fixing the first stage variable x to x_{ws} (which is from PS3-c). Using 50 samples, a point of estimate of $EWS = 7.3$ and 95% CI = [6.49 8.11].

PS4-2f

Solve this problem using the Monte Carlo sampling-based approach for assessing solution quality with common random number streams for upper and lower bound estimation. Solve one initial approximating problem in order to generate the candidate solution x̂. Use $n_g = 20$ replications. Choose the sample size, n, so that you obtain what you regard as "reasonable" results. State your recommended capacity-expansion decision and a confidence interval for the quality of the proposed solution.

Run a computeconf script to solve the problem, which implements the Mak/Morton/Wood multiple replication procedure to calculate the approximate gap of the stochastic solution. The associated files can be found in the folder "PS4/4-2f". To run, use the following command in the appropriate folder.

```
computeconf-m models -i scenariodata –scenario-tree-downsample-
fraction=0.2 –solver=gurobi –solve-ef>PartFresults.txt
```

$$\sum_{\{(i,p):aa_{k,p}=1,dd_{i,p}=1\}} y_{i,p} \le x_k r_k \qquad \forall k \in K \qquad (28)$$

$$\{y,z\} \in \mathbb{R}^+$$

The objective (24) minimizes the expected shortage over all scenarios. Constraint (25) enforces the budget. Constraint (27) ensures assigned demand plus shortage add up to point-to-point demand. Constraint (28) enforces capacity along each arc.

PS4-2b

Calculate EV.

Run a Python script in a Python shell to solve the problem. The associated deterministic model and scenario data can be found in the folder "PS4/4-2b". The optimal value is 6.1 and the optimal solution is as follows.

	x_{RP}
x(1)	7.07143
x(2)	0
x(3)	5.07143
x(4)	0
x(5)	2.92857
x(6)	4.92857
x(7)	0

PS4-2c

We cannot calculate WS exactly because it would involve solving $4.1*10^7$ variables. Instead, form a sample mean estimate of WS based on a sample size of 100. State the point estimate and the associated approximate 95% confidence interval.

Run a Python script to solve the problem. The associated files can be found in the folder "PS4/4-2c". Using 50 samples, a point estimate of WS = 6.0 and 95% CI = [5.11 6.89].

PS4-2d

Use the capacity expansion decision from part (b), x_{ev}, to form a sample mean estimate of EEV with a sample size of 100. State the point estimate and the associated approximate 95% confidence interval.

Run a Python script to solve the problem. The associated files can be found in the folder "PS4/4-2d". In the deterministic model, a constraint is added fixing the first stage variable x to x_{ev}. Using 50 samples, a point estimate of EEV = 8.3and 95% CI = [7.29 9.31].

PS4-2e

Based on your calculations in part (c), form a capacity expansion decision, x_{ws}. Form a point estimate and approximate 95% confidence interval for $EWS = cx_{ws} + Eh(x_{ws}, \tilde{\xi})$. Again, use a sample size of 100.

Run a Python script to solve the problem. The associated files can be found in the folder "PS4/4-2e". In the deterministic model, a constraint is added fixing the first stage variable x to x_{ws} (which is from PS3-c). Using 50 samples, a point estimate of EWS = 7.3and 95% CI = [6.49 8.11].

PS4-2f

Solve this problem using the Monte Carlo sampling-based approach for assessing solution quality with common random number streams for upper and lower bound estimation. Solve one initial approximating problem in order to generate the candidate solution $\hat{x}$. Use n_g = 20 replications. Choose the sample size, n, so that you obtain what you regard as "reasonable" results. State your recommended capacity-expansion decision and a confidence interval for the quality of the proposed solution.

Run a computeconf script to solve the problem, which implements the Mak/Morton/Wood multiple replication procedure to calculate the approximate gap of the stochastic solution. The associated files can be found in the folder "PS4/4-2f". To run, use the following command in the appropriate folder.

```
computeconf–m models –i scenariodata --scenario-tree-downsample-fraction=0.2 --solver=gurobi --solve-ef>PartFresults.txt
```

Work was done by Nikola Juskovica from REBC01