DTU

Decision Making under Uncertainty (02435)

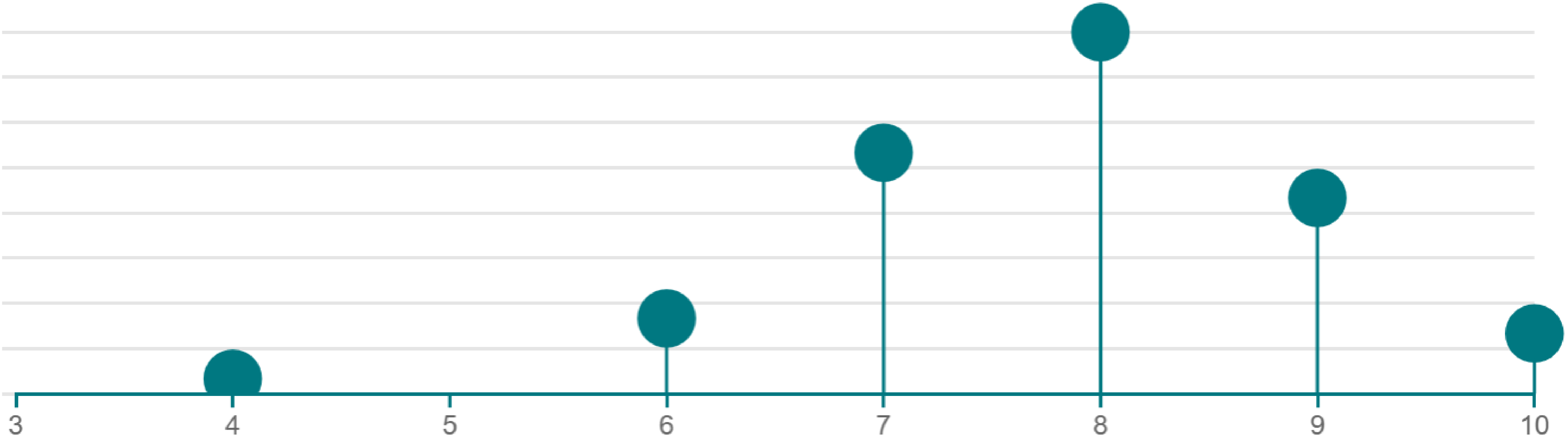
Section for Dynamical Systems, DTU Compute.

Feedback & Follow-up

1 How was today (scale 1-10)?

63

Mean average: 7.86



What you liked

- Concise
- I think the concepts and lectures are easy to understand and have a good relation to the project
- Good visualisations.
- The clustering explanations
- multi stage sp policy examples
- The length of the lecture, the clarity of your explanations
- More clear than last time
- Going over examples
- explanation with tree diagrams
- Well visually explained
- Nice
- clearer than yesterday,
- Generally everything but it was especially nice that time was taken to ensure that we understood the principles
- Good explanations, good visualisations and i liked the length
- This lecture was one of the best I've had in a very long time! Good examples, I also liked that you included the part of how to do use the information in the assignment
- Like the concepts
- Examples
- Concept of non anticipativity
- Proactive
- The recap and the flow
- It was nice to go through last week's questions and the quiz questions were easy to think about in the short time we have to answer.
- Good overviews of how to move forward on the multistage approaches
- I liked to the link the assignment after the theory
- Thorough explanation of concepts
- Same as always
- Clear
- the fact that this Time you have spent more time on the most tedious concepts
- Clear explanation
- visual support
- Good walkthrough of everything. Though I still have some doubt about a few things such as the Non-anticipativity for multi stage
- Explanation
- The concepts were explained well and tie into each other. Really helps having the same exercise to use it on
- practical examples
- more clear
- Extensive explanation
- The recap/summary of the precious lecture in the start today. It's very very useful.
- The umbrella example
- How the path of scenarios are drawn clearly
- The multi stage SP was simple to understand
- More concise lecture, good slides
- The whole lecture (including the Recap)
- The content of the lecture
- The illustrations was nice.
- More complex material Actually talking about multi-stage
- Recap
- Go through visualization
- clear problems easy to understand
- The slides were very explanative
- Good walkthrough
- Lot of visualization
- New content, explained well
- Good examples and quite clearly formulated
- Good

What you disliked

Maybe some coding ideas in how to generate the sets for the non-anticipativity constraints for a 24 hour horizon would be nice:)	I would be nice to have the concepts explained more in depth - maybe with example and draw when you are talking	I think some of the explinations made me more confused. Especially the umbrella example
Too quick on the non-anticipativity Confusion on the multi-stage scenarios	not having the slides from the beggining	Could have more content, felt easy
Not a lot of new material, we could cover more in that time	please explain the non-anticipacy and sets	Coding examples would be great, its hard for me to see the relation.
I still dont really grasp the concept, but i guess i just need time	but still fast, not the teaching pace but the speaking pace are really fast	Could have used a second explanation of the s and s' equality constraint (can't remember the name)
i dont know	could we get some coding examples on how it works?	I just hink it is a little confusing as every lecture I feel like I know wat to do in the assignment, but then you tell us something new ext time
- slides not uploaded before - sometimes hard to follow	Maybe a bit too fast, with the umbrella explanation	The lecture slides were uploaded a little late. I like having them open on my computer during the lecture so I can go back and look at things I missed
Could spend a bit more time with respect to the assignment	Some of the concepts are not well enough coupled with the assignments. It can be hard to visualize how this should be implemented.	Feel like there were fewer questions throughout the lecture and it was oberall a little less dynamic
NA	Too much time spend on feedback	Nothing.

Topic is tricky to understand	I still don't see the connection to the assignment. how to do the scenario setting in our assignment?	Nothing.
More detailed explanations of whats happening in the slides.	Nothing	Nothing
The lectures weren't uploaded beforehand which made it difficult to take notes in the start	Nothing	Nothing this time.
Still no hints on python code	Would have liked som coding examples of how to implement the mentioned theory.	As a math student I think I would learn quicker from a more structured lecture (definitions > examples) or text
Nan	No coding examples again	I would like more short quizzes
Not having slides in the beginning	Nothing much	NA

Plan

→ ~~Task 0~~

→ ~~Task 1~~
~~Building an evaluation framework for sequential decision-making methods~~

→ ~~Task 2~~
~~Stochastic Programming policy (2-stage)~~
~~+ Expected Value policy a.k.a. MPC~~

→ ~~Task 2~~

~~Multi-stage Stochastic Programming + caveats~~

→ Week 5: Assignment Work for Task 2 and Q&A

→ Weeks 6-7: Task 3
Approximate Dynamic Programming

→ Week 8: Assignment Work for Task 3 and Q&A

→ Weeks 9-11: Assignment B
Robust Optimization

Task 4 is about reporting the results from Tasks 2 and 3

Agenda for today

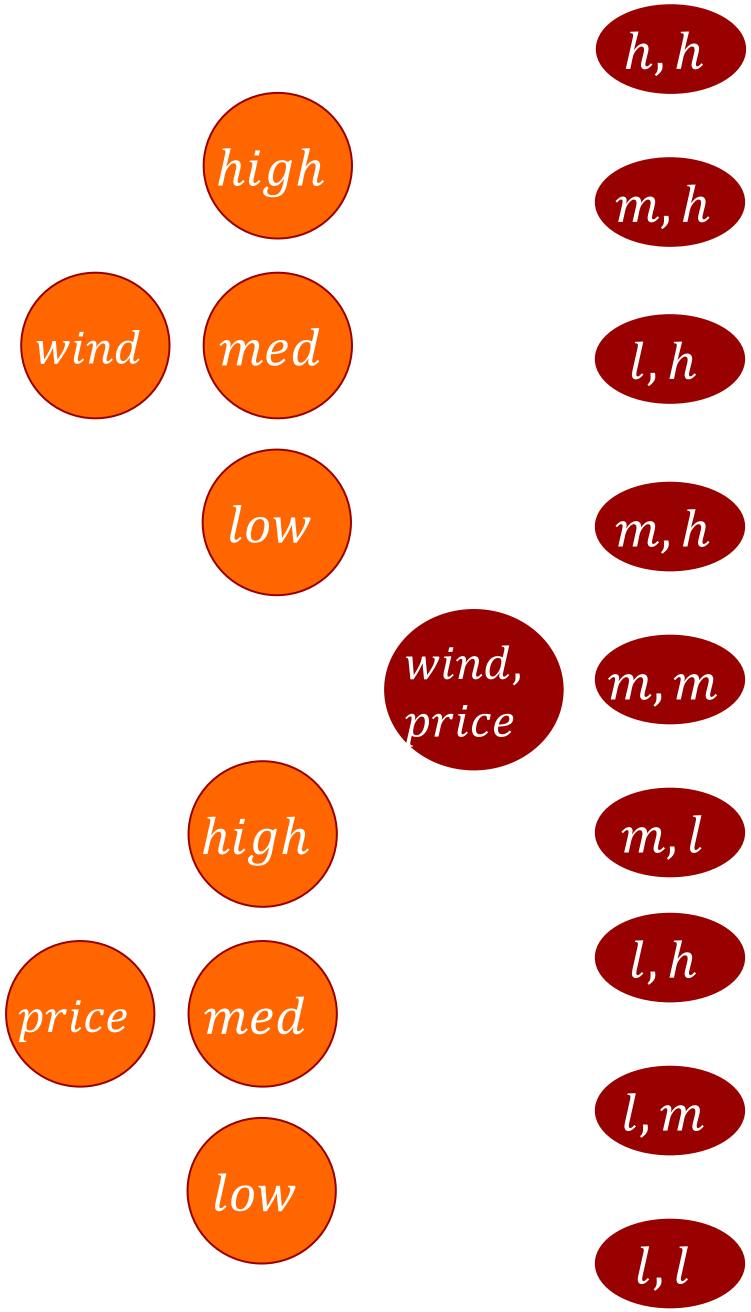
1. Quick Multi-stage Stochastic Programming recap + addressing some of your feedback comments
2. Assignment Work and Q&A

Number of Scenarios

wind_process

price_process

What we call a “scenario”, refers to a joint realization (combination) of all uncertain variables.



Scenario Reduction

How to apply clustering to scenario reduction:

- One scenario equals one data point
- Distance measure has to be selected, e.g., euclidean distance, manhattan distance

Resulting scenarios:
One scenario per cluster (here: the centroid/medoid)

Redistribution of probabilities:
The new scenario probability is the sum of probabilities of scenarios in the cluster

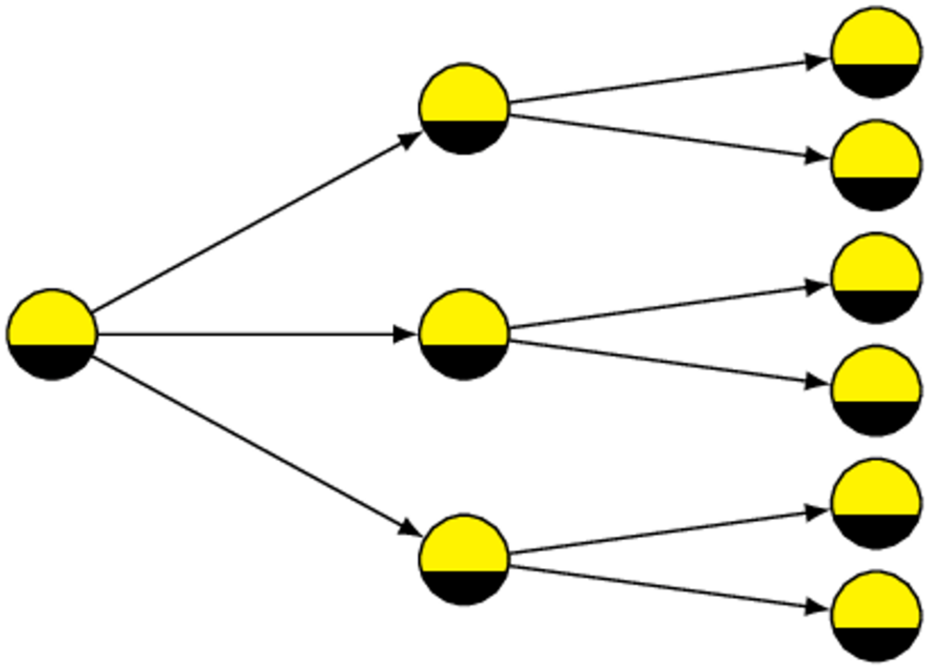
Advantage: Ready-to-use implementation in many programming languages

Decision-making under uncertainty

Multi-stage Stochastic Programming

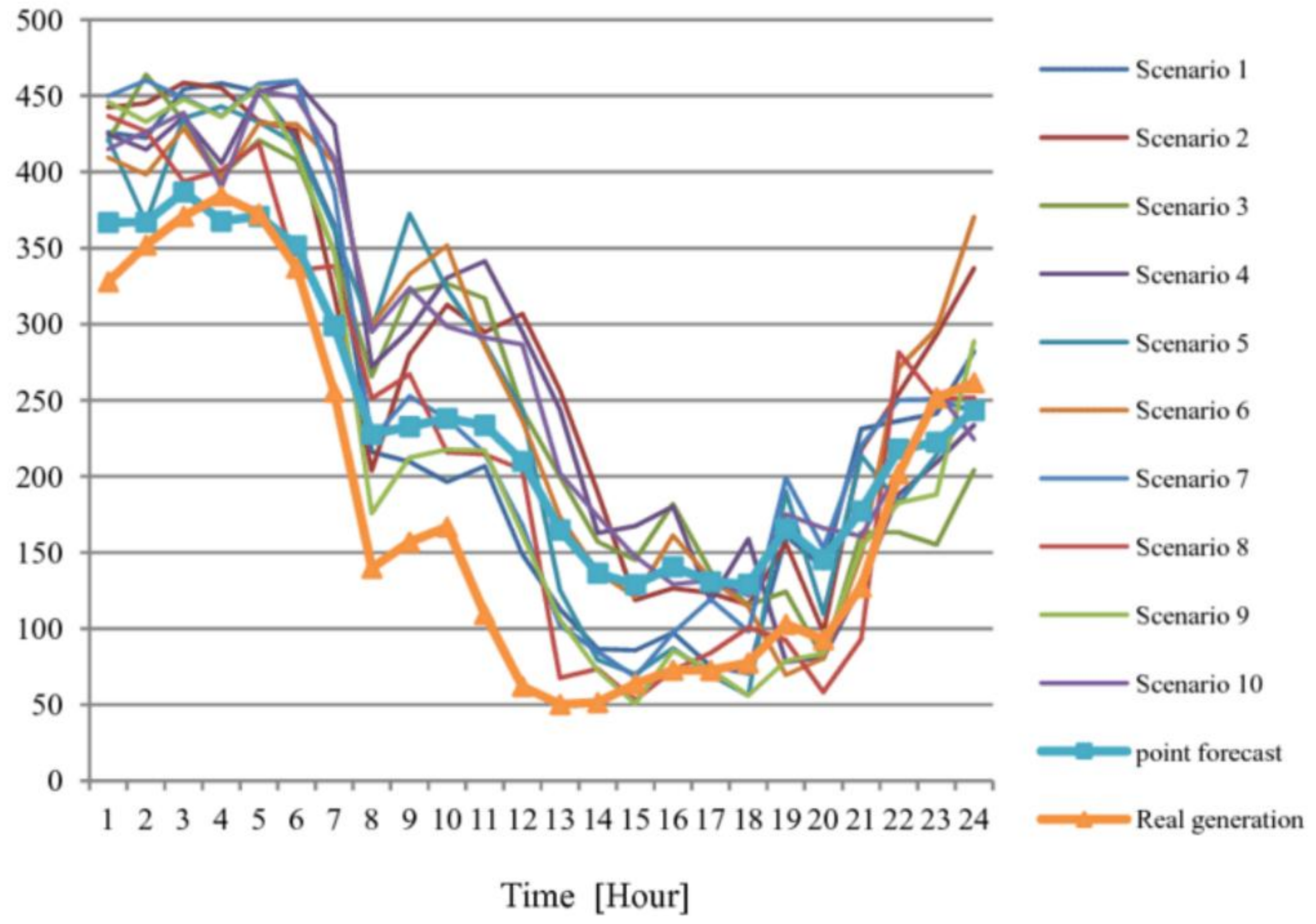
Scenario Trees

Example: Wind output across 3 stages

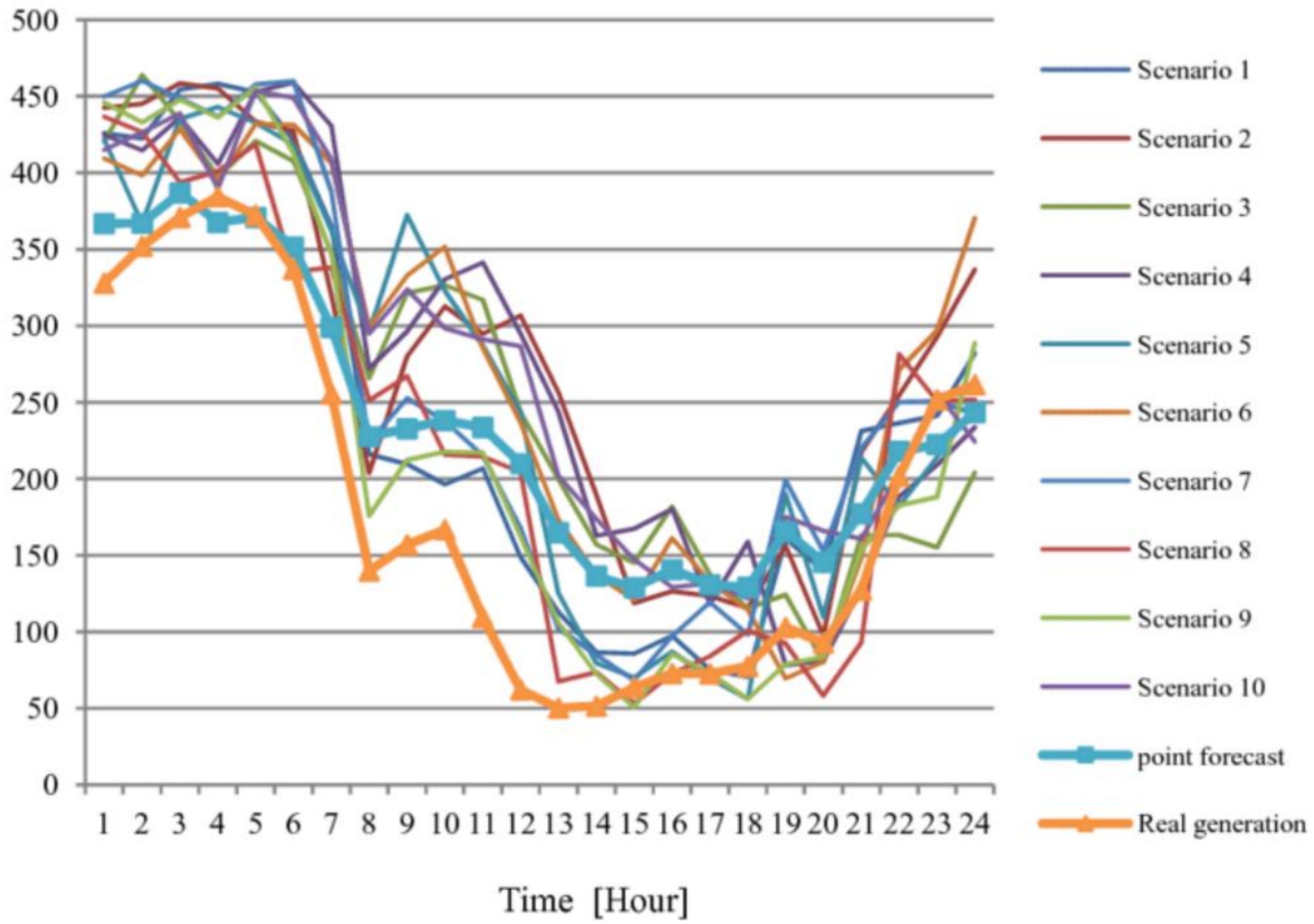


Nodes	represent the points where decisions must be made
Branches	represent different realizations of uncertainty
Root node	represents first-stage decision at the beginning of the planning horizon
# leaf nodes	equals # scenarios
Scenario	is a path from root node to a leaf node

Non-Anticipativity in Multi-stage SP

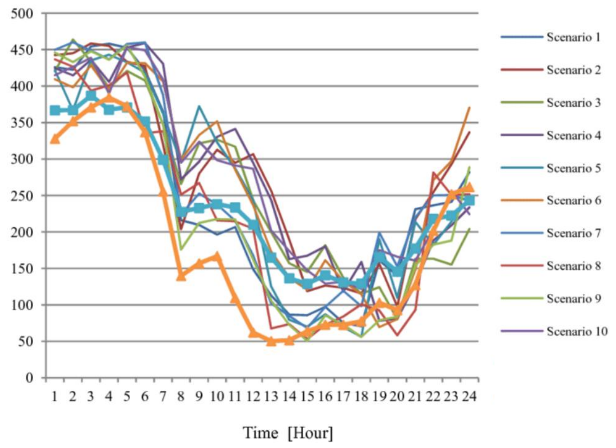


Non-Anticipativity in Multi-stage SP



Let's say you solve such a program and it gives you an expected profit P.
Is P an overestimation / underestimation / accurate estimation?

Non-Anticipativity in Multi-stage SP

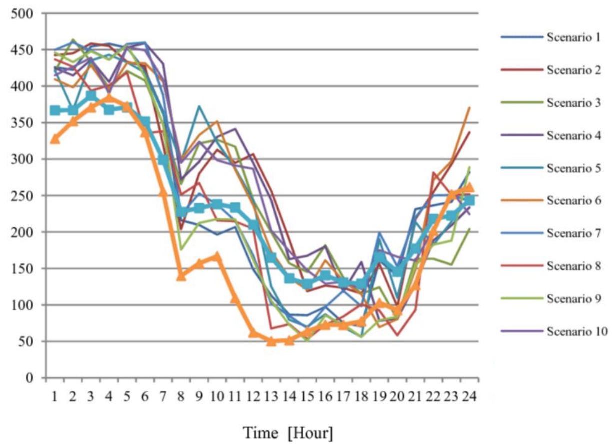


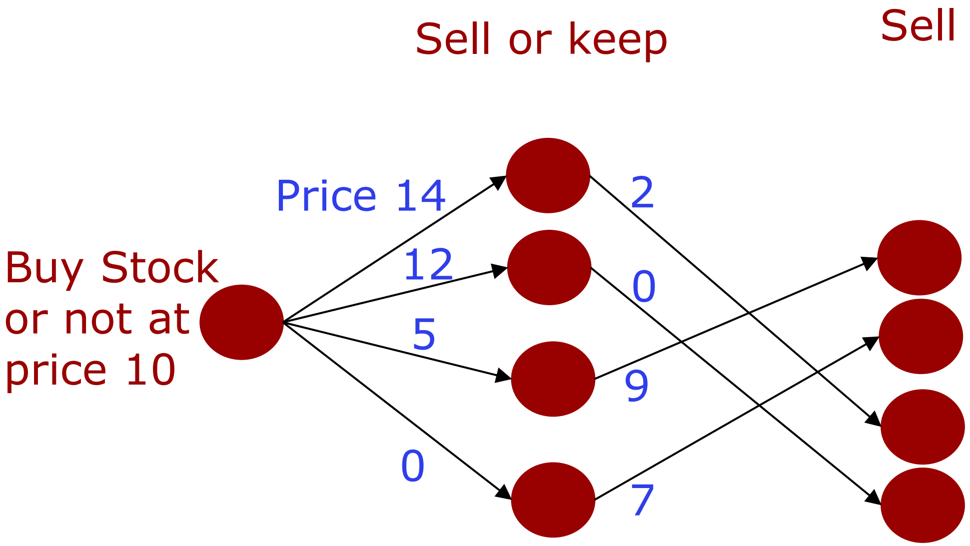
Buy Stock or not at price 10

Sell or keep

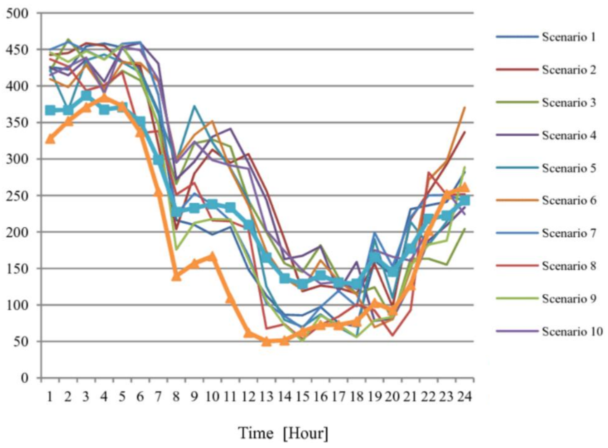
Sell

Non-Anticipativity in Multi-stage SP





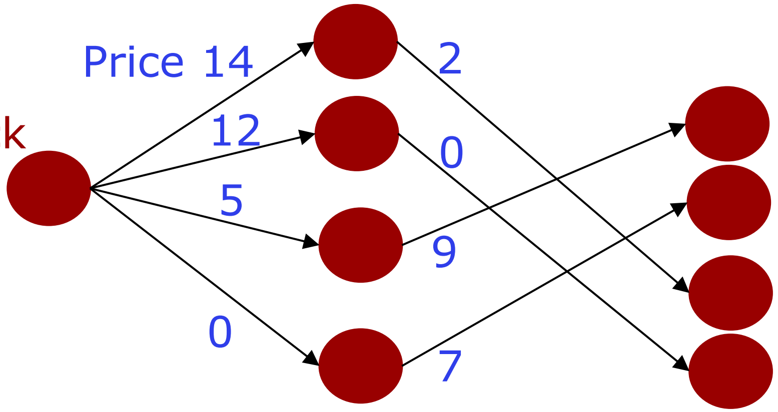
Non-Anticipativity in Multi-stage SP



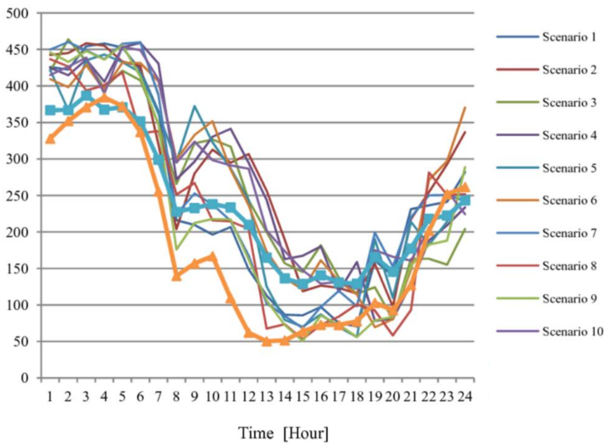
Expected Return from buying =
0.25*(14 + 12 + 9 + 7) – 10 = 0.5

Sell Sell or keep Sell

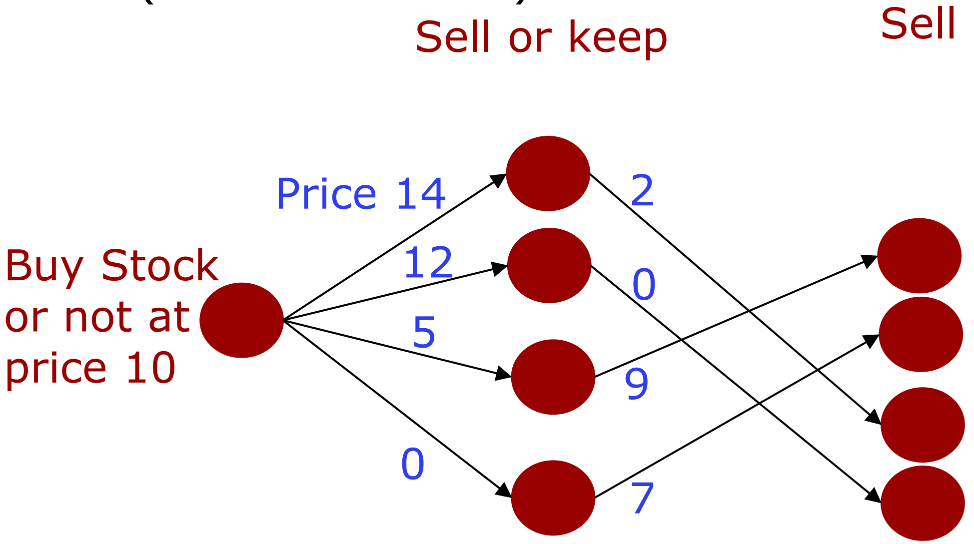
Buy Stock or not at price 10

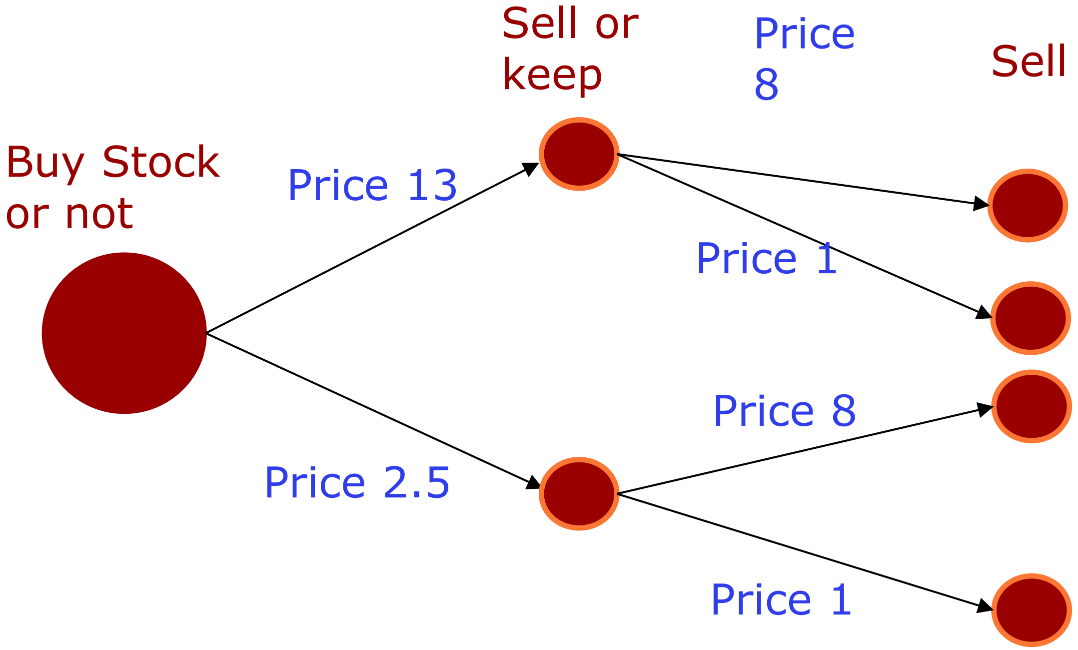


Non-Anticipativity in Multi-stage SP

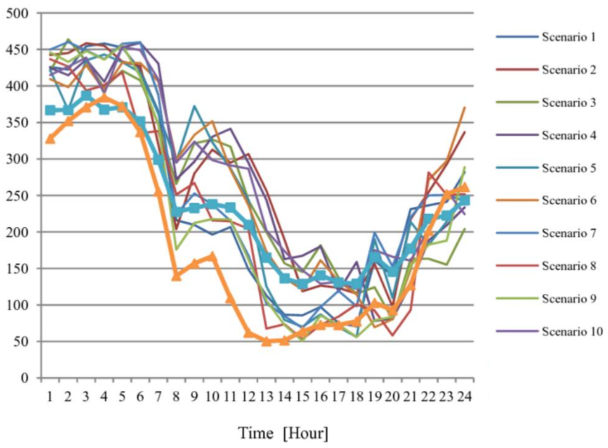


Expected Return from buying =
0.25*(14 + 12 + 9 + 7) – 10 = 0.5

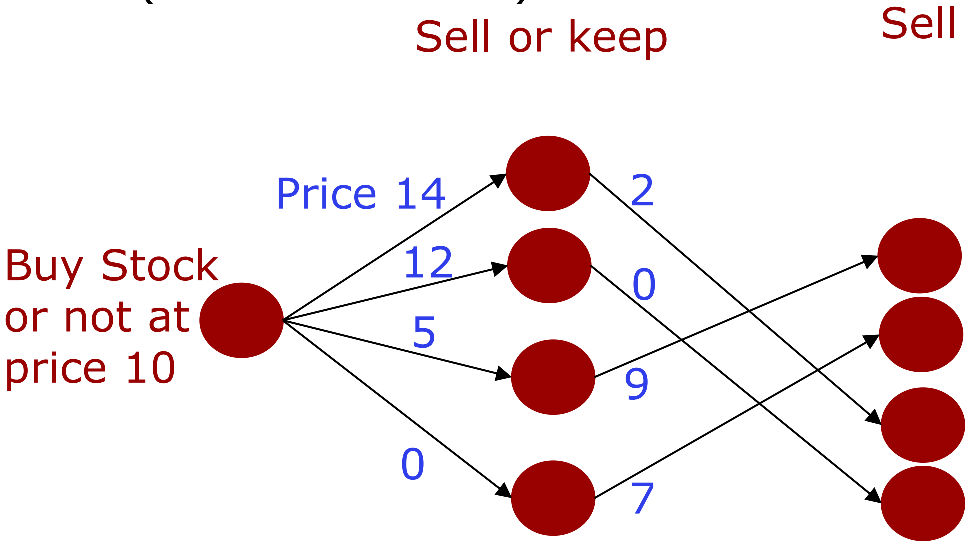




Non-Anticipativity in Multi-stage SP

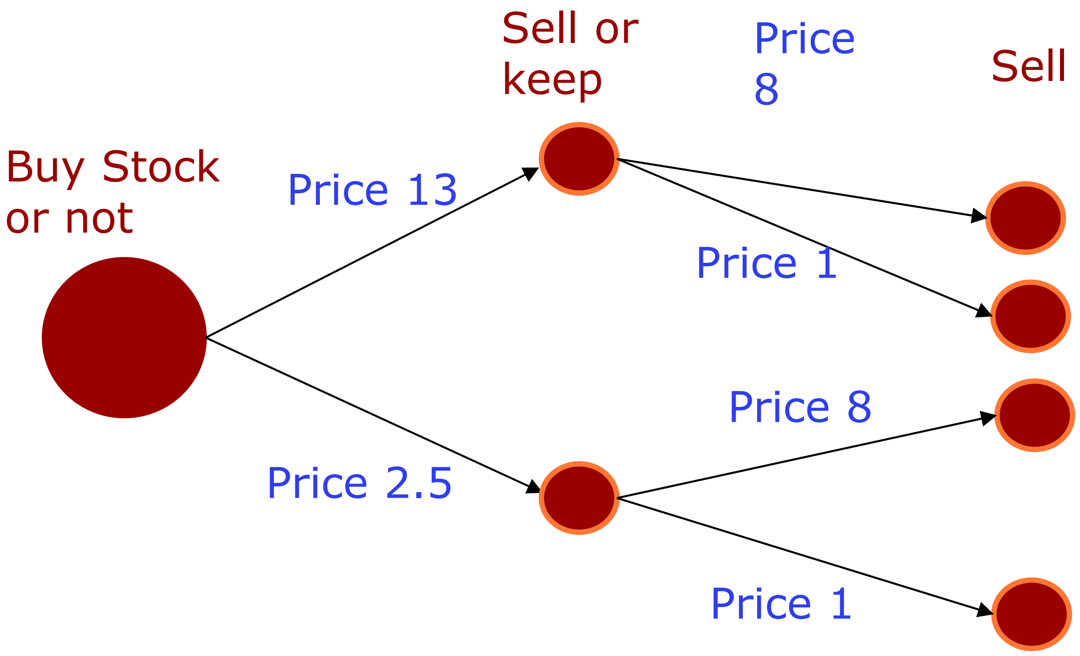


Expected Return from buying =
0.25*(14 + 12 + 9 + 7) – 10 = 0.5



Expected Return from buying =
0.25*(13 + 13 + 8 +1) – 10 = -1.25

Sell



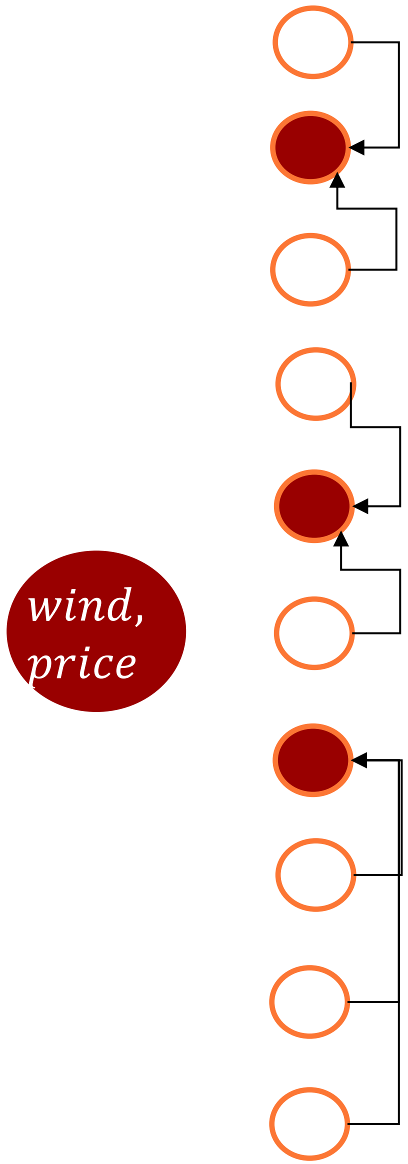
Repeated Branch-out & Cluster

wind, price

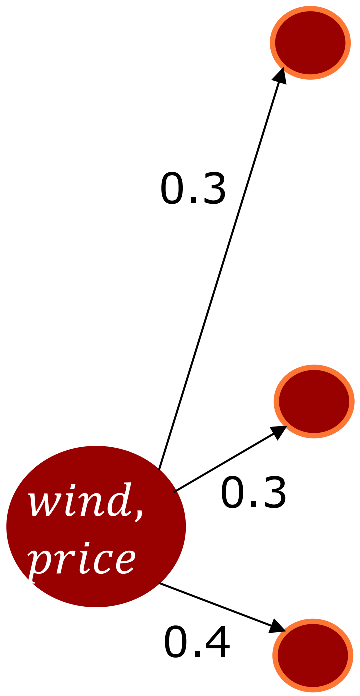
Branch-out from current stage t

Cluster

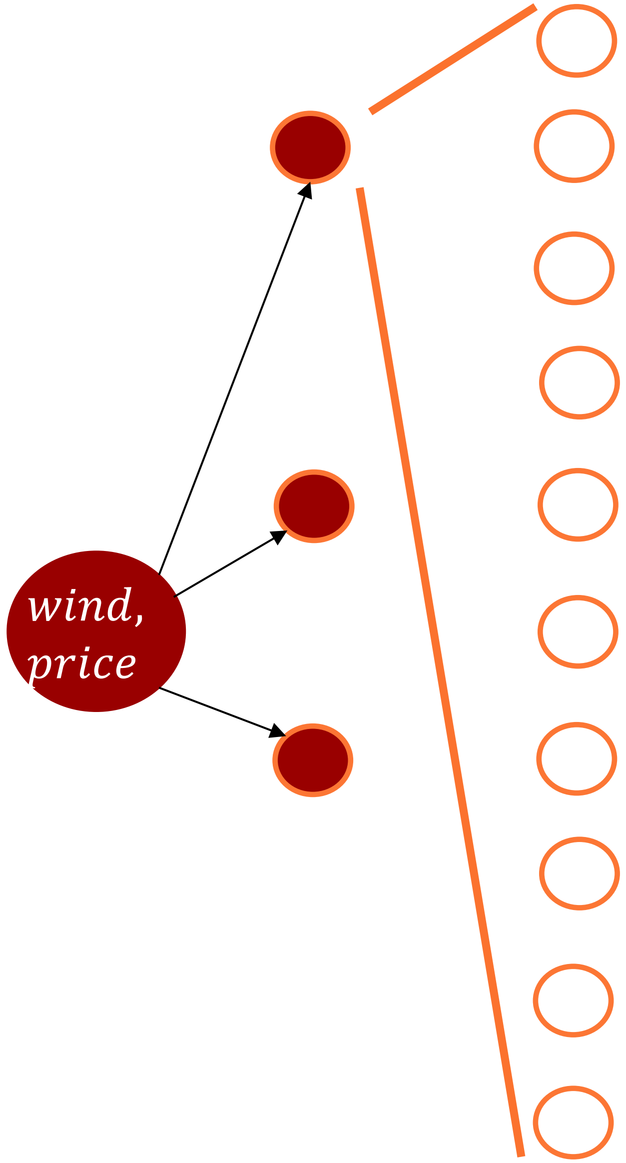
Allocate Probability to each Centroid



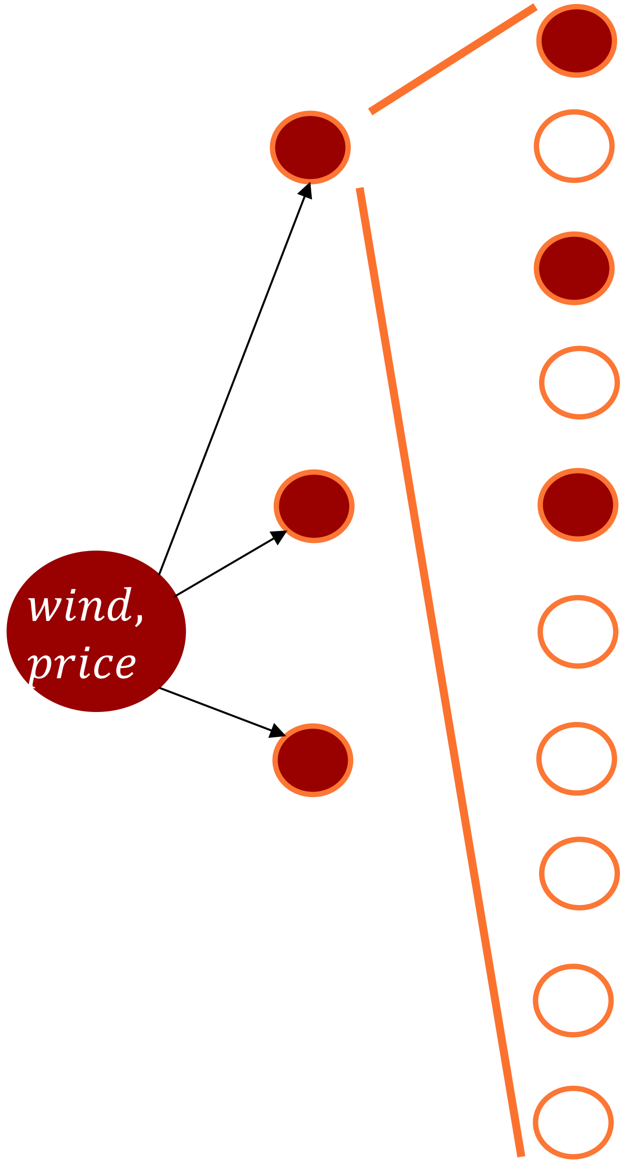
Ready to Move on…



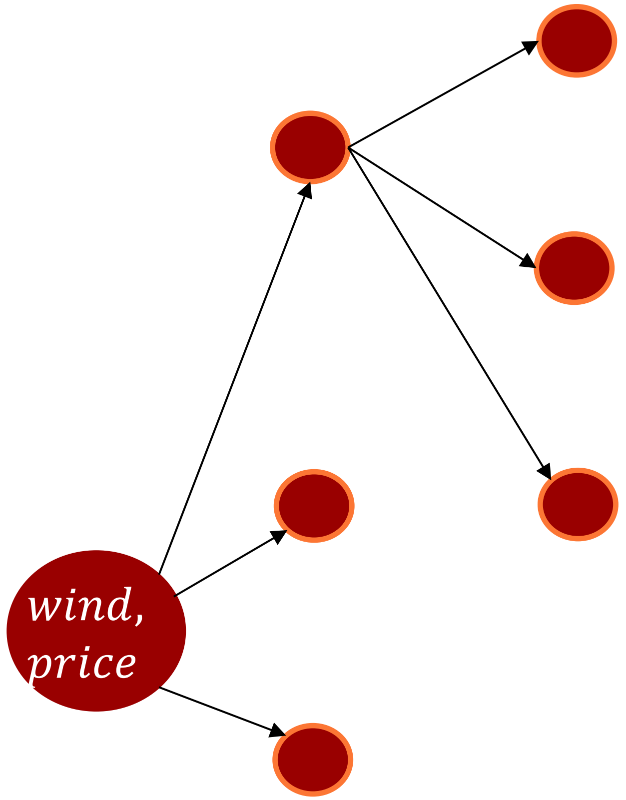
For each of the reduced $t + 1$ nodes: Branch-out



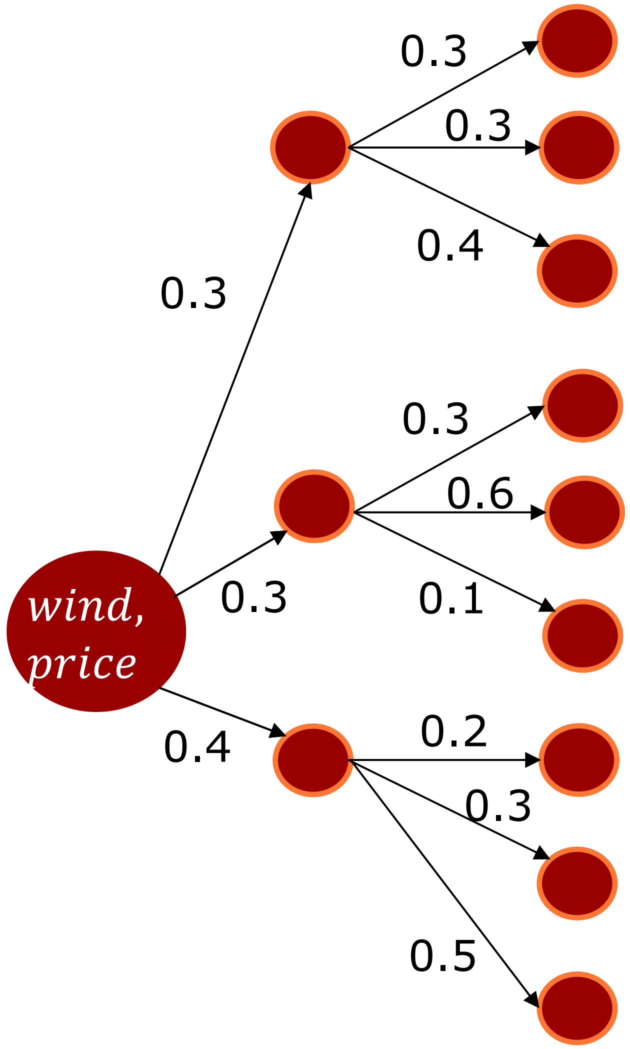
Cluster



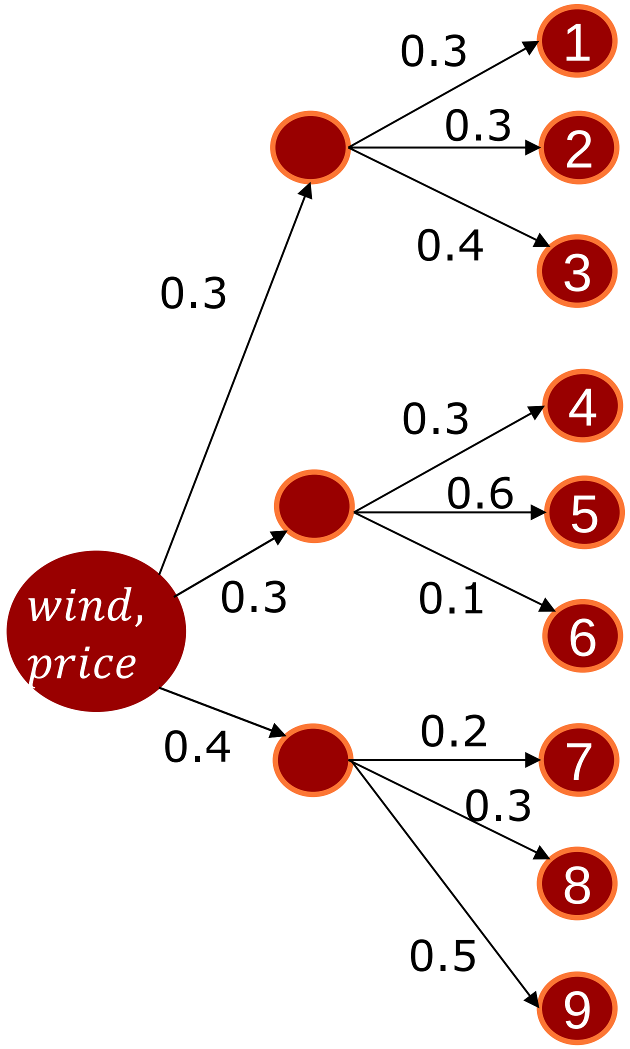
Repeat…



Repeat…



Repeat…



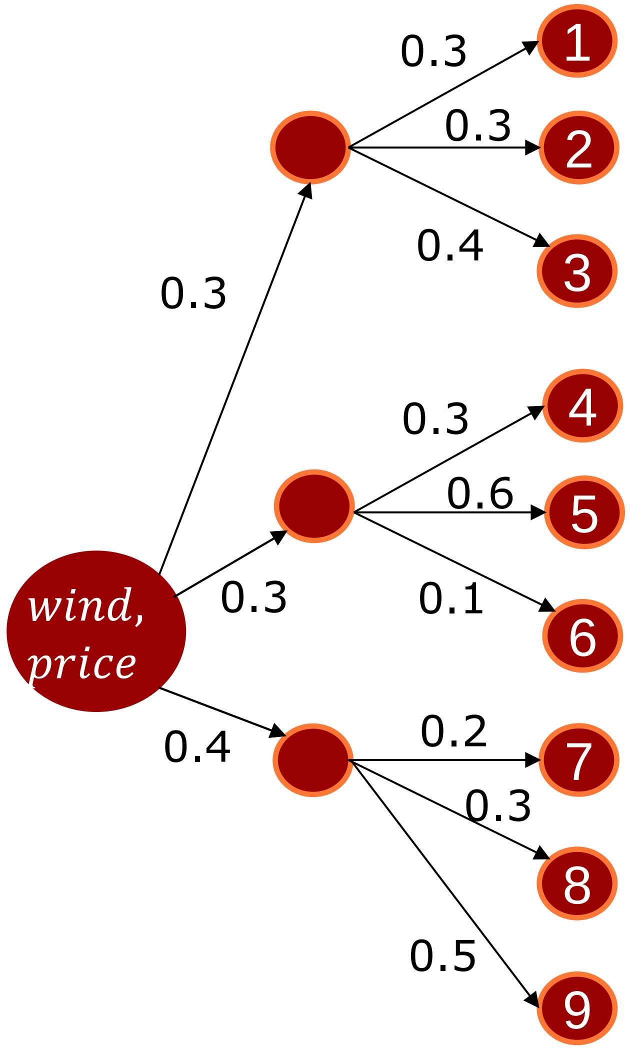
Each path (ending up to a leaf node) is one scenario.

Calculate the probability of each scenario, by multiplying the probabilities across the path.

Scenario Creation in Multi-Stage SP

1. Begin at the current stage t
2. Generate Samples for $t+1$
3. Reduce them using clustering
4. Allocate Centroid Probabilities
5. For $t+2$, branch out from each of the <u>reduced</u> samples (centroids) of $t+1$
6. Reduce the $t+2$ samples using clustering
7. For $t+3$, branch out from each of the <u>reduced</u> samples (centroids) of $t+2$
8. …
9. In the end, calculate the probability of each scenario (path) by multiplying the probabilities along the path's branches

How many Scenarios?



Each path (ending up to a leaf node) is one scenario.

2 types of uncertainties: wind & price

Each branches out to B branches at each stage.

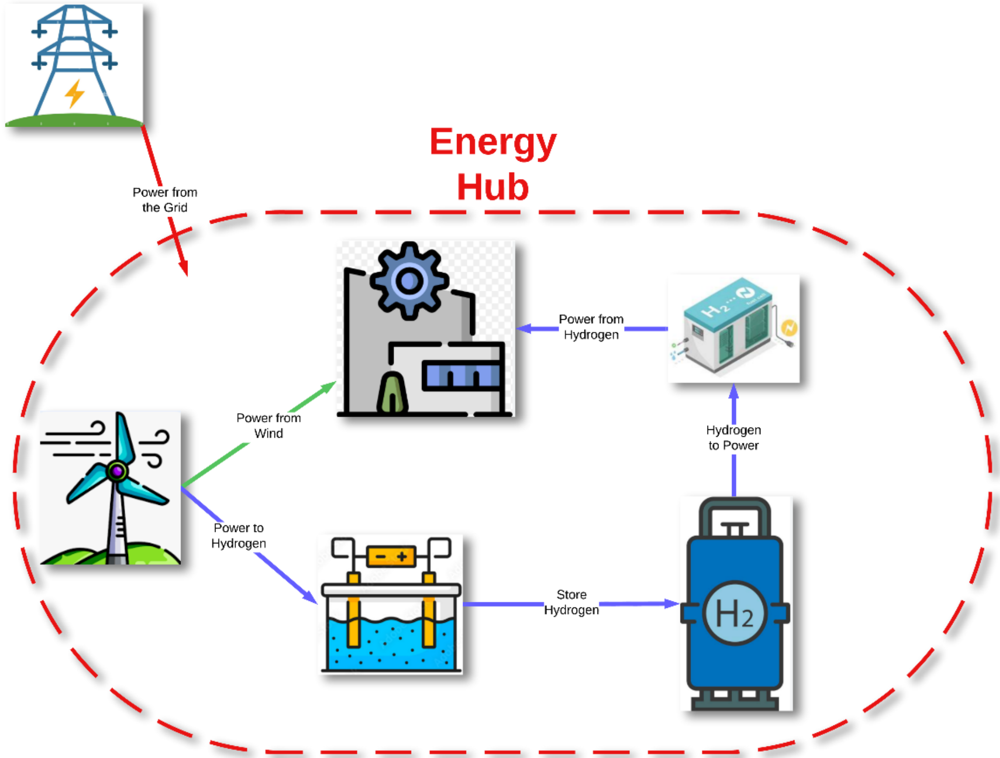
You look L stages ahead.

Number of Scenarios = $B^{2*(L-1)}$

Total number of variables $(x_{t,s})$ = $V * L * B^{2*(L-1)}$

Trade-off between the branching factor B and the length of the lookahead horizon L.

Assignment A, Task 1



Deliverable 1: MDP

State variables $\boldsymbol{x}_t = \{x_{1,t}, x_{2,t}, \dots\}$

Decision variables $\boldsymbol{u}_t = \{u_{1,t}, u_{2,t}, \dots\}$

Dynamics $\boldsymbol{x}_{t+1} = f(\boldsymbol{x}_t, \boldsymbol{u}_t)$

Cost function $c_t = g(\boldsymbol{x}_t, \boldsymbol{u}_t)$

Deliverable 2: Policy Evaluation Framework

Input: *policy* (python function that returns decisions)
Initialize state variables
For experiment 1 to E:
  For stage 1 to H:
    decisions = *policy*(state)
    check/correct decisions if inconsistent
    calculate cost for this stage and experiment
    calculate state at next stage
  calculate total cost of policy for this experiment
Return: expected policy cost (average over experiments)

Decision-making under uncertainty

How to build your Policy

How to build a Multi-Stage SP Policy

Input: Current State

Output: Here-and-now decisions

1) Decide how many variables you can afford
2) Define the number of look-ahead stages and number of branches:
3) Generate your scenarios
4) Calculate the probability of each scenario
5) Create and populate the "non-anticipativity" sets $S(s,t)$.
6) Solve the Optimization problem.
7) Return the here-and-now decisions (only)

How to build a Multi-Stage SP Policy

Input: Current State

Output: Here-and-now decisions

3. the stochastic programming policy for different configurations (namely, longer lookahead horizon length but fewer branches/scenarios vs shorter lookahead horizon but more branches/scenarios). Try 4 configurations such that the total number of binary variables in each is not more than 1000.

1) Decide how many variables you can afford (here, it is given to you).
2) Define the number of look-ahead stages and number of branches:
3) Generate your scenarios
4) Calculate the probability of each scenario
5) Create and populate the "non-anticipativity" sets $S(s,t)$.
6) Solve the Optimization problem.
7) Return the here-and-now decisions (only)

How to build a Multi-Stage SP Policy

Input: Current State

Output: Here-and-now decisions

3. the stochastic programming policy for different configurations (namely, longer lookahead horizon length but fewer branches/scenarios vs shorter lookahead horizon but more branches/scenarios). Try 4 configurations such that the total number of binary variables in each is not more than 1000.

1) Decide how many variables you can afford (here, it is given to you).
2) Define the number of look-ahead stages and number of branches.
   remember, $\mathrm{V} * L * B^{(L-1)*2} \leq \mathit{maximum\ number\ of\ variables}$
3) Generate your scenarios
4) Calculate the probability of each scenario
5) Create and populate the "non-anticipativity" sets $S(s, t)$.
6) Solve the Optimization problem.
7) Return the here-and-now decisions (only)

How to build a Multi-Stage SP Policy

Input: Current State

Output: Here-and-now decisions

3. the stochastic programming policy for different configurations (namely, longer lookahead horizon length but fewer branches/scenarios vs shorter lookahead horizon but more branches/scenarios). Try 4 configurations such that the total number of binary variables in each is not more than 1000.

1) Decide how many variables you can afford (here, it is given to you).
2) Define the number of look-ahead stages and number of branches.
   remember, $V * L * B^{(L-1)*2} \leq maximum\ number\ of\ variables$
3) Generate your scenarios
   (repeated Branch-out & Cluster)
4) Calculate the probability of each scenario
   (multiply the probabilities across the scenario's branches)
5) Create and populate the "non-anticipativity" sets $S(s,t)$.
6) Solve the Optimization problem.
7) Return the here-and-now decisions (only)

How to build a Multi-Stage SP Policy

Input: Current State

Output: Here-and-now decisions

3. the stochastic programming policy for different configurations (namely, longer lookahead horizon length but fewer branches/scenarios vs shorter lookahead horizon but more branches/scenarios). Try 4 configurations such that the total number of binary variables in each is not more than 1000.

1) Decide how many variables you can afford (here, it is given to you).
2) Define the number of look-ahead stages and number of branches.
   remember, $V * L * B^{(L-1)*2} \leq maximum\ number\ of\ variables$
3) Generate your scenarios
   (repeated Branch-out & Cluster)
4) Calculate the probability of each scenario
   (multiply the probabilities across the scenario's branches)
5) Create and populate the "non-anticipativity" sets $S(s,t)$.
   Simply iterate through all scenario pairs and, for each pair, check:
   up to which stage they share history.
6) Solve the Optimization problem.
7) Return the here-and-now decisions (only)

```
For s in Scenarios:
  For t in (1, L):
    For s' in Scenarios:
      if s == s':
        then: S(s,t) ← s'
```

How to build a Multi-Stage SP Policy

Input: Current State

Output: Here-and-now decisions

3. the stochastic programming policy for different configurations (namely, longer lookahead horizon length but fewer branches/scenarios vs shorter lookahead horizon but more branches/scenarios). Try 4 configurations such that the total number of binary variables in each is not more than 1000.

1) Decide how many variables you can afford (here, it is given to you).
2) Define the number of look-ahead stages and number of branches.
   remember, $V * L * B^{(L-1)*2} \leq maximum\ number\ of\ variables$
3) Generate your scenarios
   (repeated Branch-out & Cluster)
4) Calculate the probability of each scenario
   (multiply the probabilities across the scenario's branches)
5) Create and populate the "non-anticipativity" sets $S(s,t)$.
   Simply iterate through all scenario pairs and, for each pair, check:
   up to which stage they share history.
6) Solve the Optimization problem.
   Include Non-Anticipativity constraints: $x_{s,t} = x_{s',t}\ \forall t, \forall s, \forall s' \in S(s,t)$
7) Return the here-and-now decisions (only)

```
For s in Scenarios:
  For t in (1, L):
    For s' in Scenarios:
      if s == s':
        then: S(s,t) ← s'
```

How to build a Multi-Stage SP Policy

Input: Current State

Output: Here-and-now decisions

3. the stochastic programming policy for different configurations (namely, longer lookahead horizon length but fewer branches/scenarios vs shorter lookahead horizon but more branches/scenarios). Try 4 configurations such that the total number of binary variables in each is not more than 1000.

1) Decide how many variables you can afford (here, it is given to you)
2) Define the number of look-ahead stages and number of branches:
   remember, $V * L * B^{(L-1)*2} \leq maximum\ number\ of\ variables$
3) Generate your scenarios
   (repeated Branch-out & Cluster)
4) Calculate the probability of each scenario
   (multiply the probabilities across the scenario's branches)
5) Create and populate the "non-anticipativity" sets $S(s,t)$.
   Simply iterate through all scenario pairs and, for each pair, check:
   up to which stage they share history.
6) Solve the Optimization problem.
   Include Non-Anticipativity constraints: $x_{s,t} = x_{s',t}\ \forall t, \forall s, \forall s' \in S(s,t)$
7) Return the here-and-now decisions (only)

```
For s in Scenarios:
  For t in (1, L):
    For s' in Scenarios:
      if s == s':
        then: S(s,t) ← s'
```