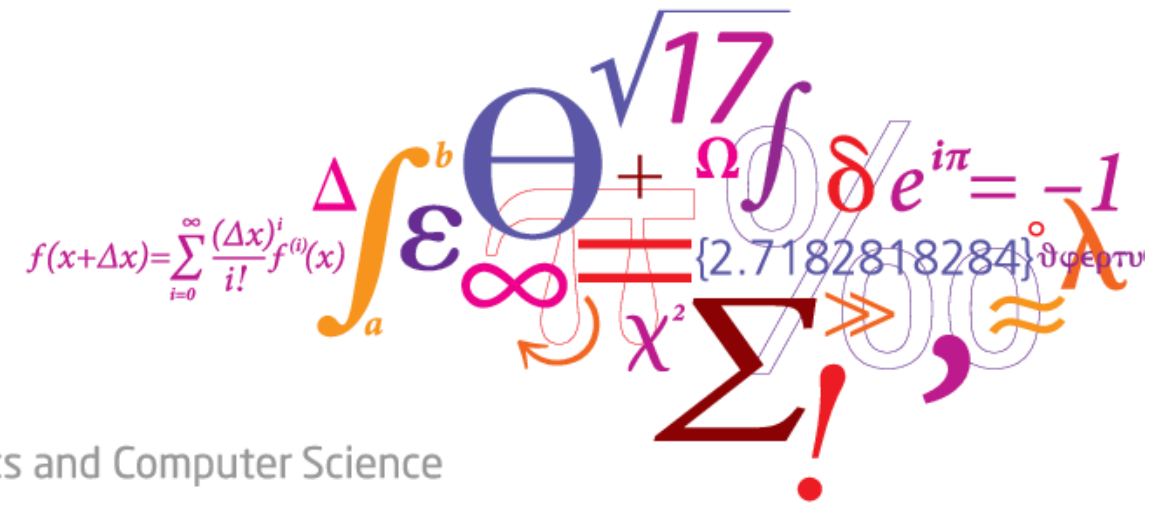

DTU

Decision Making under Uncertainty (02435)

Section for Dynamical Systems, DTU Compute.

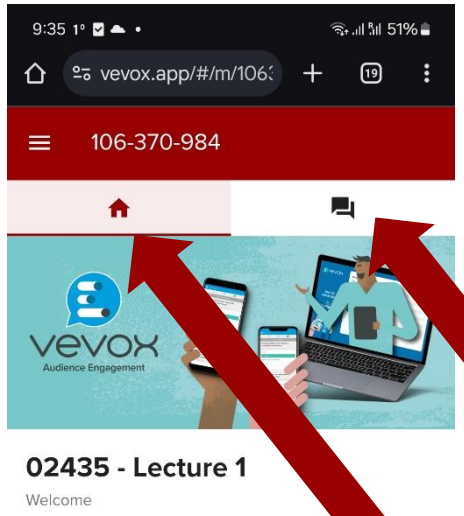


DTU Compute
Department of Applied Mathematics and Computer Science

Scan me:

Join at:
vevox.app

ID:
188-460-106



Anonymous Survey (at the end)

Anonymous Questions
(during or after the lecture)

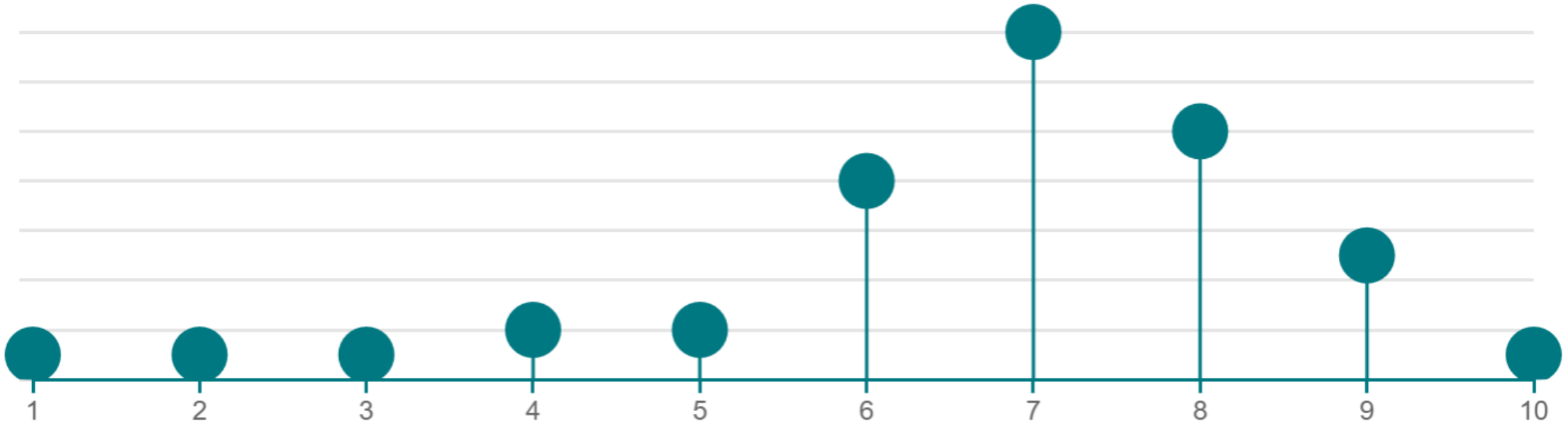
Quizzes

Feedback & Follow-up

1 How good/bad was today (scale 1-10)?

45

Mean average: 6.78



What you liked

Everything mostly.	Started understanding better the reason behind decision making under uncertainty	Good structure	Last years assingment	Quizz	Game
All great, no comments	Stochastic programming	The topic and comparison of algorithms	The flow of the lecture	Material	The examples
The content was interesting Nice that you are looking at the feedback and implementing suggestions	The use of examples	The equations and optimization part was easy to understand due to prior knowledge of julia and python.	Liked the examples and explanations.	the game !	The topic
Interactivity	the interactive way of teaching. Besides, mentioning the importance to read certain slides before class is also useful for me.	Connection with the assignments More interesting concepts introduced	The subject of the today's lecture		
Interesting stuff	Examples from other project	The quiz and the questions			
I like the concepts	Concept of stochastic optimization	The lecture			
the examples it is always a good way, but we pass by them super quick	The examples we're good in explaining the concept	Ok			
Detailed.	Well exposed theory and reenforcement of some important topics in the end	I general it was good			

What you disliked

Confusing

Could have used a better and more explicit outline of differences and trade offs of the EVP, SP and 2-Stage SP Too little time for dense questions in the quizzes, which is...

Some concepts will take me more time to grasp, so I would have preferred cramming less of them in today's lecture

The questions for the quiz are difficult to solve in 15 seconds Would appreciate releasing the same slides as presented

It would be nice to include the vevox number not just the qr-code.

No focus at all on python

The response time for some questions was too short

I feel I would understand the content much better given a well-written 5 pages, that I could read in a much shorter time Given your reactions to people's quiz answer...

I did not have enough time to read and answer in the quiz

The process part and details of 2-stage and why we do it? and why it's relevant to know the number of stages and variables was all confusing (basically the theory...

Still a little too fast to follow all the time

Too much QR coding

Why cannot we put all questions in one vevox account? It would be better with the meeting number at the side as well when you show the QR code as I prefer do it on ...

The coffee example as a 2 stage stochastic programming problem was introduced a bit extra quick for me

A bit confusing lecture. It seemed that at lot of the content was unclearly mentioned, and simple things were over complicated.

Too fast explaining

A bit difficult

I personally would like more mathematical definitions Also, I think getting grade bonus based on the quizzes is not the best idea. Instead maybe consider handing...

The example demonstrated for stochastic optimization

More comprehensive slides needed for study

it started to get more confused and the examples were given quickly the questions of the quizz should be more simple for the time we have to reply

None

Would be nice to see some actual coding going through these examples as well.

But difficult

Nothing, today was good

Was just bored at some point

Not a lot of theory

Fast

Nothing

I would like some more time explaining the coffee problem

This was way to quick - I'm uncertain where to even begin

Too fast.

It went a bit fast

The pace

I find it hard to implement in code even though I think I understand the principles.

Would be nice if the quiz is without time...

A little bit difficult to understand the concept

Plan

→ ~~Task 0~~

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

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

→ Today: Task 2

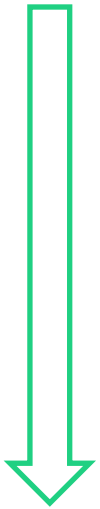
Multi-stage Stochastic Programming + caveats

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

→ Weeks 6-7: Task 3
Approximate Dynamic Programming

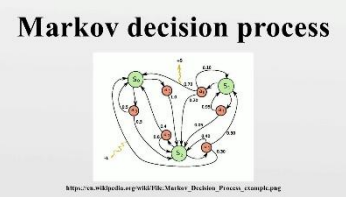
→ Week 8: Assignment Work for Task 3 and Q&A

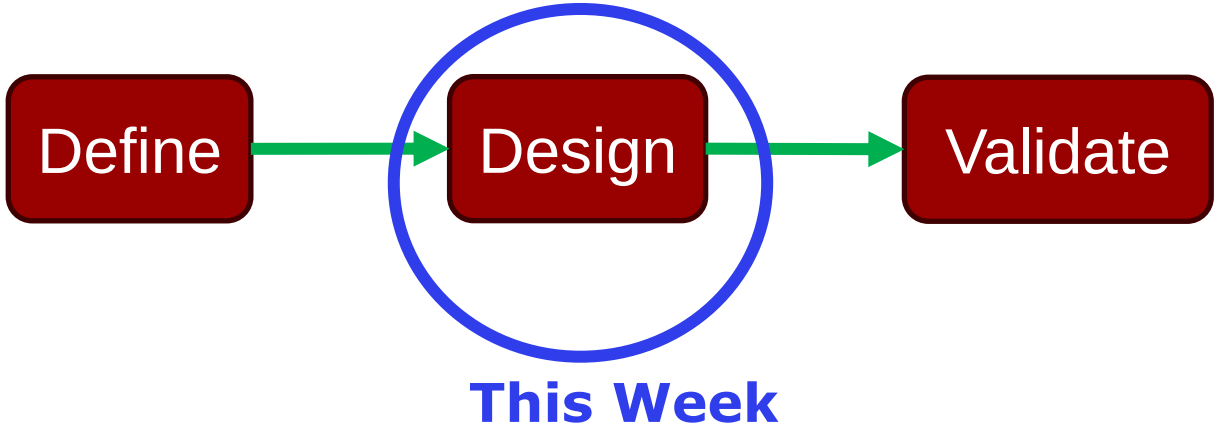
→ Weeks 9-11: Assignment B
Robust Optimization



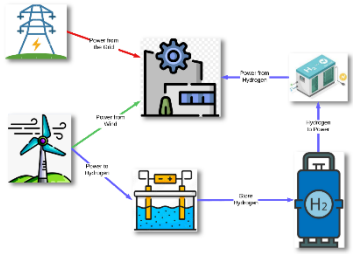
Task 4 is about reporting the results from Tasks 2 and 3

The process of designing “Decision-making” frameworks





Coding a simulation Environment to evaluate *any* decision-making policy



Agenda for today

1. Quick Stochastic Programming recap
2. Complexity and Scenario Reduction
3. Multi-stage Stochastic Programming

2-stage Stochastic Programming

In a two-stage problem (Day 1 & Day 2), the decisions in Day 1 are made using a two-stage stochastic program.

76

In Day 2 (after the uncertainty has been revealed):

we completely disregard the Stochastic Program's Day 2 decisions, and solve a new optimization problem to decide what to do on Day 2.

39.47%

we look at which scenario has been realized and apply the Stochastic Program's decisions that corespond to that scenario.

43.42%

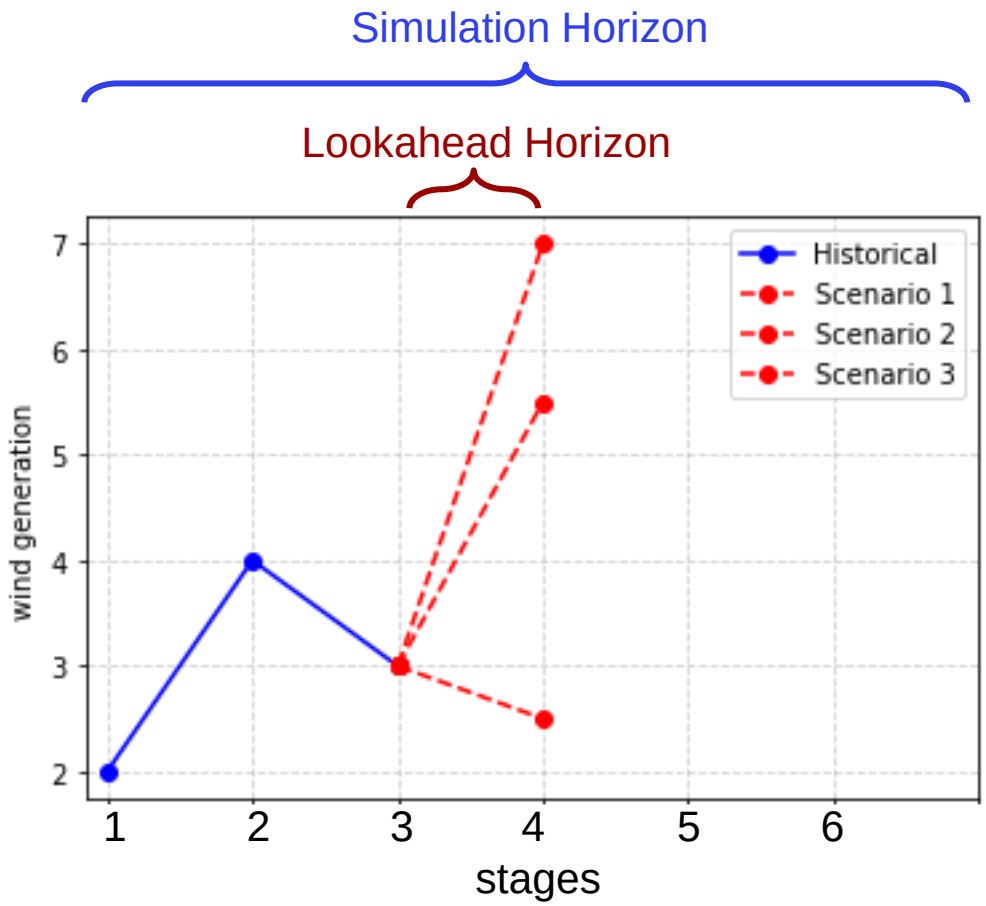
we apply the Day 2 decisions that would have been, if the Stochastic Program was aware of the uncertainty realization beforehand.

17.11%

Allowed selections: 1

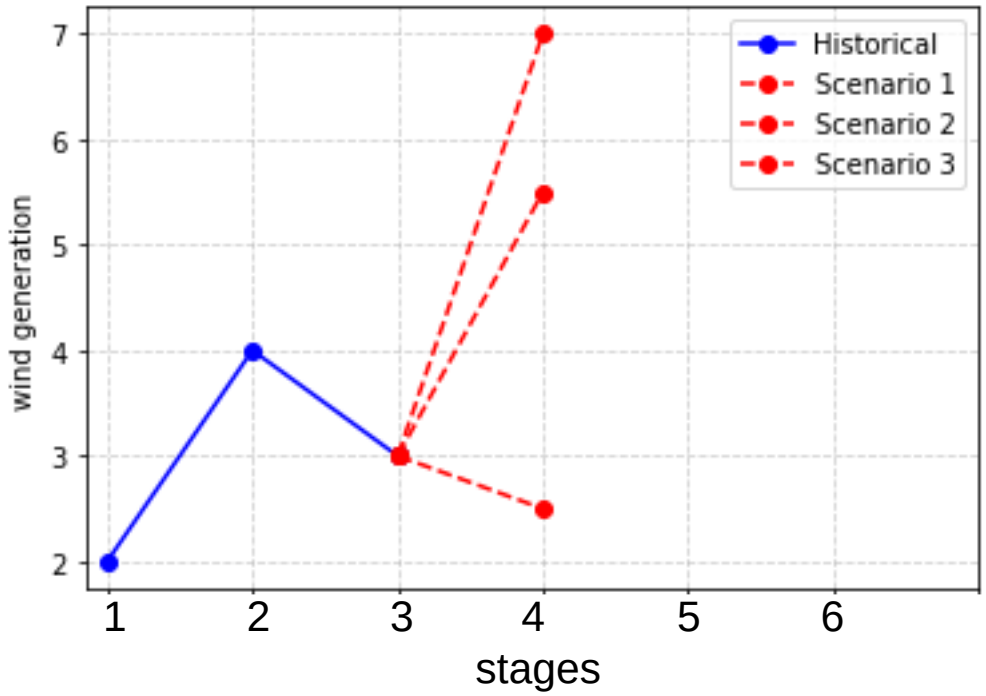
vevox Audience Engagement

2-stage Stochastic Programming

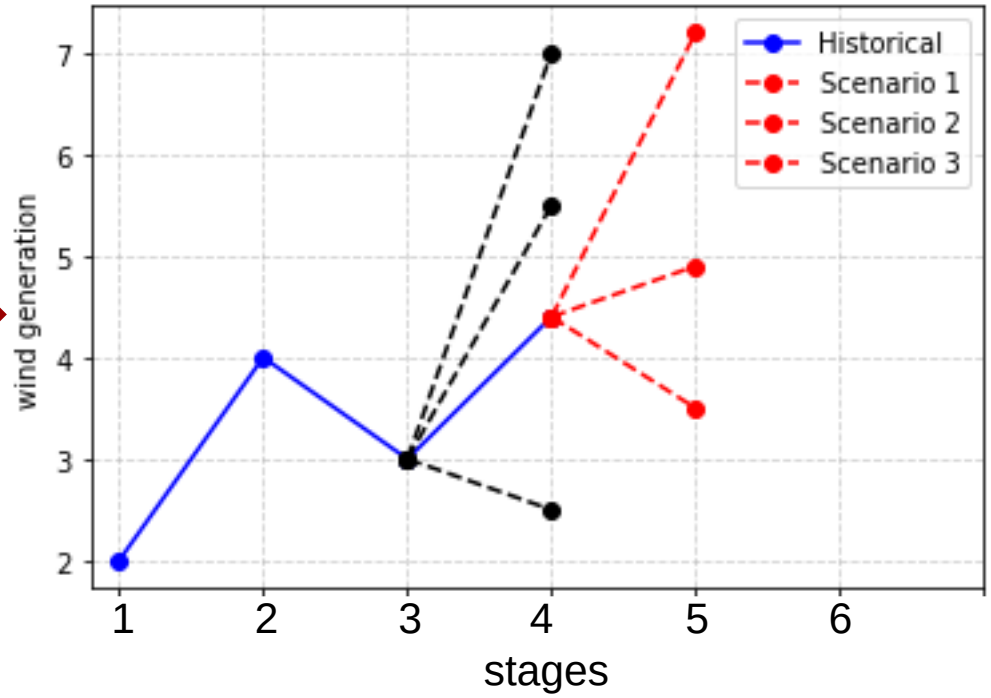


2-stage Stochastic Programming

Let's say we are in stage 3. We observe the current state and make predictions (scenarios) for the state at stage 4, to inform the here-and-now decision



When stage 4 comes, we observe the new state and do the same to calculate the here-and-now decision for stage 4.



So, we solve an optimization problem again, and do not use the previously made prospective decisions.

How to generate scenarios

Not in this course

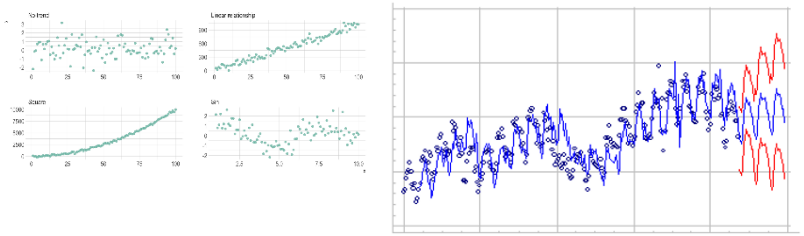
Previous values

Data

Forecasting Model

wind_process

price_process



Next values

Number of Scenarios

wind_process

price_process

With 2 random variables and 3 possible realizations for each, we have… how many scenarios?

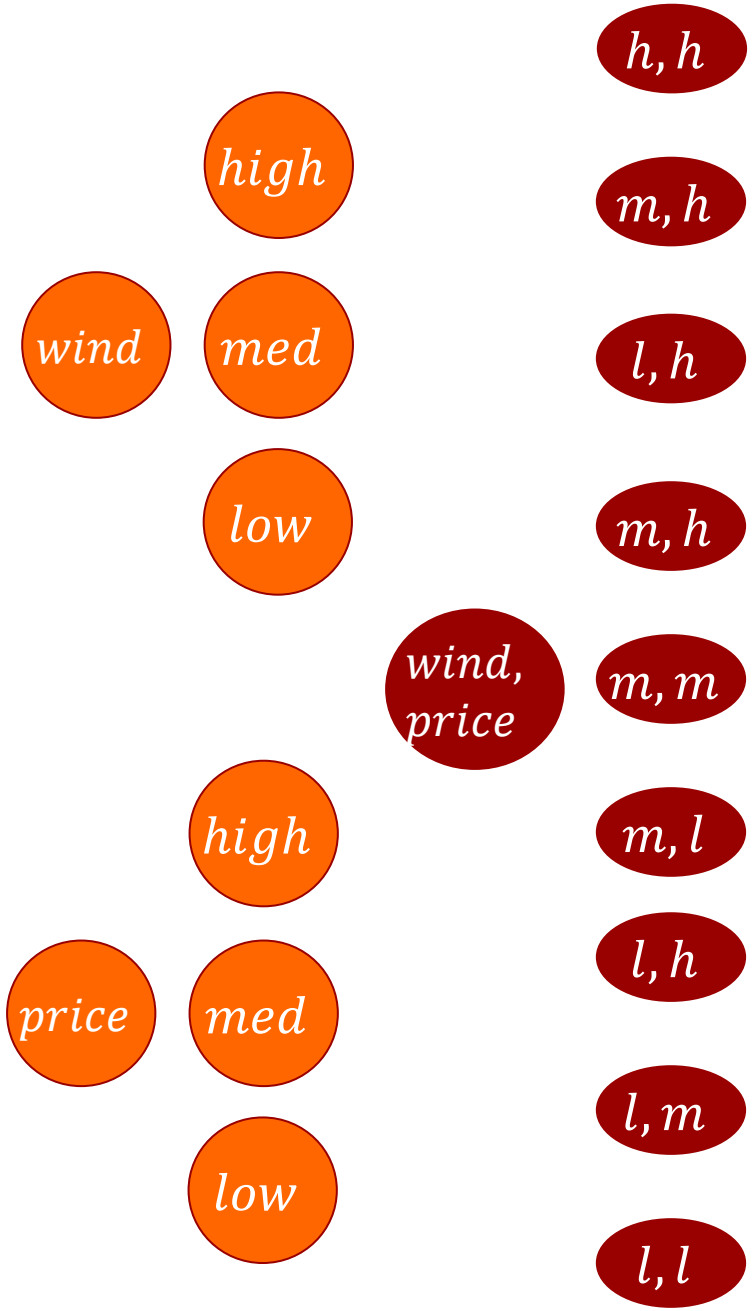
Number of Scenarios

wind_process

price_process

With 2 random variables and 3 possible realizations for each, we have $3^2 = 9$ possible combinations (scenarios)

With V random variables and B possible realizations (branches) for each, we have B^V scenarios



More scenarios mean more variables.

More variables mean longer computational time.

But you need to make a decision before the next stage arrives (e.g. within 1 hour/ 1 day)

Coffee Problem Nomenclature

Join at:
vevox.app

ID:
188-460-106

Sets

- $w, q \in W$: set of warehouses, where w and q belong to $\{1, 2, 3\}$
- $t \in T$: set of timeslots (daily), in $\{1, 2, ...\}$

Parameters

- $D_{w,t}$: coffee demand for warehouse w in period t, being constant at 4 units per day
- $C_w^{storage}$: storage capacity limit for warehouse w
- $C_{w,q}^{transp}$: daily transportation capacity limit for what warehouse w can send to warehouse q, where if $w = q$ then the capacity is zero
- $p_{w,t}$: external coffee price for warehouse w at time t, continuous in $[0, 10]$
- $e_{w,q}$: transportation per-unit cost between warehouse w and q, where if $w = q$ then the cost is zero
- b_w: per-unit cost of missing daily demand for warehouse w

Variables

- $x_{w,t}$: continuous, amount of coffee ordered in timeslot t by warehouse w
- $z_{w,t}$: continuous, amount of coffee stored by the end of timeslot t in warehouse w
- $m_{w,t}$: continuous, amount of coffee missing when daily demand is not met in warehouse w in t
- $y_{w,q,t}^{send}$: continuous, amount of coffee sent by warehouse w to q in timeslot t
- $y_{w,q,t}^{receive}$: continuous, amount of coffee received by warehouse w from q in timeslot t

For T = 24, how many *variables* does this program have (roughly)?

5 types of variables

Coffee Problem Nomenclature

Sets

- $w, q \in W$: set of warehouses, where w and q belong to $\{1, 2, 3\}$
- $t \in T$: set of timeslots (daily), in $\{1, 2, ...\}$

Parameters

- $D_{w,t}$: coffee demand for warehouse w in period t, being constant at 4 units per day
- $C_w^{storage}$: storage capacity limit for warehouse w
- $C_{w,q}^{transp}$: daily transportation capacity limit for what warehouse w can send to warehouse q, where if $w = q$ then the capacity is zero
- $p_{w,t}$: external coffee price for warehouse w at time t, continuous in $[0, 10]$
- $e_{w,q}$: transportation per-unit cost between warehouse w and q, where if $w = q$ then the cost is zero
- b_w: per-unit cost of missing daily demand for warehouse w

Variables

3x24 • $x_{w,t}$: continuous, amount of coffee ordered in timeslot t by warehouse w

3x24 • $z_{w,t}$: continuous, amount of coffee stored by the end of timeslot t in warehouse w

3x24 • $m_{w,t}$: continuous, amount of coffee missing when daily demand is not met in warehouse w in t

3x3x24 • $y_{w,q,t}^{send}$: continuous, amount of coffee sent by warehouse w to q in timeslot t

3x3x24 • $y_{w,q,t}^{receive}$: continuous, amount of coffee received by warehouse w from q in timeslot t

How many variables does this program have?

$\sim O(|W|^2|T|)$

Coffee Problem 2-stage Stochastic Program

Sets

- $s \in S$: set of scenarios, in $\{1, 2, ...\}$

Parameters

- $p1_w$: external coffee price for warehouse w on day 1, continuous in $[0, 10]$
- $p2_{w,s}$: external coffee price for warehouse w on day 2 in scenario s, continuous in $[0, 10]$
- $prob_s$: probability of occurrence of scenario s.
- $z0_w$: initial stock for each warehouse in $t = 1$.

The previously defined variables had to be separated into 1^{st} stage (equal for all scenarios) and 2^{nd} stage (different for each scenario) in the following way:

Variables : First $(t = 1)$ and Second $(t = 2)$ Stage Decision

- $x1_w$: continuous, amount of coffee ordered in timeslot 1 by warehouse w.
- $x2_{w,s}$: continuous, amount of coffee ordered in timeslot 2 by warehouse w in scenario s.
- $z1_w$: continuous, amount of coffee stored by the end of timeslot 1 in warehouse w.
- $z2_{w,s}$: continuous, amount of coffee stored by the end of timeslot 2 in warehouse w in scenario s.
- $m1_w$: continuous, amount of coffee missing when daily demand is not met in warehouse w in timeslot 1.
- $m2_{w,s}$: continuous, amount of coffee missing when daily demand is not met in warehouse w in timeslot 2 in scenario s.
- $y1^{send}_{w,q}$: continuous, amount of coffee sent by warehouse w to q in timeslot 1.
- $y2^{send}_{w,q,s}$: continuous, amount of coffee sent by warehouse w to q in timeslot 2 in scenario s.
- $y1^{receive}_{w,q}$: continuous, amount of coffee received by warehouse w from q in timeslot 1.
- $y2^{receive}_{w,q,s}$: continuous, amount of coffee received by warehouse w from q in timeslot 2 in scenario s.

Coffee Problem 2-stage Stochastic Program

Sets

- $s \in S$: set of scenarios, in $\{1, 2, ...\}$

Parameters

- $p1_w$: external coffee price for warehouse w on day 1, continuous in $[0, 10]$
- $p2_{w,s}$: external coffee price for warehouse w on day 2 in scenario s, continuous in $[0, 10]$
- $prob_s$: probability of occurrence of scenario s.
- $z0_w$: initial stock for each warehouse in $t = 1$.

The previously defined variables had to be separated into 1^{st} stage (equal for all scenarios) and 2^{nd} stage (different for each scenario) in the following way:

Variables : First $(t = 1)$ and Second $(t = 2)$ Stage Decision

- $x1_w$: continuous, amount of coffee ordered in timeslot 1 by warehouse w.
- $x2_{w,s}$: continuous, amount of coffee ordered in timeslot 2 by warehouse w in scenario s.
- $z1_w$: continuous, amount of coffee stored by the end of timeslot 1 in warehouse w.
- $z2_{w,s}$: continuous, amount of coffee stored by the end of timeslot 2 in warehouse w in scenario s.
- $m1_w$: continuous, amount of coffee missing when daily demand is not met in warehouse w in timeslot 1.
- $m2_{w,s}$: continuous, amount of coffee missing when daily demand is not met in warehouse w in timeslot 2 in scenario s.
- $y1^{send}_{w,q}$: continuous, amount of coffee sent by warehouse w to q in timeslot 1.
- $y2^{send}_{w,q,s}$: continuous, amount of coffee sent by warehouse w to q in timeslot 2 in scenario s.
- $y1^{receive}_{w,q}$: continuous, amount of coffee received by warehouse w from q in timeslot 1.
- $y2^{receive}_{w,q,s}$: continuous, amount of coffee received by warehouse w from q in timeslot 2 in scenario s.

Alternatively: same type of variable with more indices: $x_{w,t,s}$

And impose the constraints:

$x_{w,1,s} = x_{w,1,s'}, \quad \forall w \in W, (s, s') \in S$

Coffee Problem 2-stage Stochastic Program

Sets

- $s \in S$: set of scenarios, in $\{1, 2, ...\}$

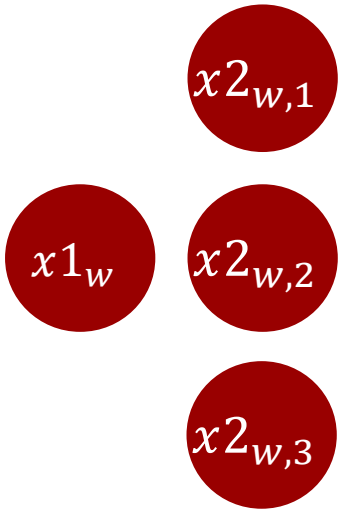
Parameters

- $p1_w$: external coffee price for warehouse w on day 1, continuous in $[0, 10]$
- $p2_{w,s}$: external coffee price for warehouse w on day 2 in scenario s, continuous in $[0, 10]$
- $prob_s$: probability of occurrence of scenario s.
- $z0_w$: initial stock for each warehouse in $t = 1$.

The previously defined variables had to be separated into 1^{st} stage (equal for all scenarios) and 2^{nd} stage (different for each scenario) in the following way:

Variables : First $(t = 1)$ and Second $(t = 2)$ Stage Decision

- $x1_w$: continuous, amount of coffee ordered in timeslot 1 by warehouse w.
- $x2_{w,s}$: continuous, amount of coffee ordered in timeslot 2 by warehouse w in scenario s.



Alternatively: same type of variable with more indices: $x_{w,t,s}$

And impose the constraints:

$x_{w,1,s} = x_{w,1,s'}, \quad \forall w \in W, (s, s') \in S$

Coffee Problem 2-stage Stochastic Program

Sets

- $s \in S$: set of scenarios, in $\{1, 2, ...\}$

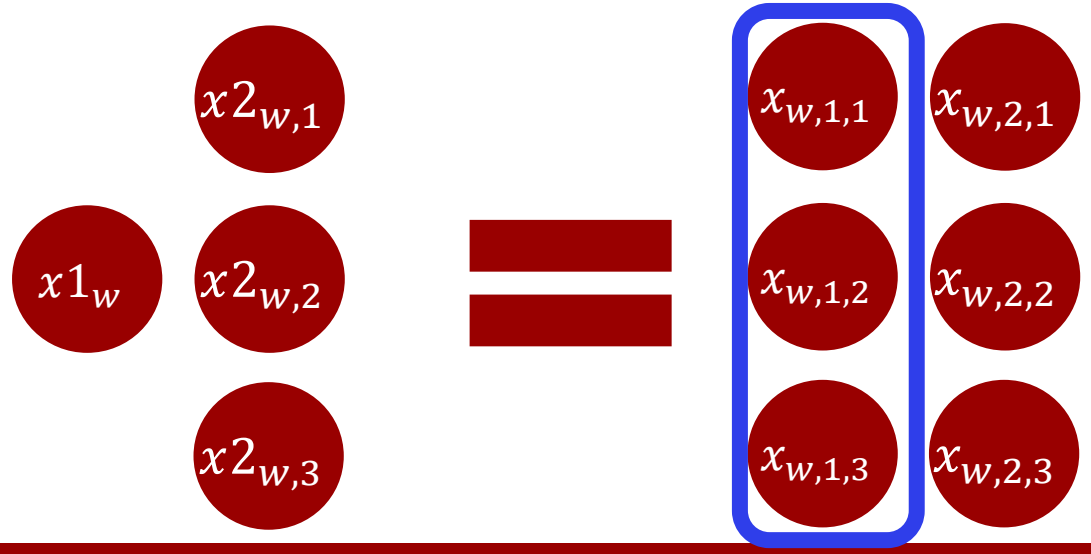
Parameters

- $p1_w$: external coffee price for warehouse w on day 1, continuous in $[0, 10]$
- $p2_{w,s}$: external coffee price for warehouse w on day 2 in scenario s, continuous in $[0, 10]$
- $prob_s$: probability of occurrence of scenario s.
- $z0_w$: initial stock for each warehouse in $t = 1$.

The previously defined variables had to be separated into 1^{st} stage (equal for all scenarios) and 2^{nd} stage (different for each scenario) in the following way:

Variables : First $(t = 1)$ and Second $(t = 2)$ Stage Decision

- $x1_w$: continuous, amount of coffee ordered in timeslot 1 by warehouse w.
- $x2_{w,s}$: continuous, amount of coffee ordered in timeslot 2 by warehouse w in scenario s.



Alternatively: same type of variable with more indices: $x_{w,t,s}$

And impose the constraints:

$x_{w,1,s} = x_{w,1,s'}, \quad \forall w \in W, (s, s') \in S$

Coffee Problem 2-stage Stochastic Program

Sets

- $s \in S$: set of scenarios, in $\{1, 2, ...\}$

Variables:

$x_{w,t,s}$

$z_{w,t,s}$

$m_{w,t,s}$

$y^{send}_{w,q,t,s}$

$y^{receive}_{w,q,t,s}$

If the previous (deterministic) program had roughly $O(|W|^2|T|)$ variables, how many variables does this 2-stage stochastic program have?

Coffee Problem 2-stage Stochastic Program

Sets

- $s \in S$: set of scenarios, in $\{1, 2, ...\}$

Variables:

$x_{w,t,s}$

$z_{w,t,s}$

$m_{w,t,s}$

$y^{send}_{w,q,t,s}$

$y^{receive}_{w,q,t,s}$

If the previous (deterministic) program had roughly $O(|W|^2|T|)$ variables, how many variables does this 2-stage stochastic program have?

$$\sim O(|W|^2|T||S|)$$

$$|W| = 3$$
$$|T| = 2$$

How many scenarios can we afford?

How many would we like to have?

How many scenarios can we afford?

How many variables can we afford?

Around (say) 10,000

How many variables do we have?

$\sim O(|W|^2|T||S|)$

$|W| = 3$

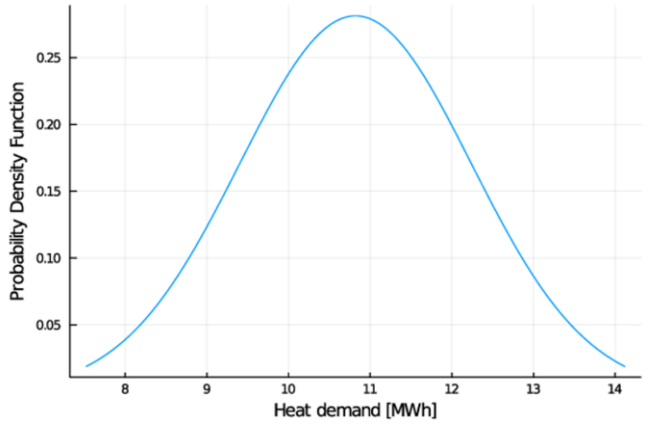
$|T| = 2$

Around 18 x number_of_scenarios

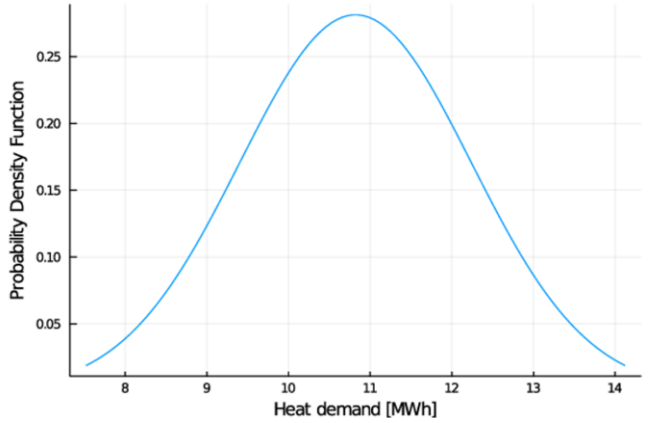
So, how many scenarios can we afford?

Around 10,000 / 18 =~ 555

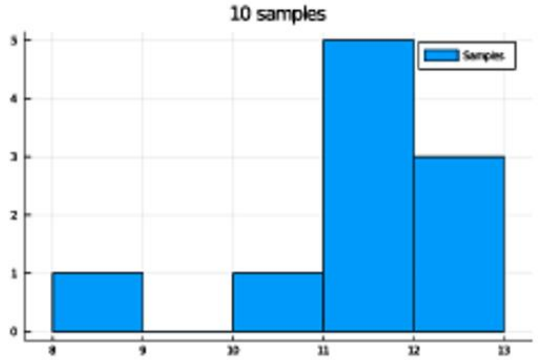
How many scenarios would we like to have?

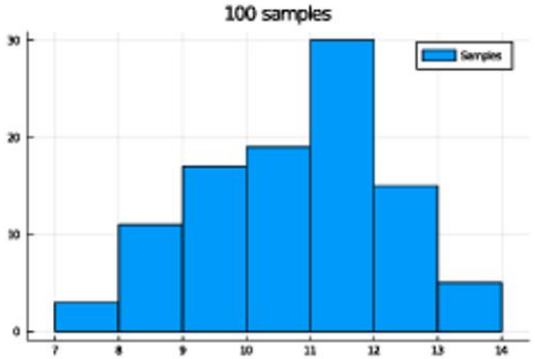


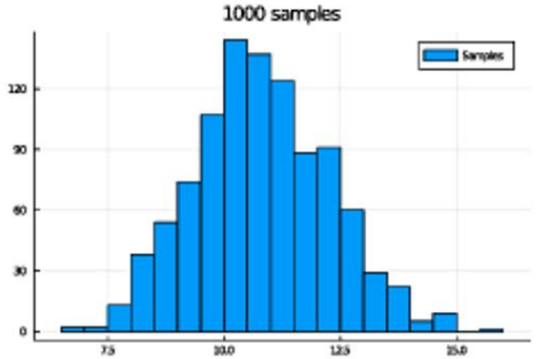
How many scenarios would we like to have?

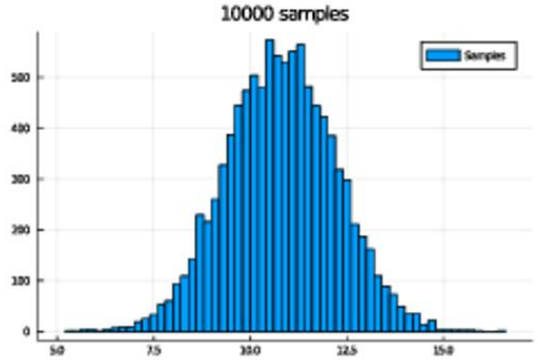


Different sample sizes for the heat demand example



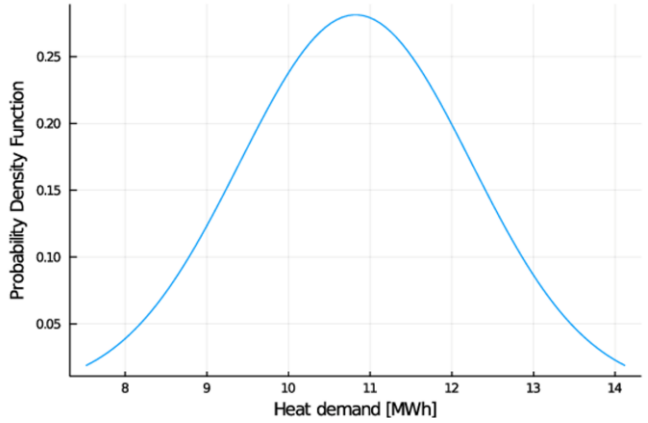






A large number of samples is required get a representative sample set with similar properties as the original distribution.

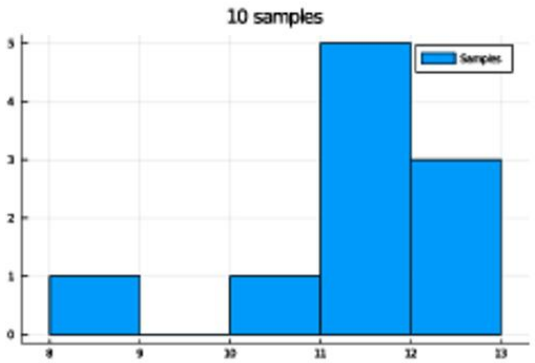
How many scenarios would we like to have?

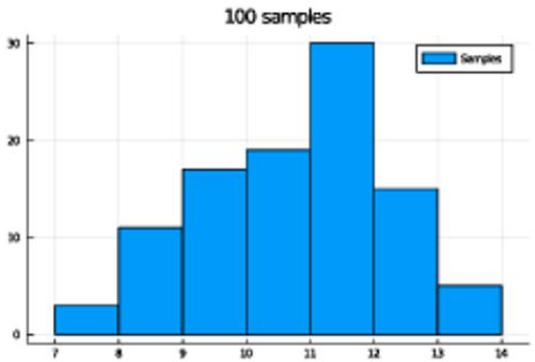


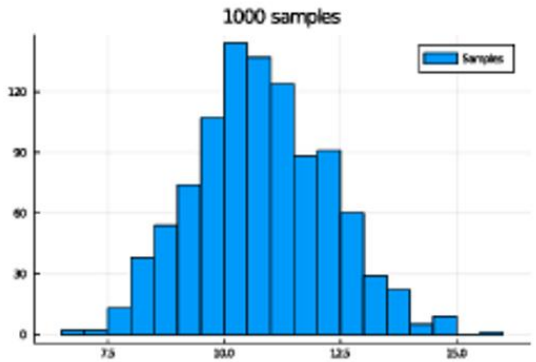
We would like 100-1000 samples to capture the distribution (for a single random variable)…

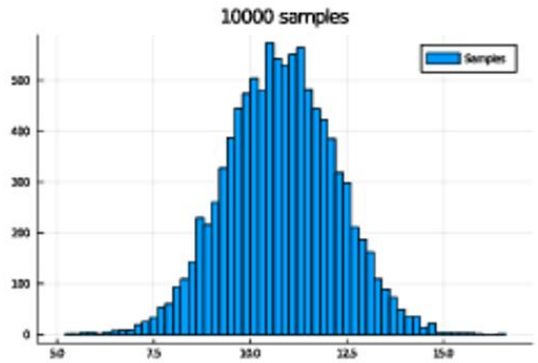
But, recall, $|S| = B^V$

Different sample sizes for the heat demand example









A large number of samples is required get a representative sample set with similar properties as the original distribution.

We need to reduce the number of scenarios

Clustering

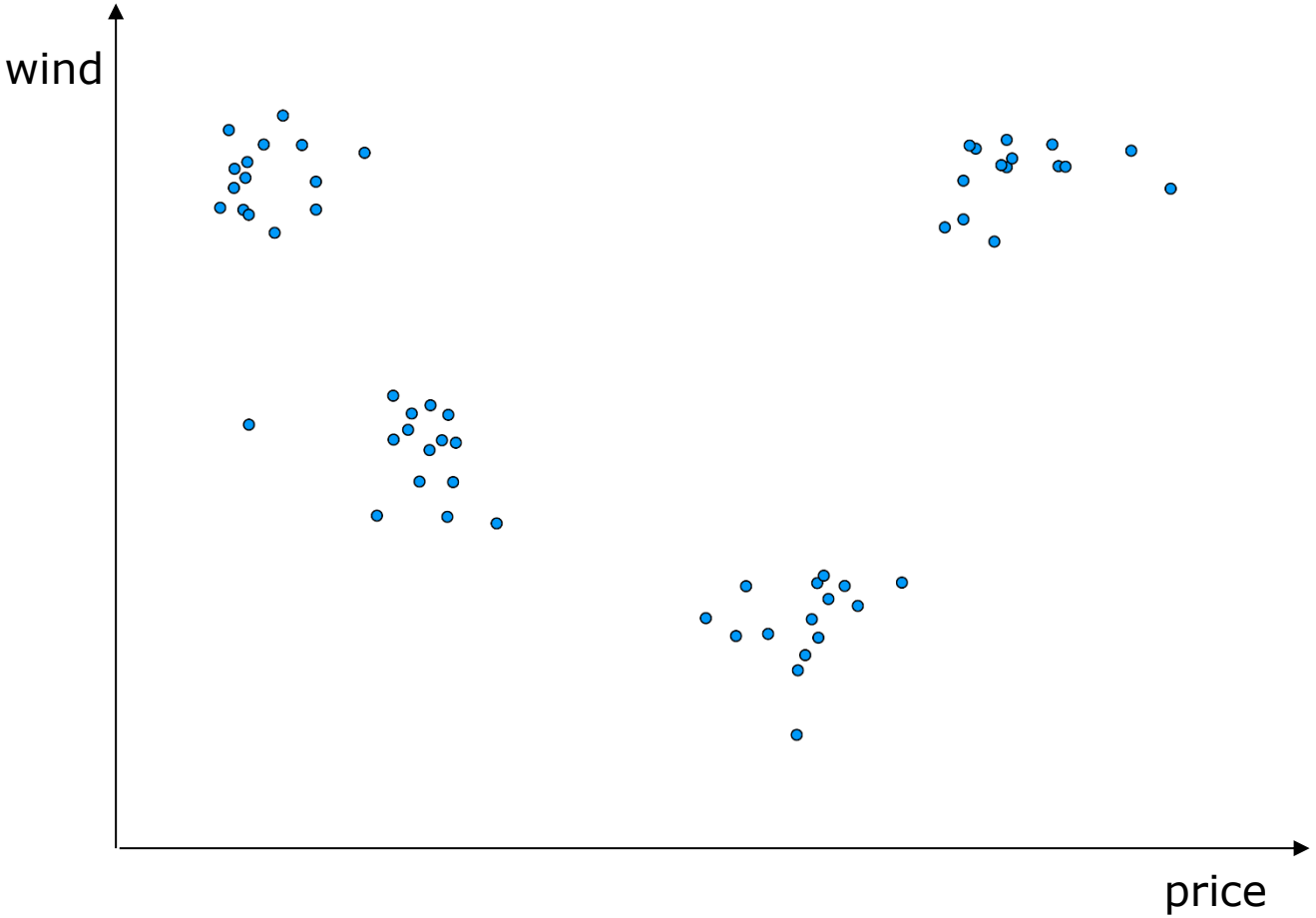
Partition a set of objects into clusters(groups) so that the objects in each cluster show a high degree of similarity, while objects belonging to different clusters are as dissimilar as possible.

Centroid-based clustering algorithms:

The clusters are represented by so-called *centroids*, i.e, cluster centers. All data points are assigned to one centroid and, thus, belong to the respective cluster.

- *k-means clustering*: Clusters the data objects into k clusters. The centroid is the mean of all data objects in the cluster.
- *k-medoid clustering / Partition around medoids (PAM)*: similar to k-means, but the centroids are chosen from the data objects (called medoids then)

K-means Clustering



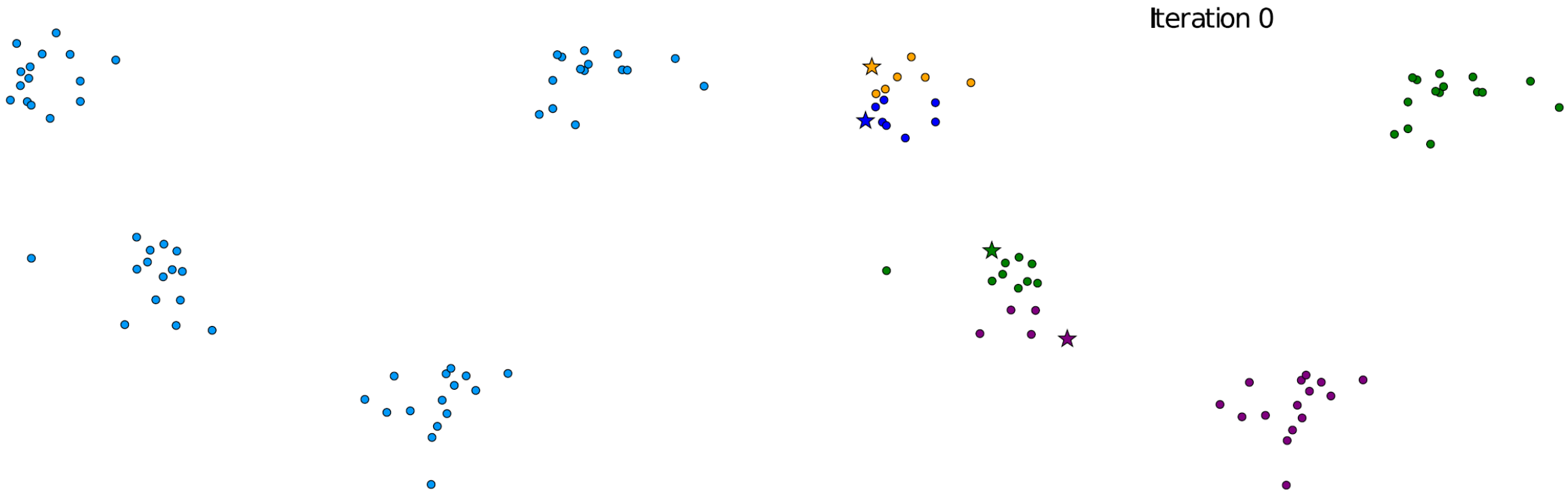
1) We generate a lot of scenarios by Monte Carlo sampling.

2) We specify how many scenarios we can afford (here, say for example: 4).

3) We reduce the initial set of scenarios to 4 representative centroids/scenarios.

4) We assign a probability to each of the 4 final scenarios.

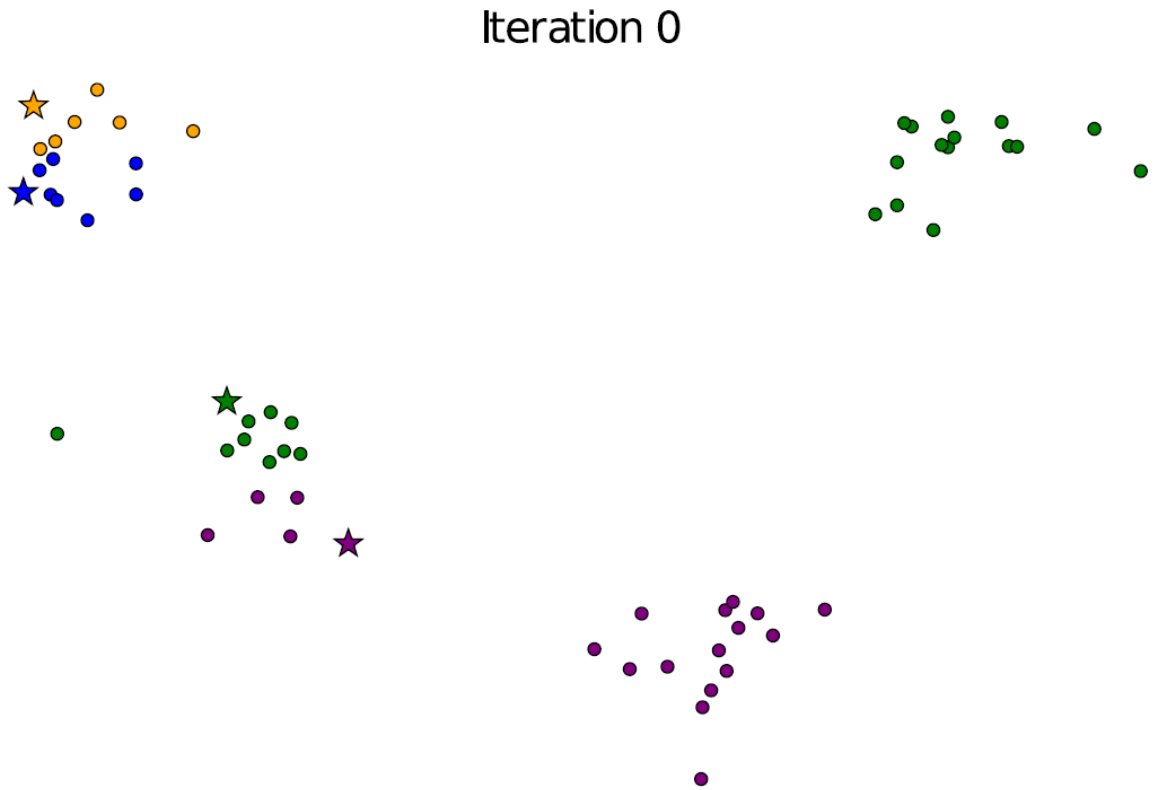
K-means Clustering

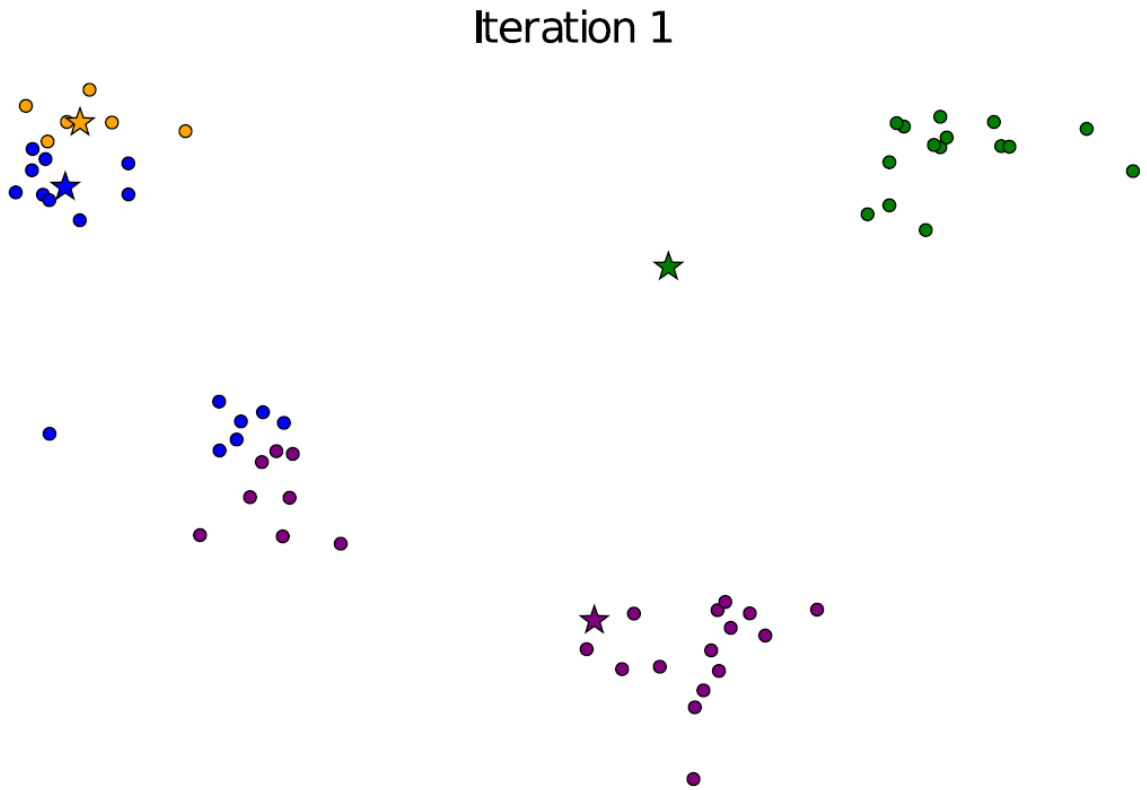


Place centroids randomly

Assign each scenario to its closest centroid

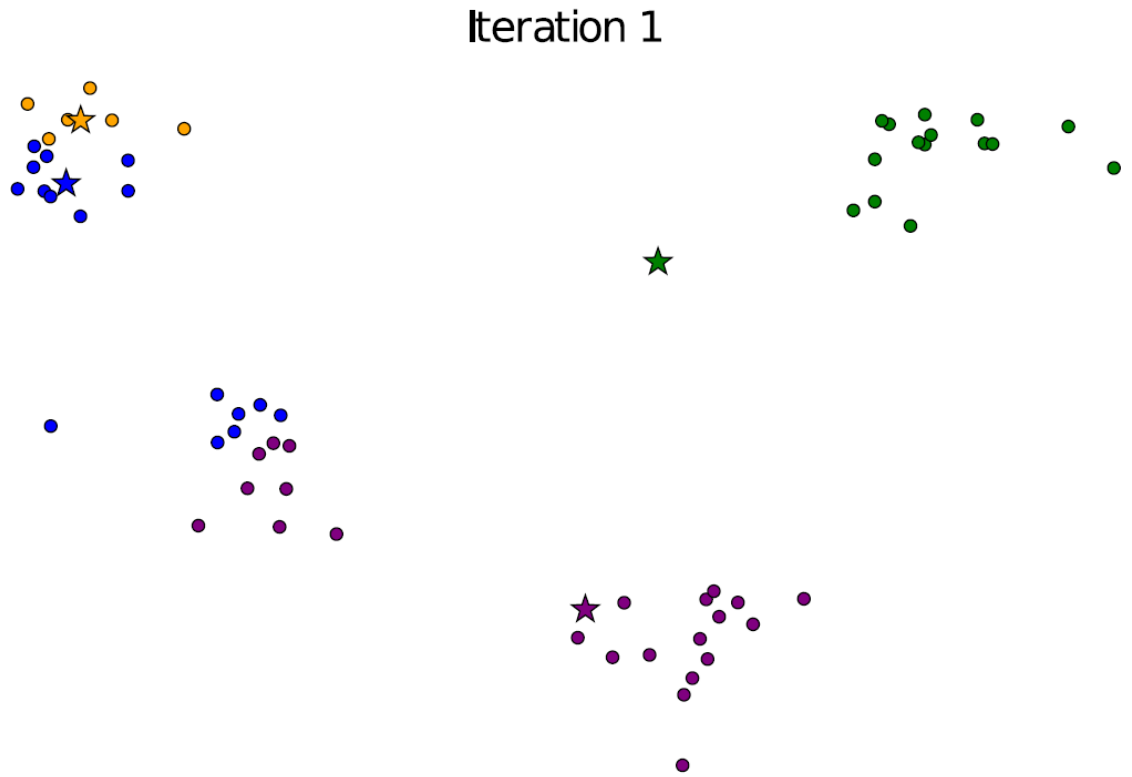
K-means Clustering

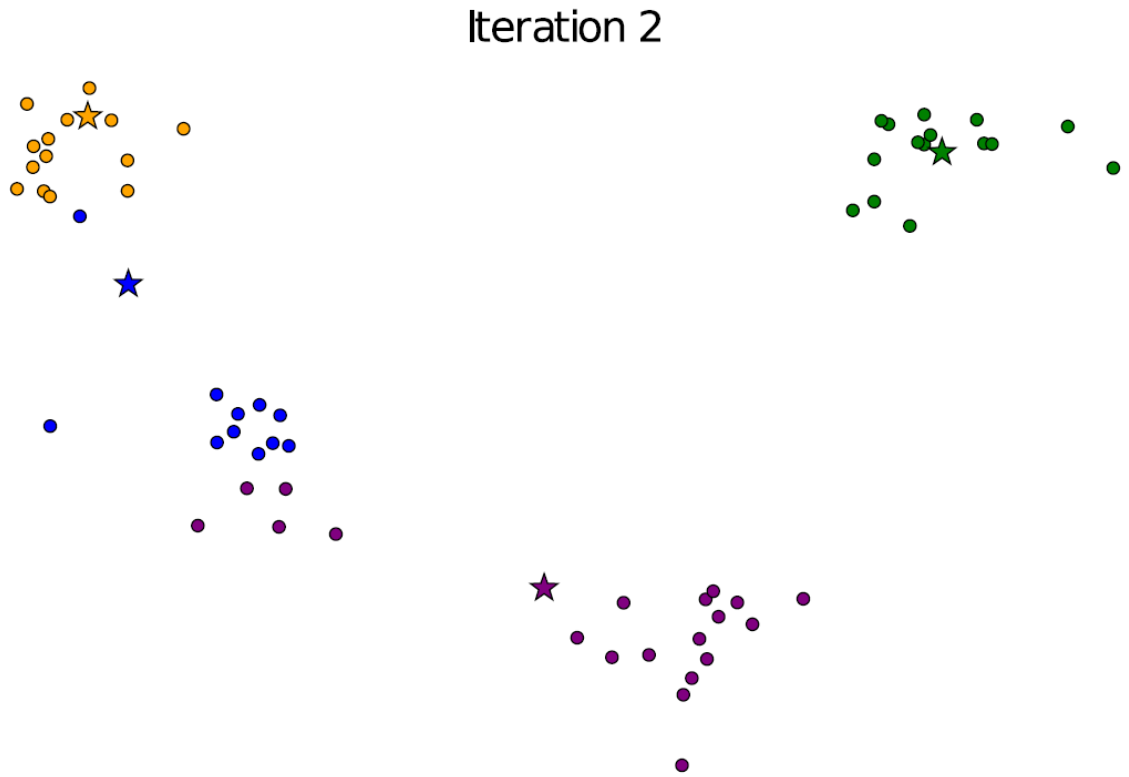




Move each centroid to the “center” of its cluster

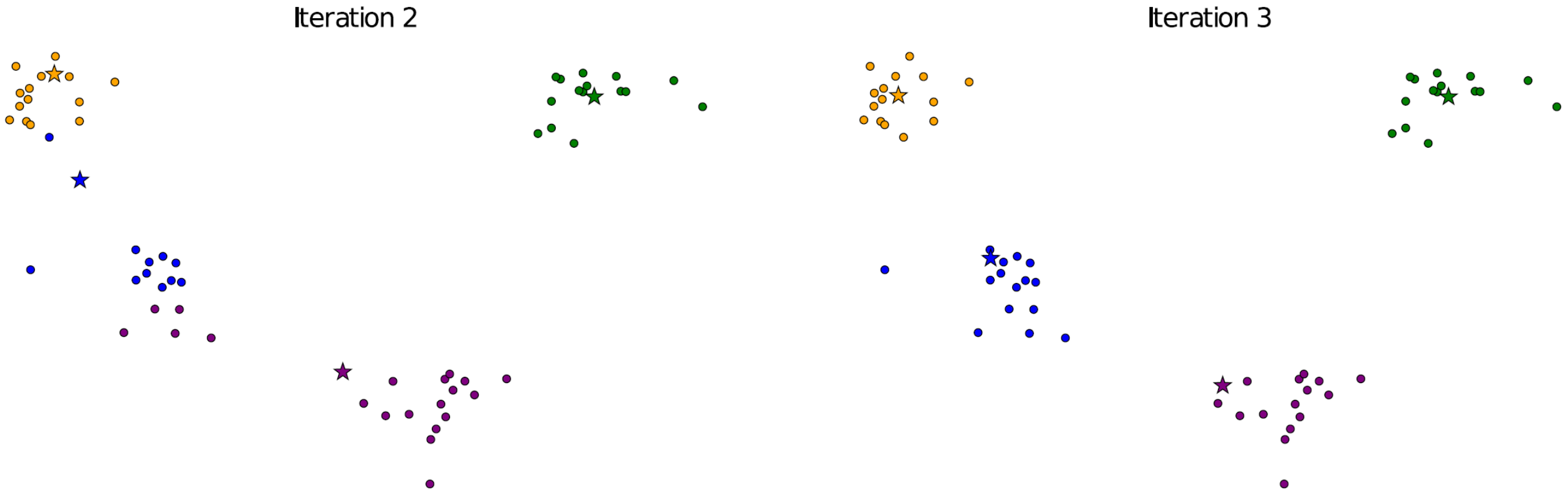
K-means Clustering



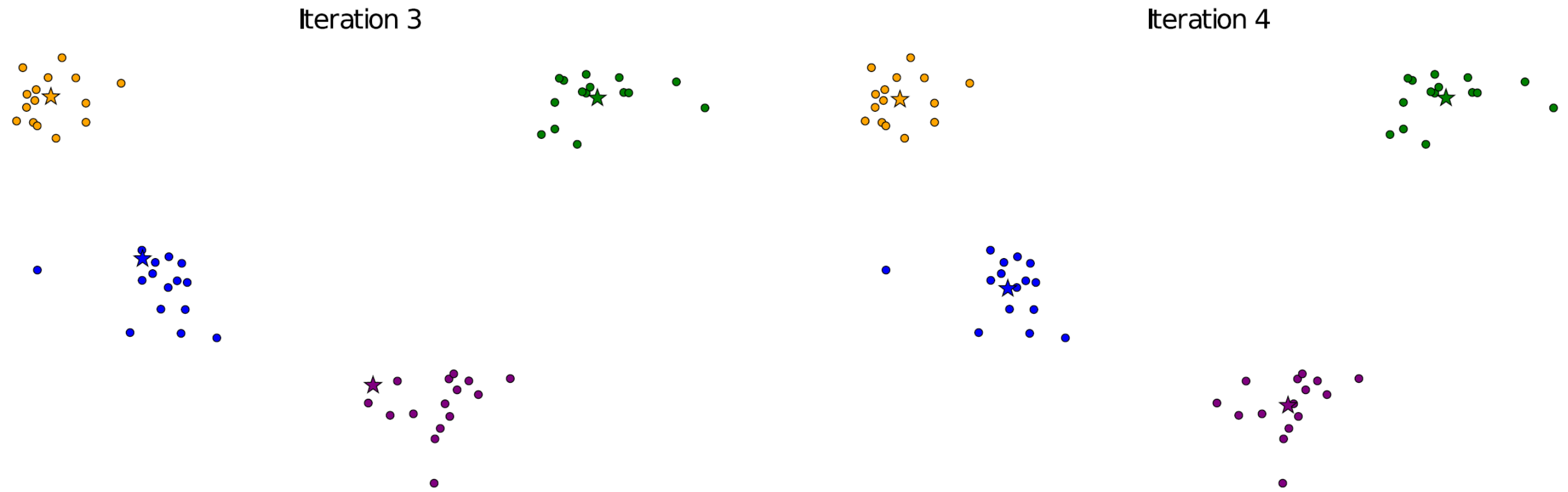


With the new centroids…
re-assign each scenario to its closest centroid

K-means Clustering

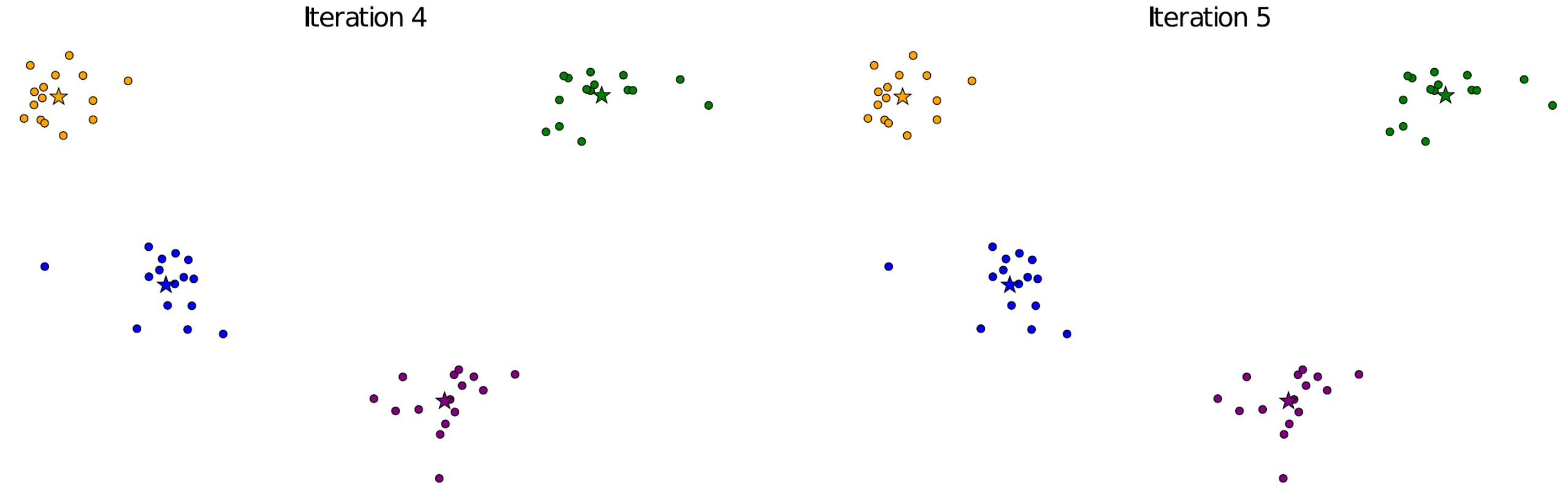


K-means Clustering



K-means Clustering

Centroids did not move → Algorithm Terminates



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:

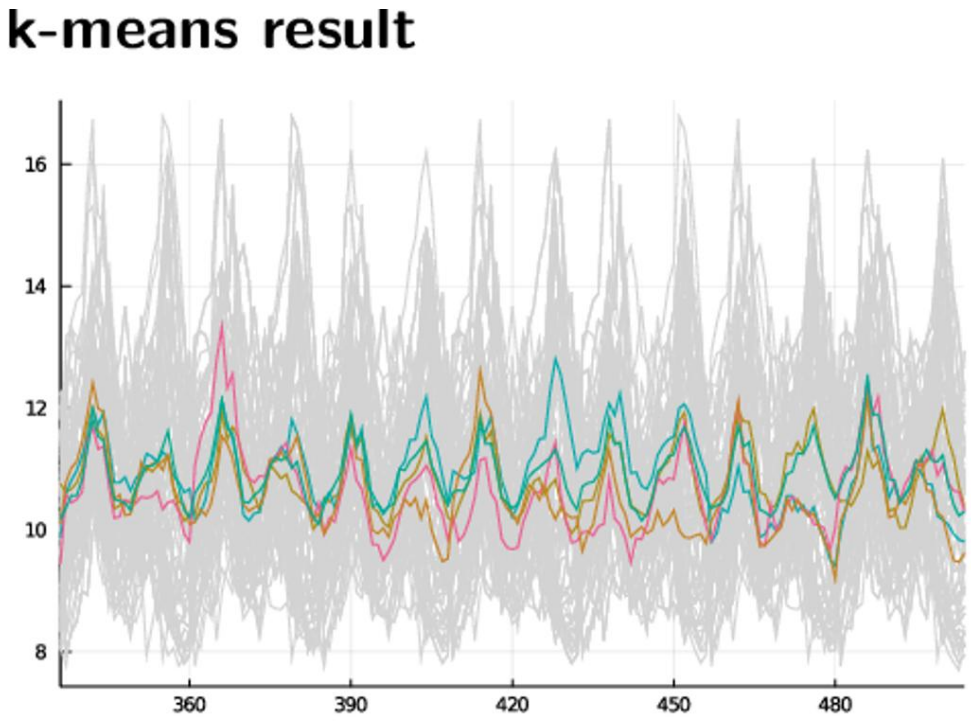
One scenario per cluster (here: the centroid/medoid)

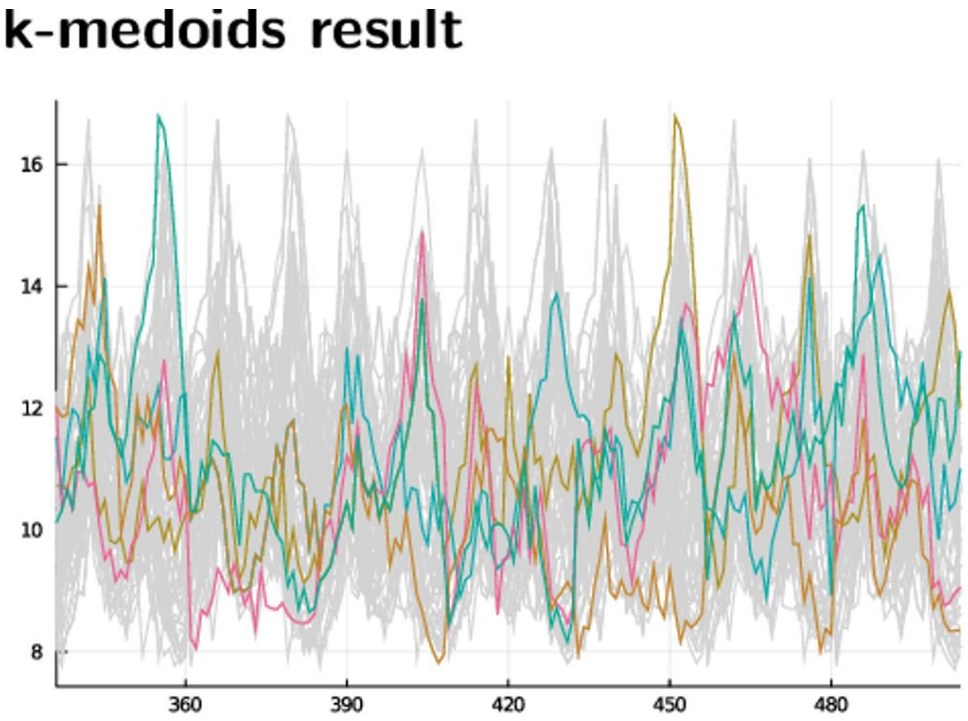
Redistribution of probabilities:

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

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

K-means vs K-medoids





k-means leads to less volatile scenarios because each time period is averaged. k-medoids still uses realistic scenarios by selecting the most represenative ones.

If we apply k-means, with k = 1, what do we end up with?

How to build a Two-Stage SP Policy

Input: Current State

Output: Here-and-now decisions

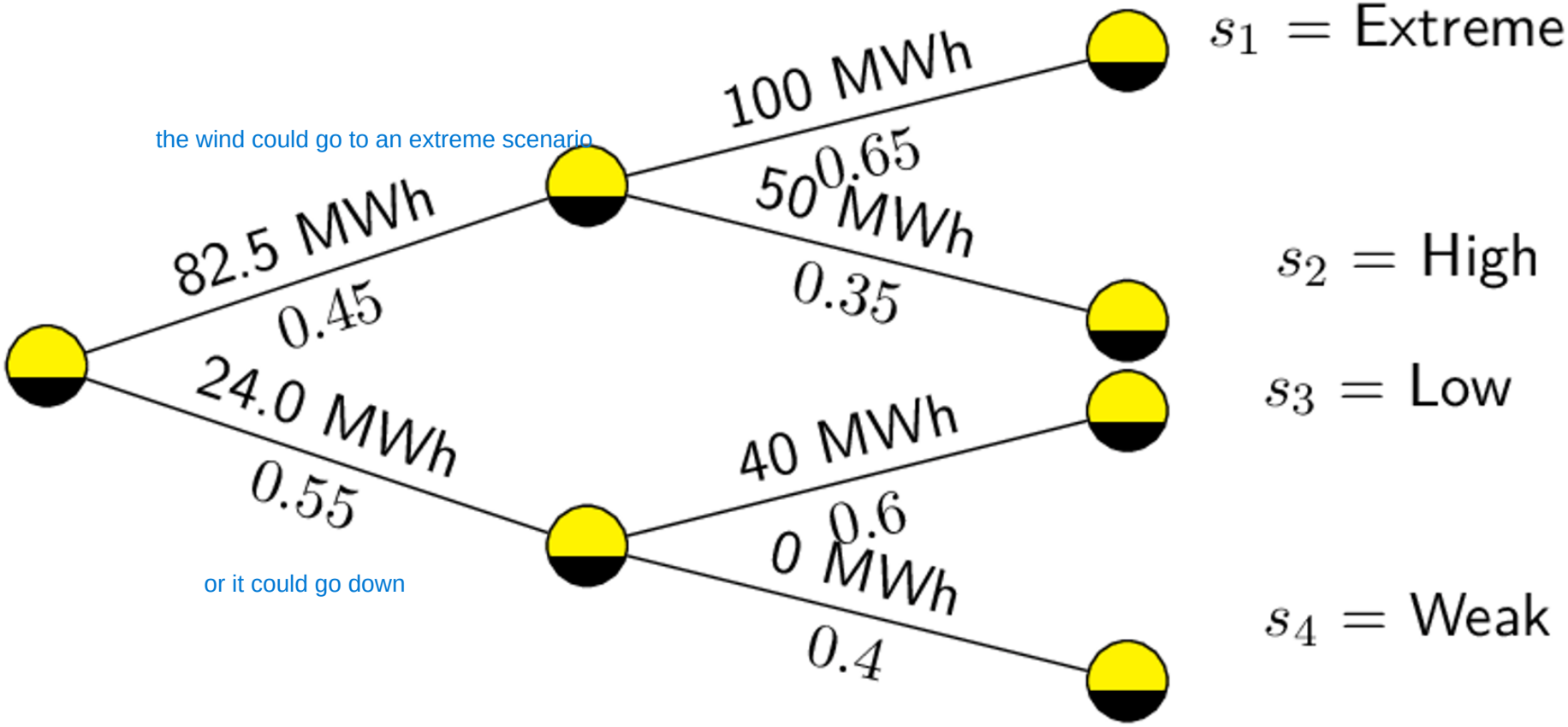
1) Assess how many variables you can afford
2) Calculate the number of scenarios:
   remember, $variable\_types * S \leq maximum\ number\ of\ variables\ you\ can\ afford$
3) Generate many scenarios (e.g. 10 or 100 times S).
4) Reduce them to S scenarios, using clustering
5) Calculate the probability of each final scenario s
   (add the probabilities of the scenarios in cluster s)
6) Solve the 2-stage Stochastic Optimization problem
7) Return the here-and-now decisions (only)

Decision-making under uncertainty

Multi-stage Stochastic Programming

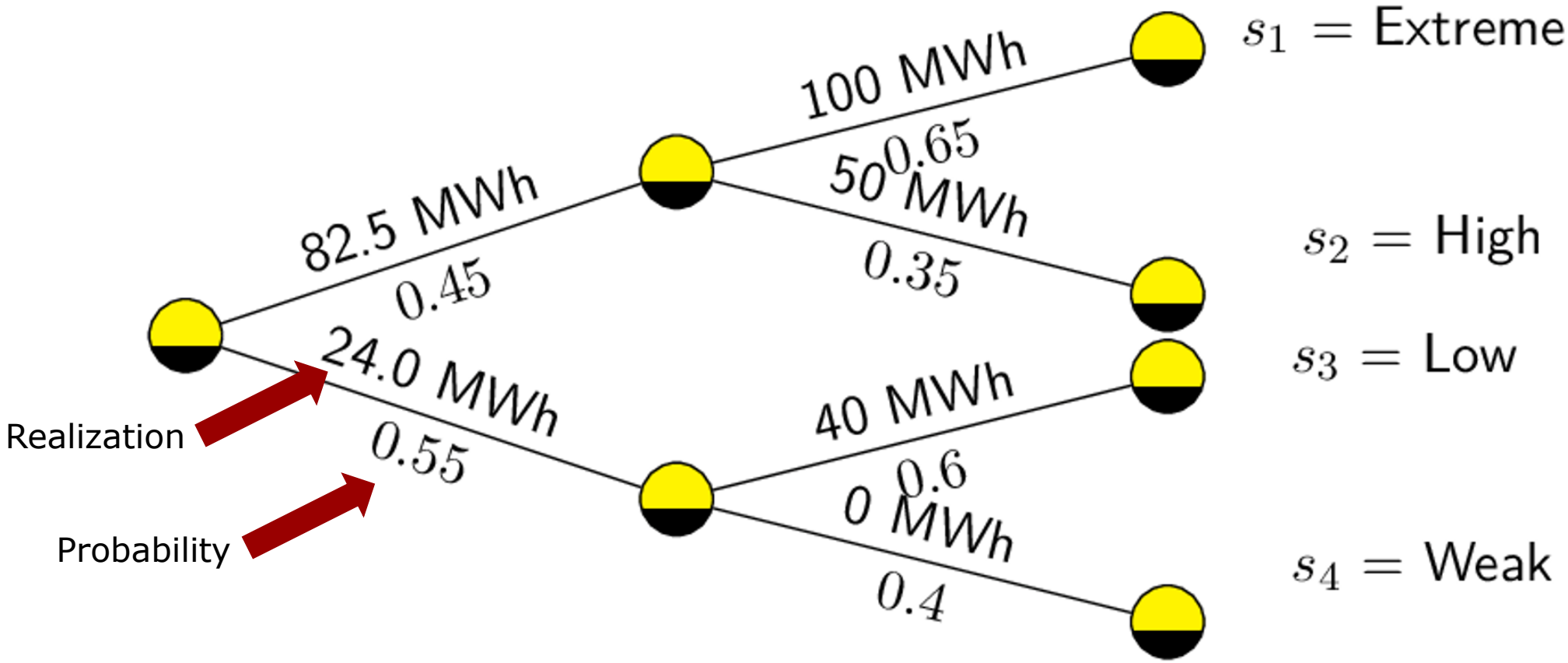
Scenario Trees

Example: Wind output across 3 stages



Scenario Trees

Example: Wind output across 3 stages

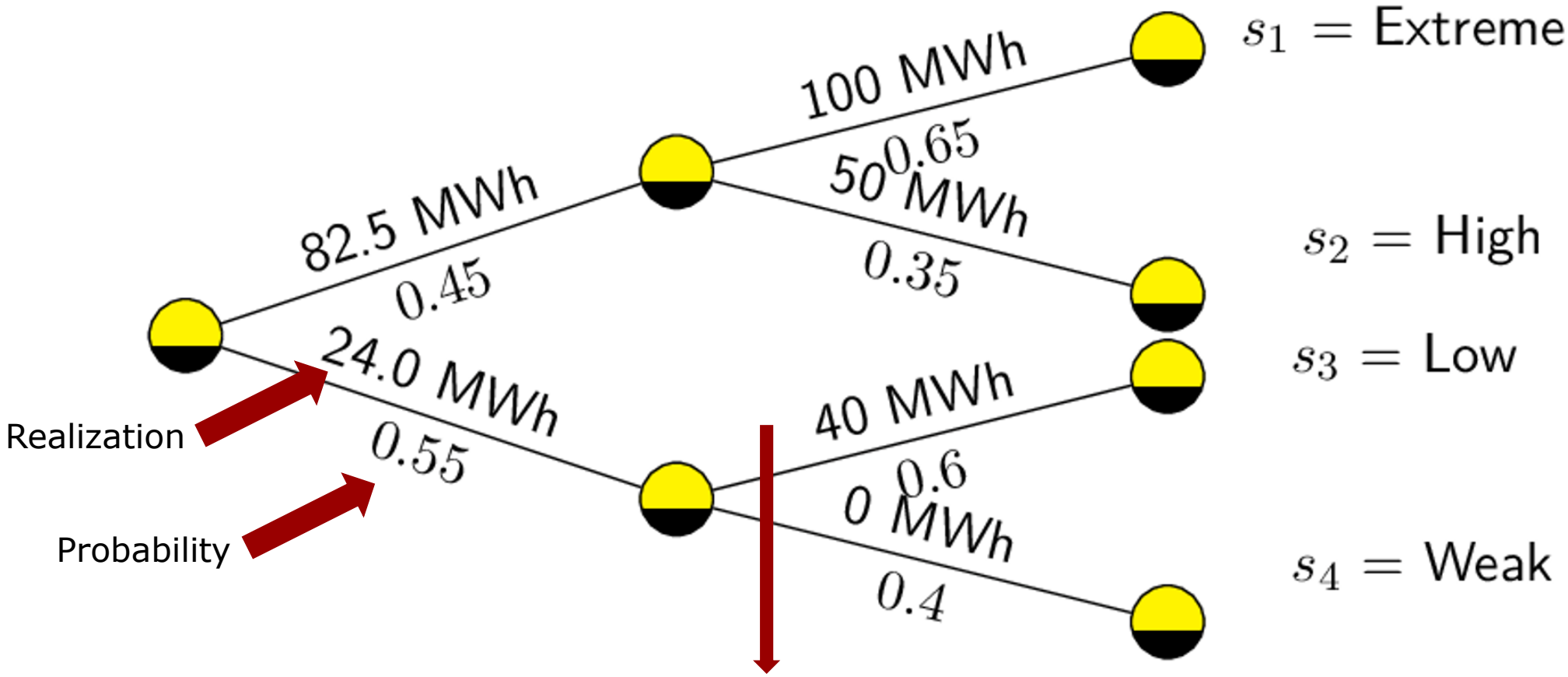


branches do not represent decisions, nodes are decisions, branches are realizations, and then we have the probability

branches geting out have too sum to 1

Scenario Trees

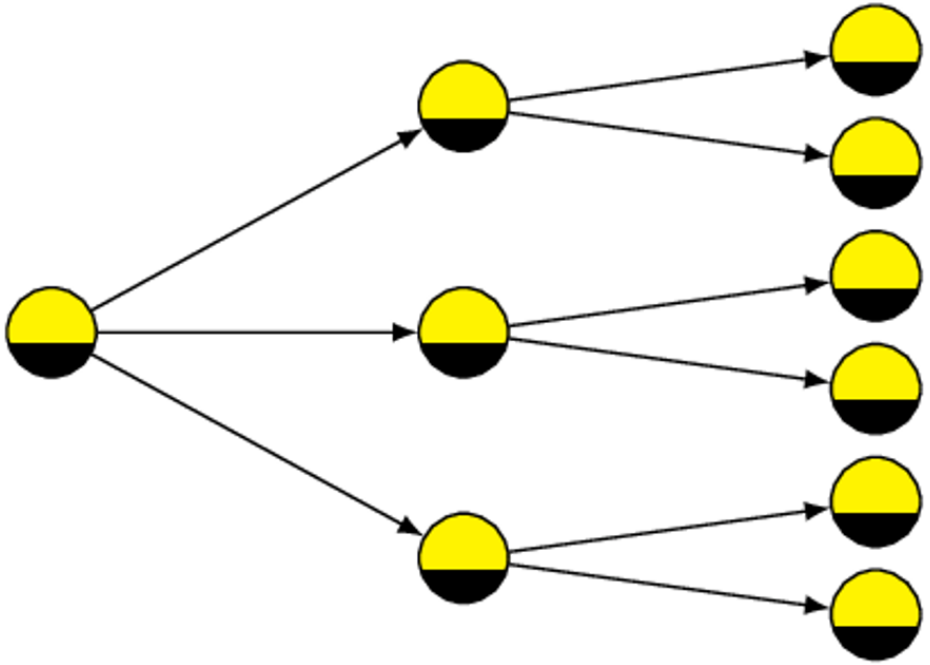
Example: Wind output across 3 stages



At each node, the probabilities of all child nodes must sum to 1.

Scenario Trees

Example: Wind output across 3 stages



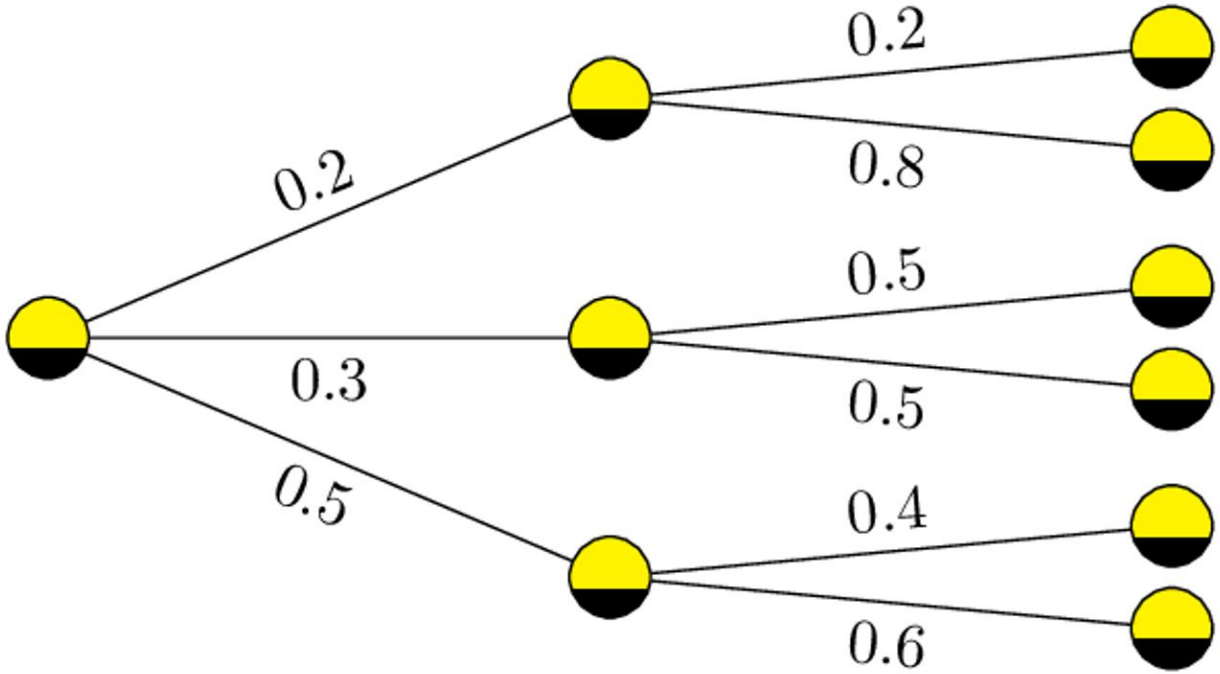
a scenario is a pathe acros this 3, not a point

we had 6 scenarios

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

Scenario Trees

Example: Wind output across 3 stages



What is the probability of Scenario 1 (uppermost path)?

0.2*0.2=0.004

Join at:
vevox.app

ID:
188-460-106

Scenario Creation in Multi-Stage SP

before we just sampled, and now it is a path

1. Begin at the current stage t
2. Generate Samples for $t+1$
3. Reduce them using clustering
4. Allocate Centroid Probabilities — as before
5. For $t+2$, branch out from each of the reduced samples (centroids) of $t+1$ — we use only the centroids of stage 2, and we branch out from the centroid, we sample them many times asdn we reudce centroidds
6. Reduce the $t+2$ samples using clustering
7. For $t+3$, branch out from each of the reduced samples (centroids) of $t+2$
8. …

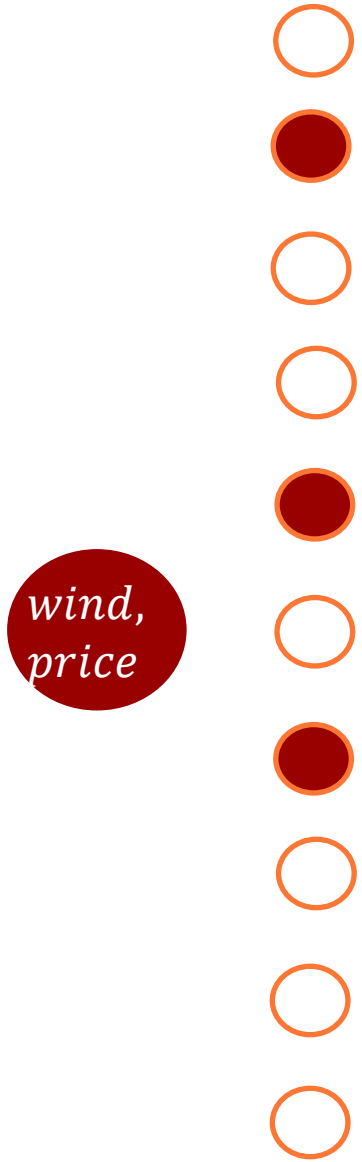
Repeated Branch-out & Cluster

Branch-out from current stage t

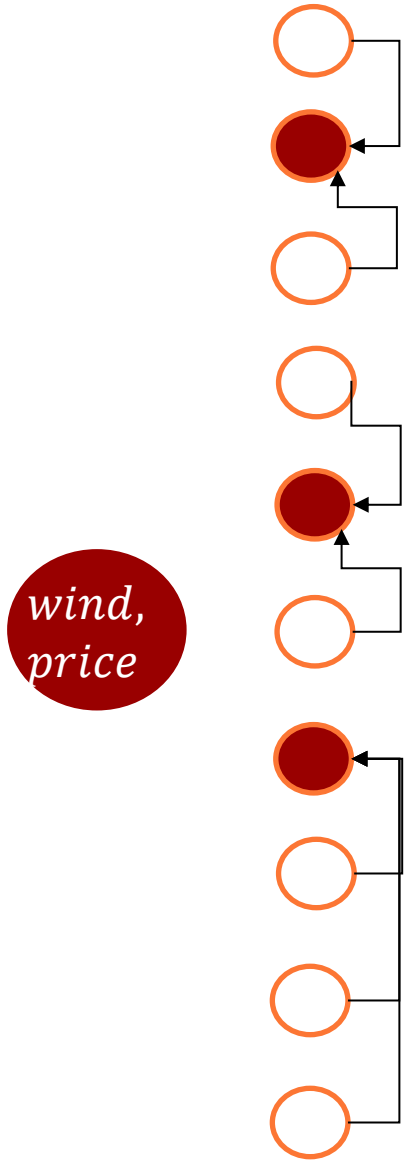
we sample many scenario 1000

Cluster

we reduce them to the number we want

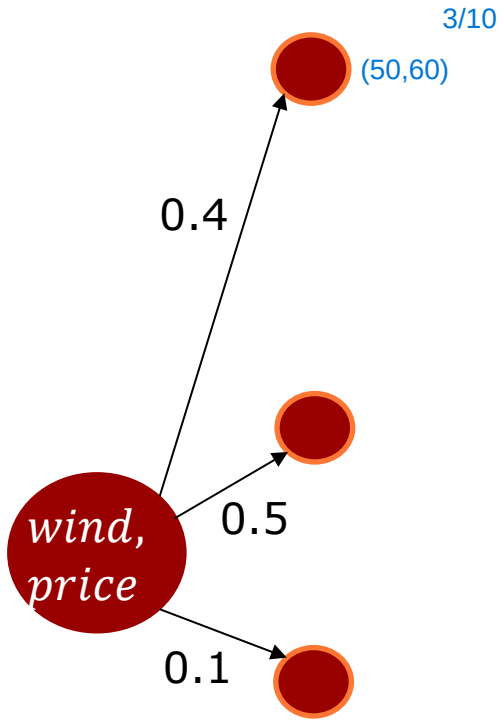


Allocate Probability to each Centroid

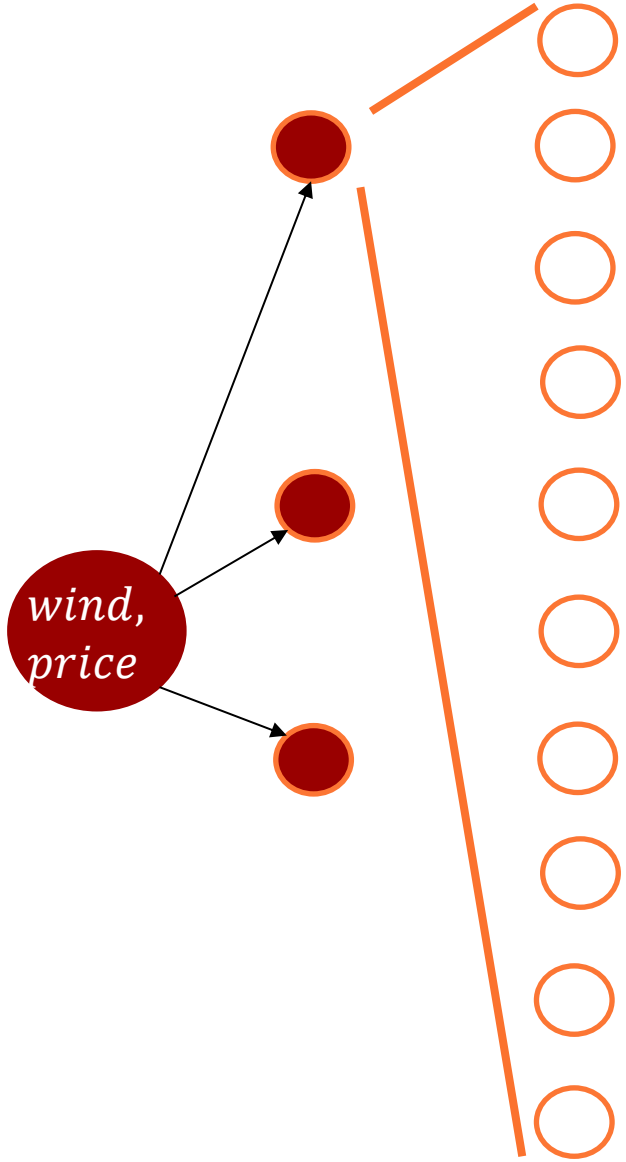


we assigne the probability too eache centroid

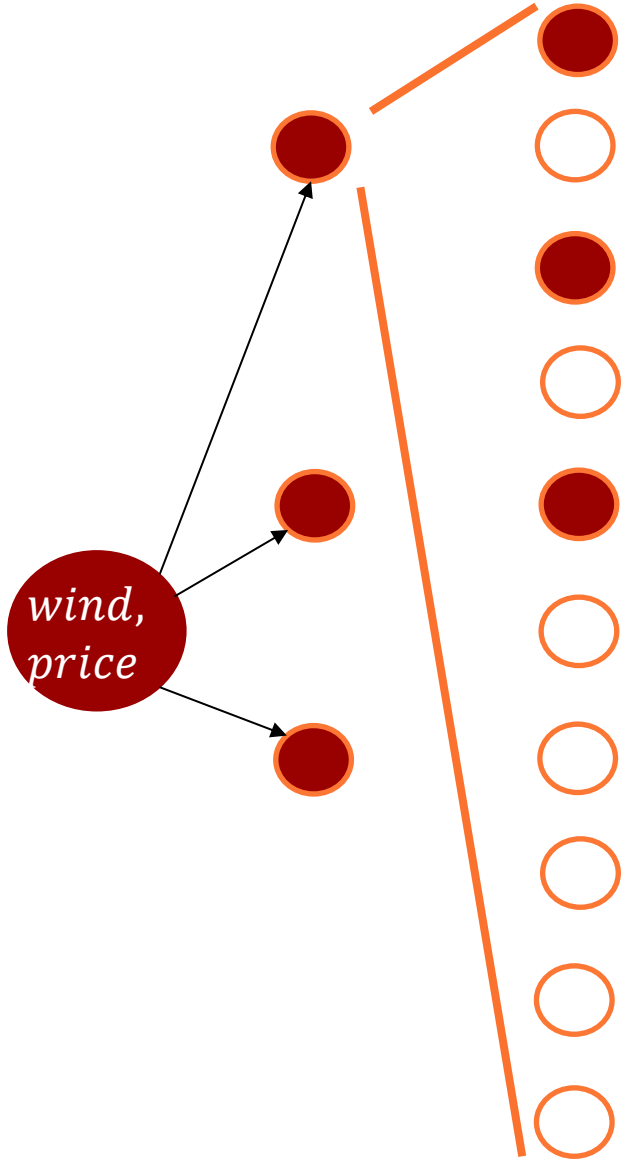
Ready to Move on…



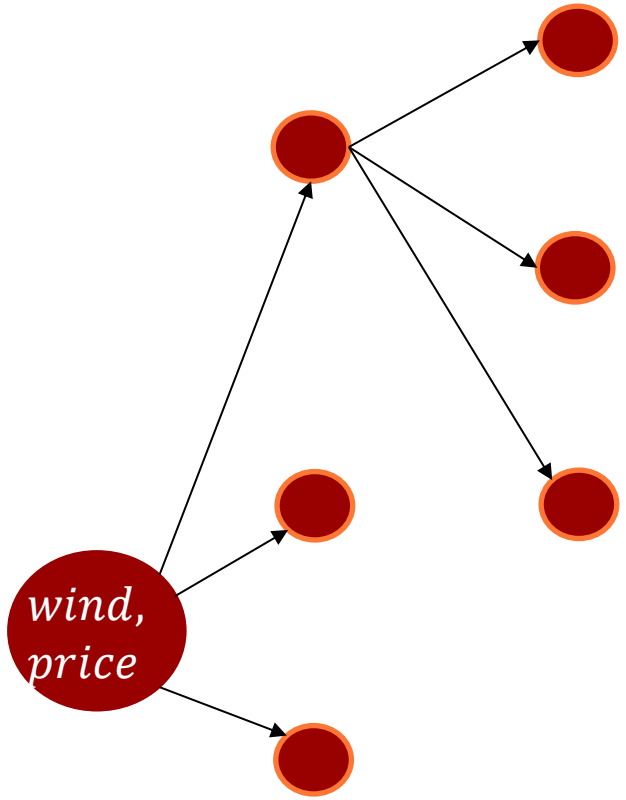
For each of the reduced $t + 1$ nodes: Branch-out



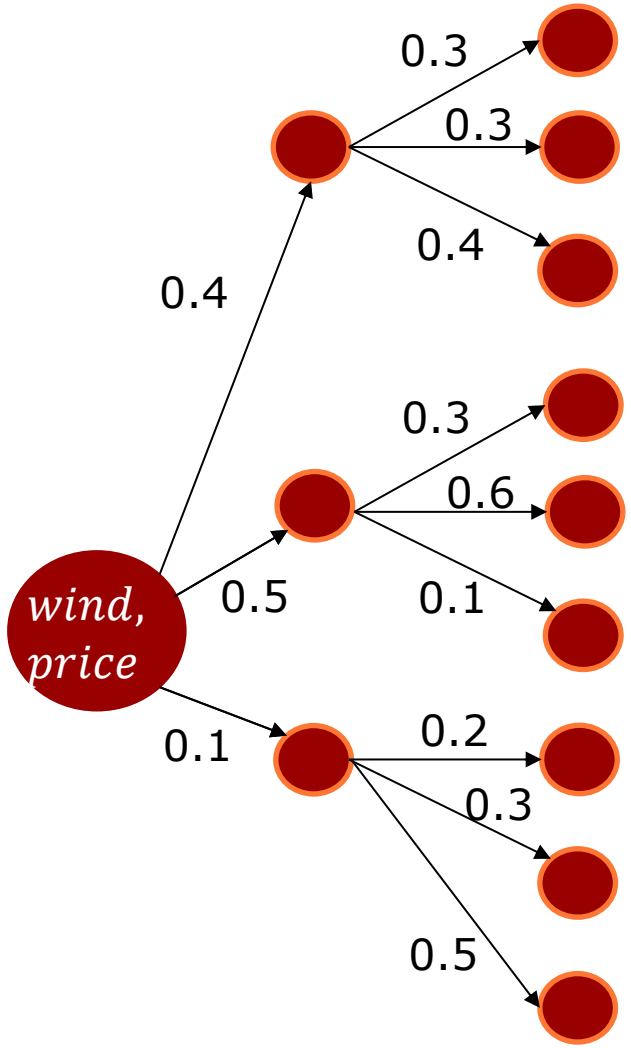
Cluster



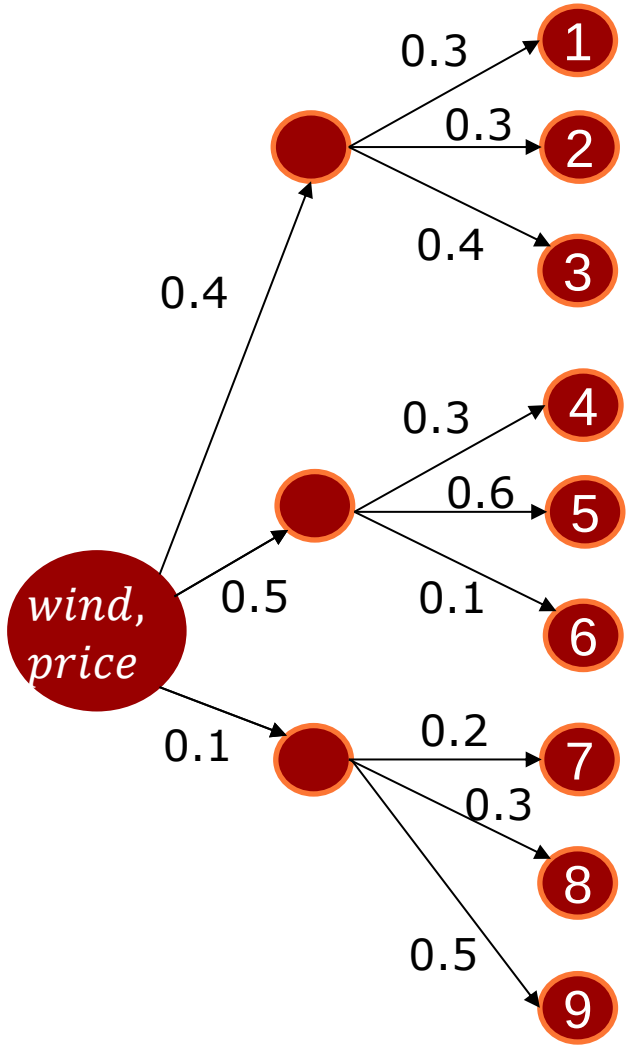
Repeat…



Repeat…



Repeat…



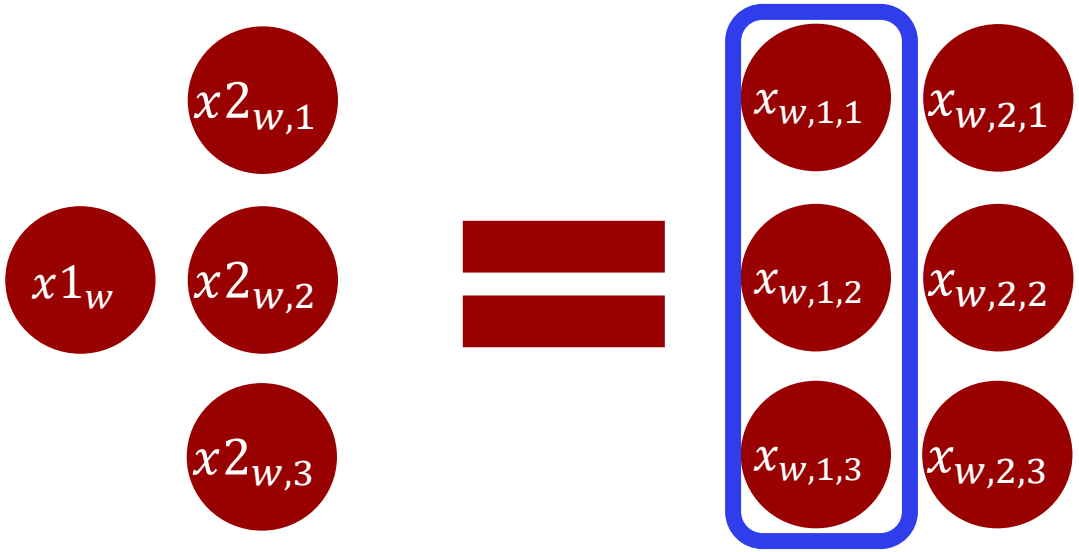
Each path (ending up to a leaf node) is one scenario.

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

Non-Anticipativity

we did multiple copies, they dont depend on the stage of the scenario

Two-Stage



Alternatively: same type of variable with more indices: $x_{w,t,s}$

And impose the constraints:

$x_{w,1,s} = x_{w,1,s'}, \quad \forall w \in W, (s, s') \in S$

What do these constraints encode?

the decsions are all the same for all scenarios

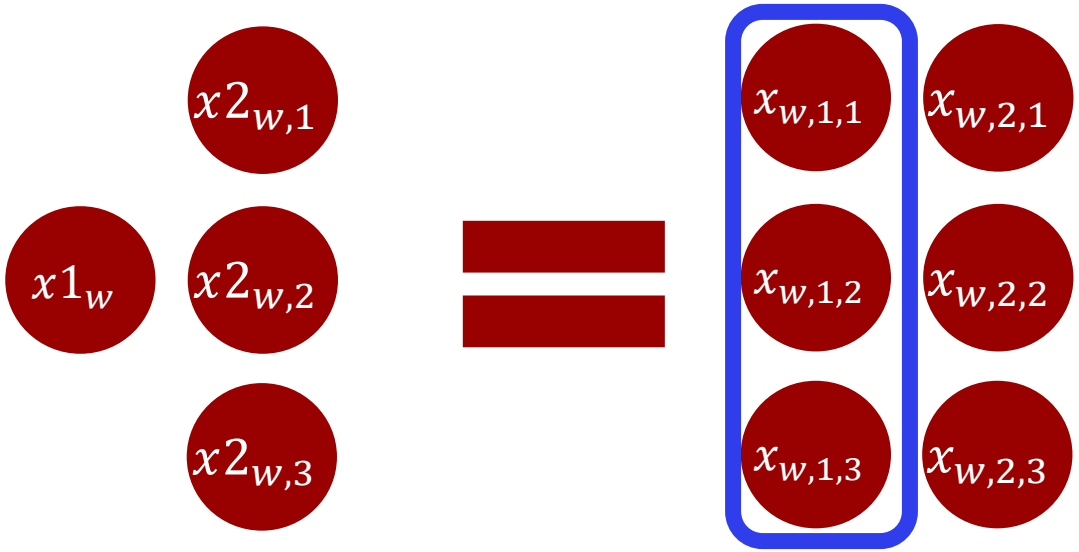
scnario do not depend of

Join at:
vevox.app

ID:
188-460-106

Non-Anticipativity

Two-Stage

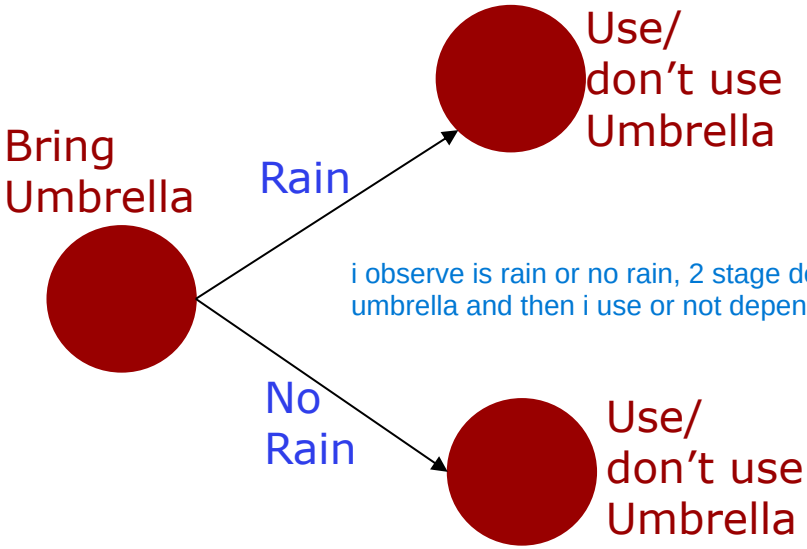


Alternatively: same type of variable with more indices: $x_{w,t,s}$

And impose the constraints:

$x_{w,1,s} = x_{w,1,s'}, \quad \forall w \in W, (s, s') \in S$

What do these constraints encode?

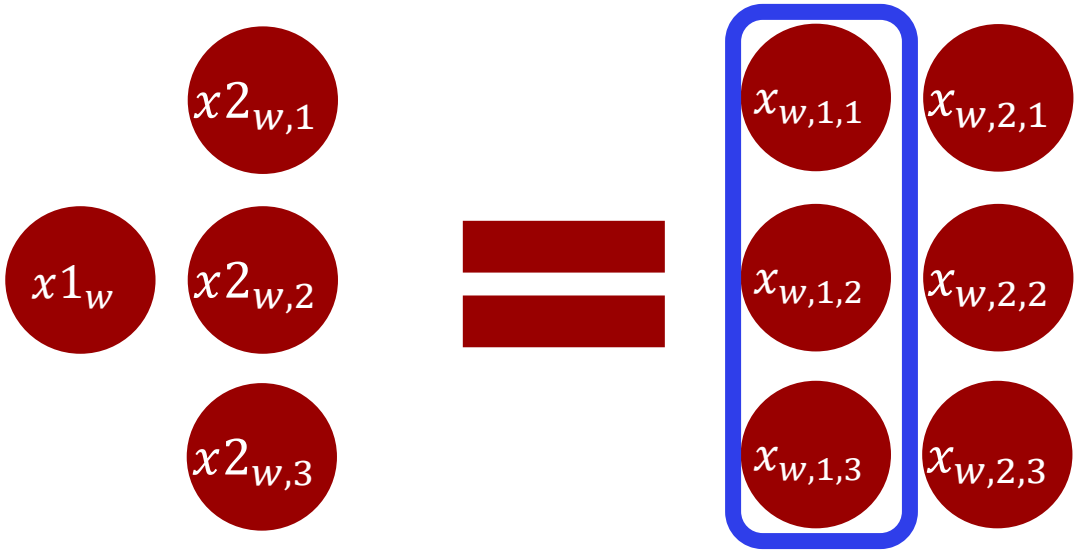


i observe is rain or no rain, 2 stage decsions, i brnig umbrella and then i use or not depending of weather

when i leaving my hojme, i cannot say i am bringing an umbrela if it reains and no if it doesnt rain, i cannot condition for a scenarios that o dont know yet

Non-Anticipativity

Two-Stage

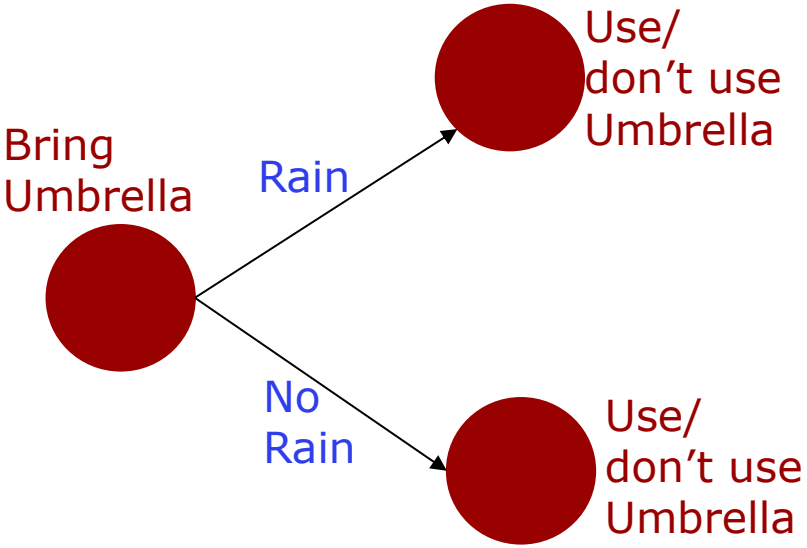


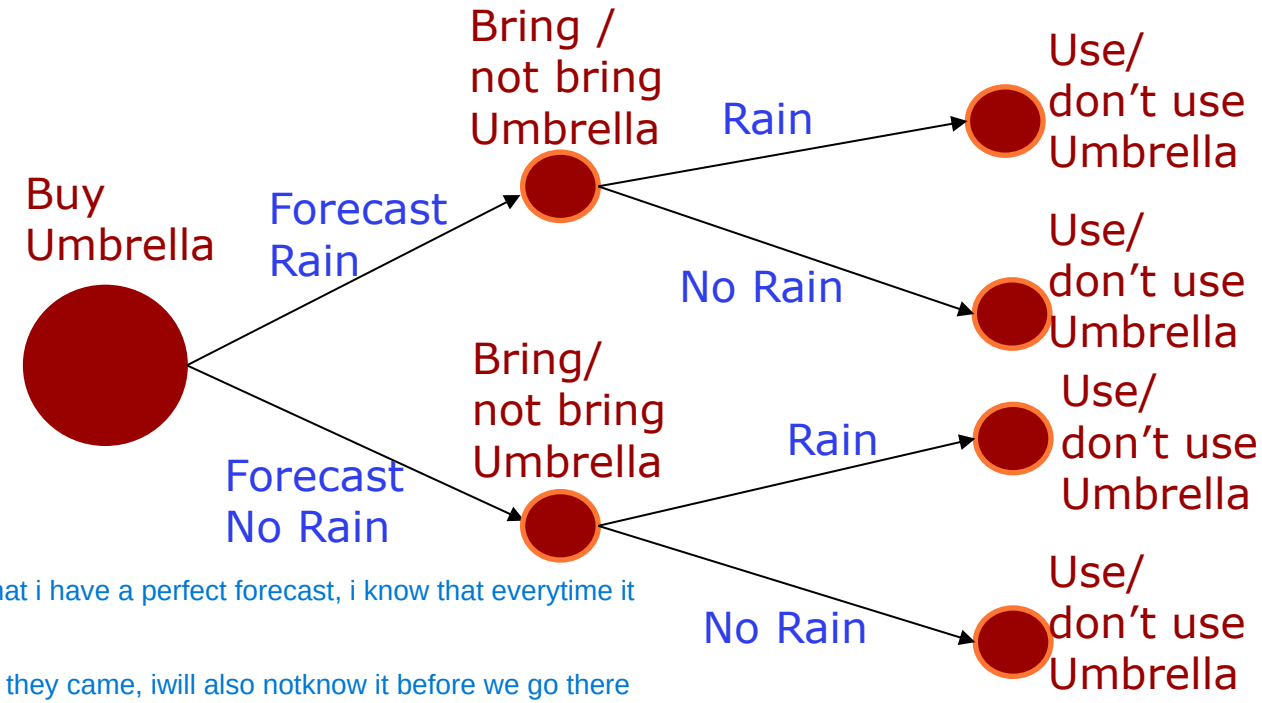
Alternatively: same type of variable with more indices: $x_{w,t,s}$

And impose the constraints:

$x_{w,1,s} = x_{w,1,s'}, \quad \forall w \in W, (s, s') \in S$

What do these constraints encode?



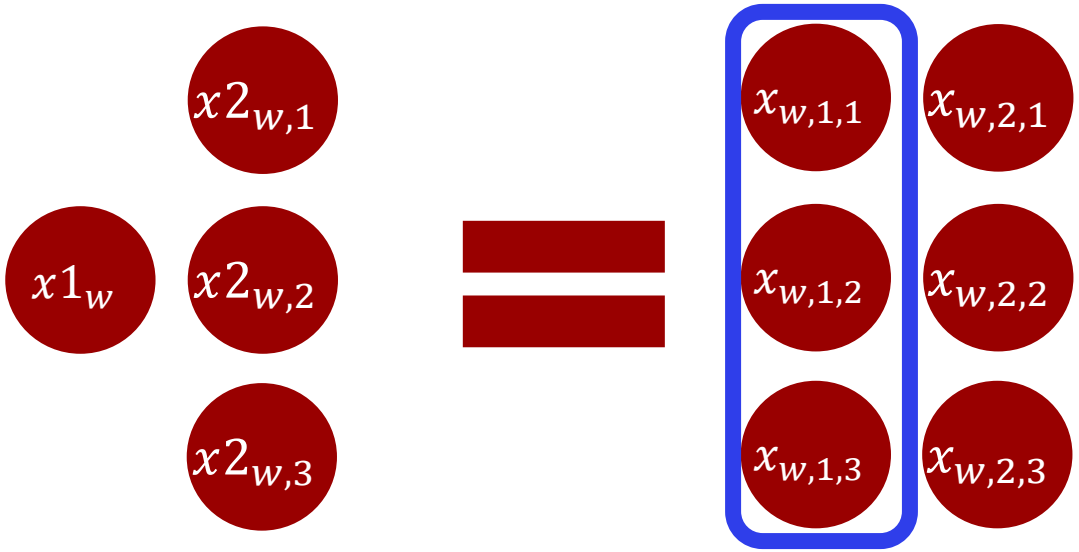


i have a forecast but i dont know yet
othwewise ill infor decsion, if program assumes that i have a perfect forecast, i know that everytime it rains so i will always have the pumbrella

we have to includ when the future decision, when they came, iwill also notknow it before we go there

Non-Anticipativity

Two-Stage

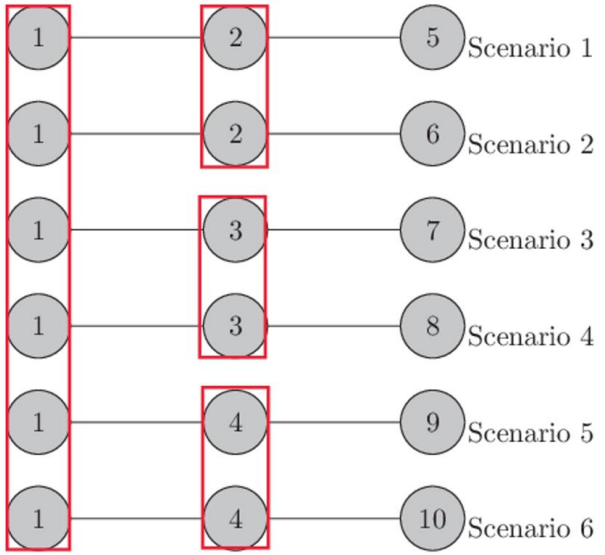


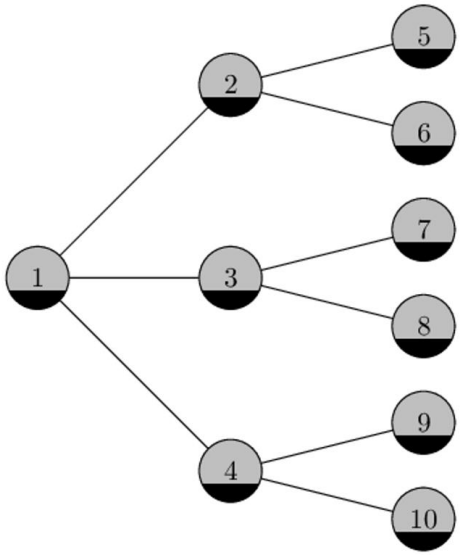
Alternatively: same type of variable with more indices: $x_{w,t,s}$

And impose the constraints:

$x_{w,1,s} = x_{w,1,s'}, \quad \forall w \in W, (s, s') \in S$

Three-Stage





$x_{w,t,s} = x_{w,t,s'}, \forall w \in W, t \in T, s \in S, s' \in S_s^t$ that shares scnario witj S

now i cannot predict stage 2 weather, but when i am in 2, i cannot also predicte after

Non-Anticipativity

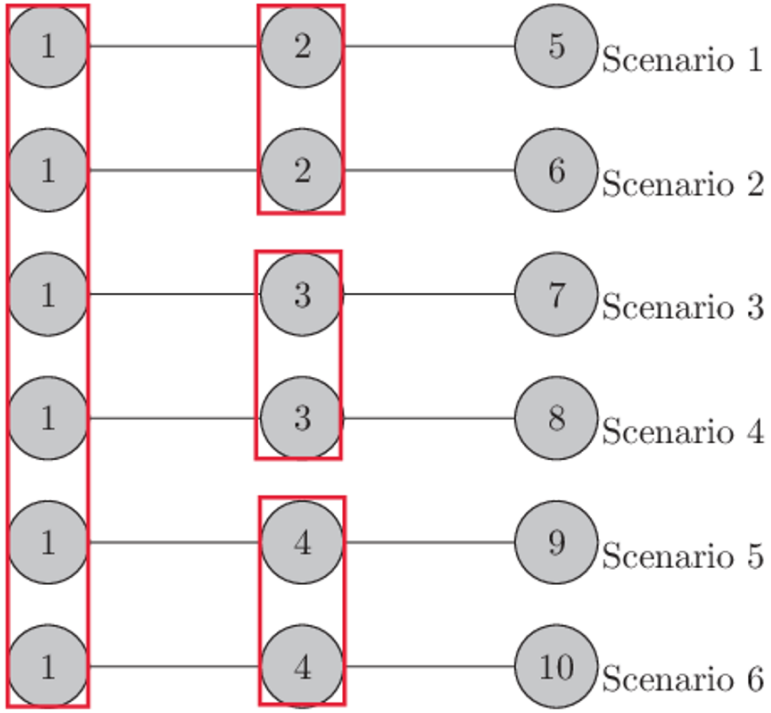
$$x_{w,t,s} = x_{w,t,s'}\ , \forall w \in W, t \in T, s \in S, s' \in S_s^t$$

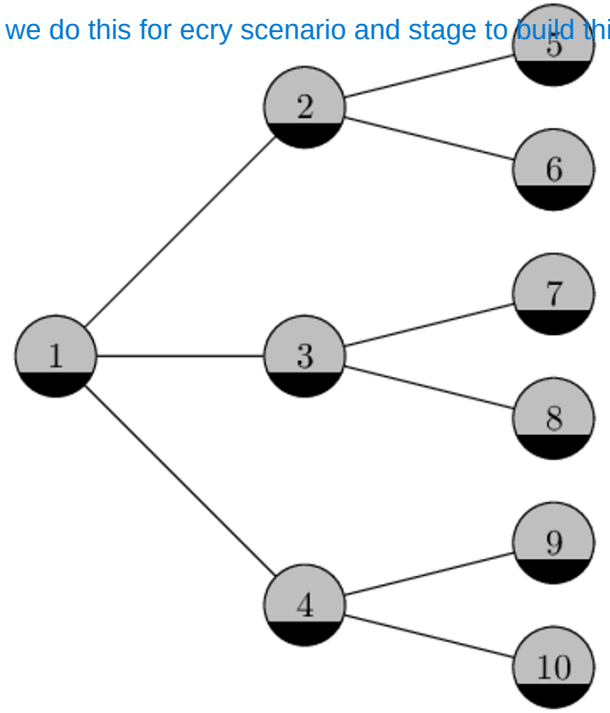
We define all variables based on scenarios and ensure non-anticipativity explicitly by constraints (marked in red).

which scenario shares history with scenrio in stage one? all scenarios

which scenarios share with scnario 1 up until stage 2, it is only scnerio 2 which has the same path as scenrio 1 (ball 5)

we do this for ecry scenario and stage to build this sets





$$S_1^1 = \{2,3,4,5,6\}$$
$$S_2^1 = \{1,3,4,5,6\}$$

$$S_1^2 = \{2\}$$
$$S_2^2 = \{1\}$$

$$S_3^2 = \{4\}$$
$$S_4^2 = \{3\}$$

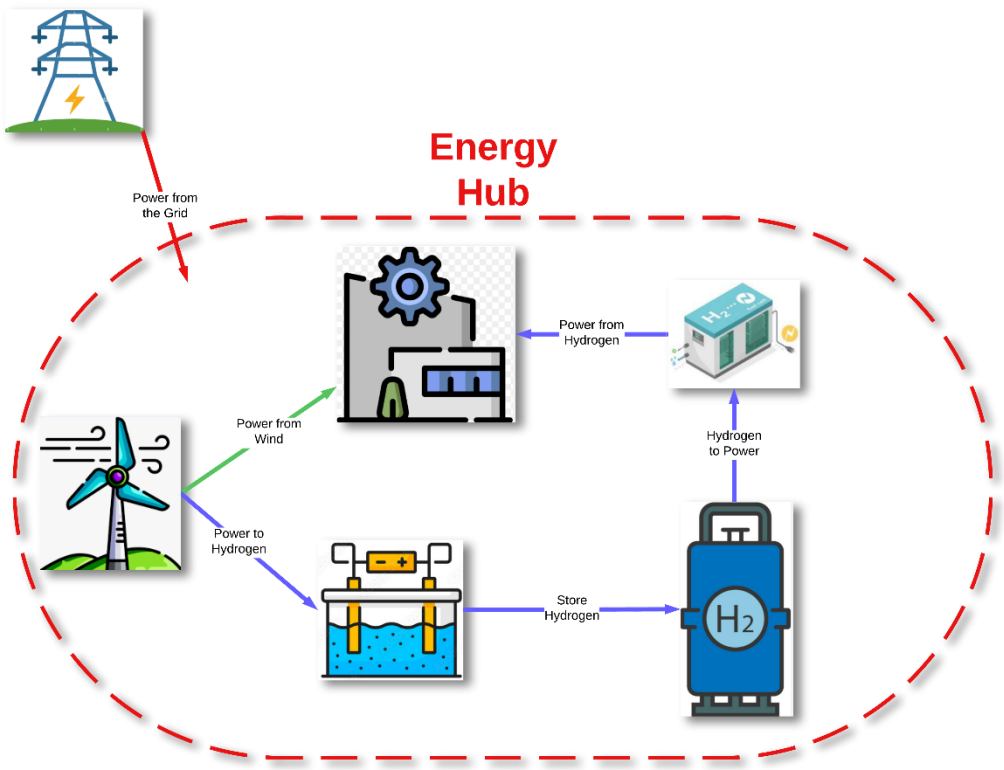
$$S_5^2 = \{6\}$$
$$S_6^2 = \{5\}$$

New sets:

S_s^t = Scenarios that need the same value as scenario s on stage t

Assignment A, Task 1

in near future i cannot either anticipate what is going to happend in the next future



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)

Stochastic Lookahead Policy

5-stage simulation horizon

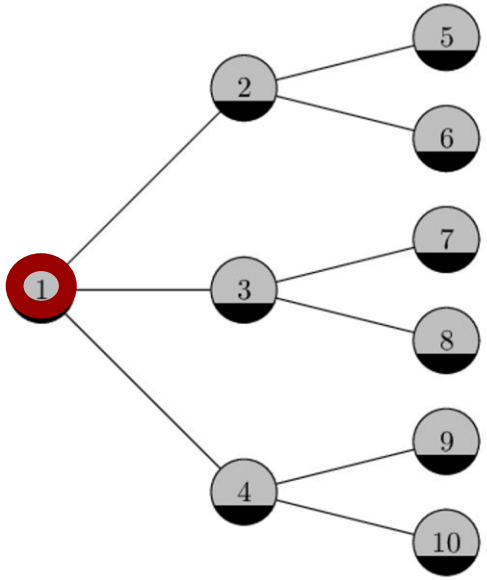
Policy: Lookahead horizon of 3 stages with 6 scenarios

Stages: 1 2 3 4 5

Stochastic Lookahead Policy

5-stage problem

Policy: Lookahead horizon of 3 stages with 6 scenarios

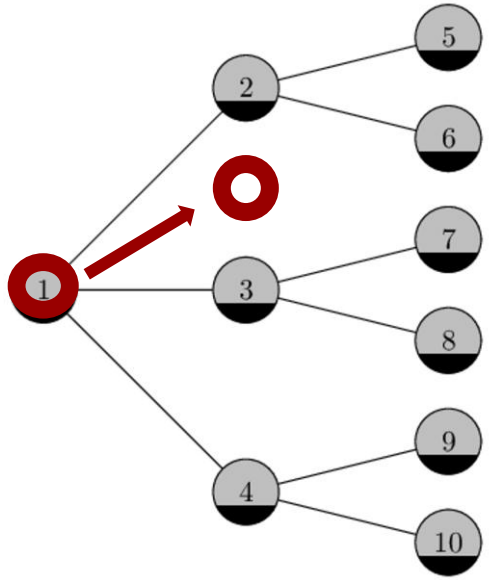


Stages: 1 2 3 4 5

Stochastic Lookahead Policy

5-stage problem

Policy: Lookahead horizon of 3 stages with 6 scenarios



the environment produces the next realization, what actually happens, we can observe wjat the wonf is actally, the uncertainity is revealed, it is almost impossible that is exactl ty the same as our scenarios

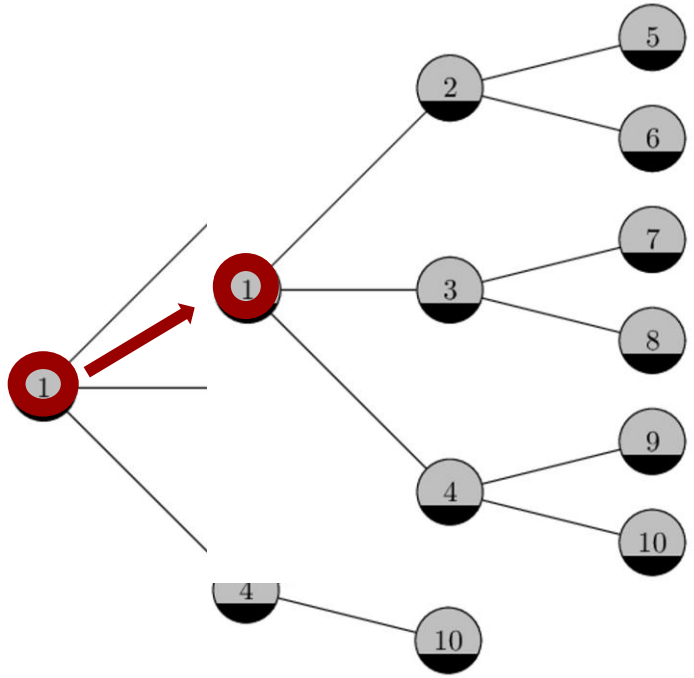
we trhow aways and we build nwew scenario 3, with the new uncertainity we have now

Stages: 1 2 3 4 5

Stochastic Lookahead Policy

5-stage problem

Policy: Lookahead horizon of 3 stages with 6 scenarios

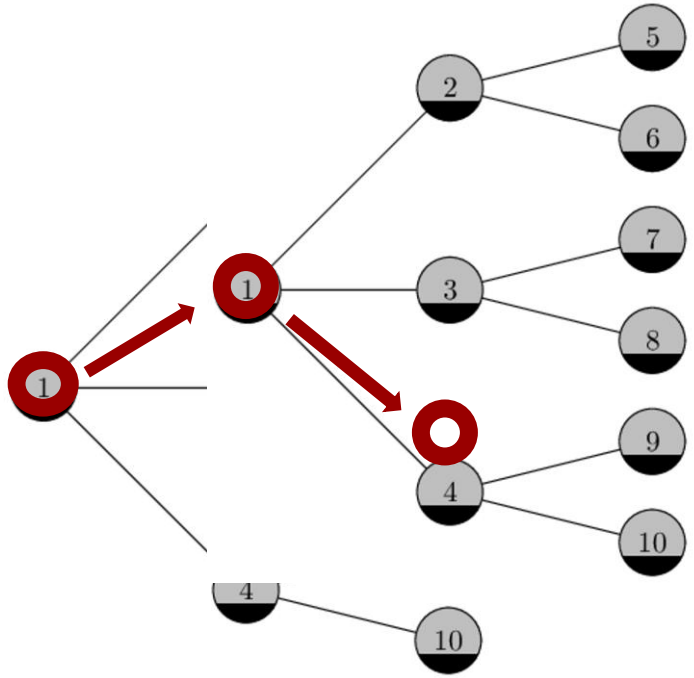


Stages: 1 2 3 4 5

Stochastic Lookahead Policy

5-stage problem

Policy: Lookahead horizon of 3 stages with 6 scenarios

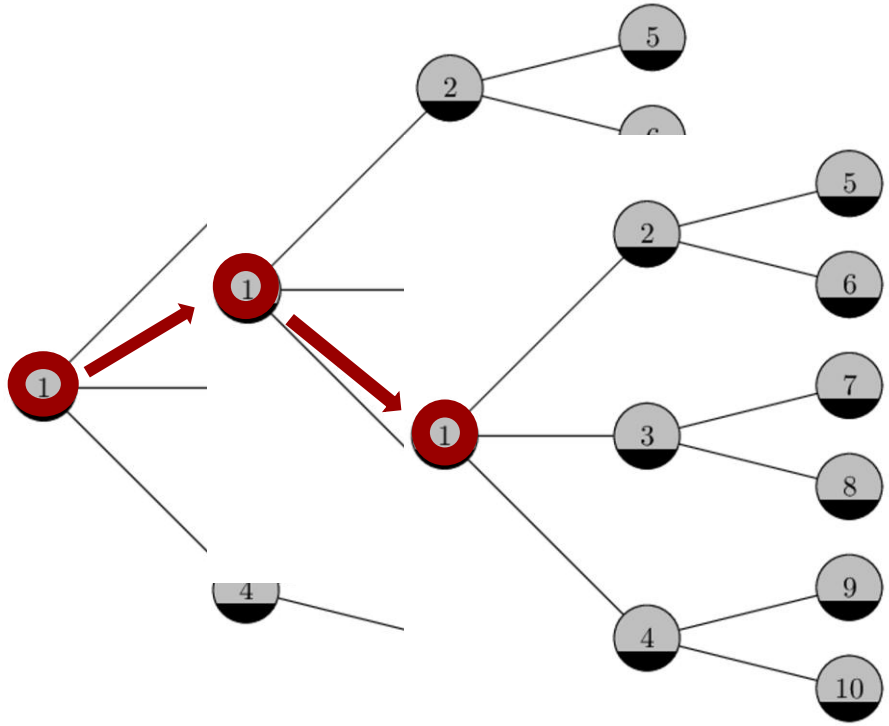


Stages: 1 2 3 4 5

Stochastic Lookahead Policy

5-stage problem

Policy: Lookahead horizon of 3 stages with 6 scenarios

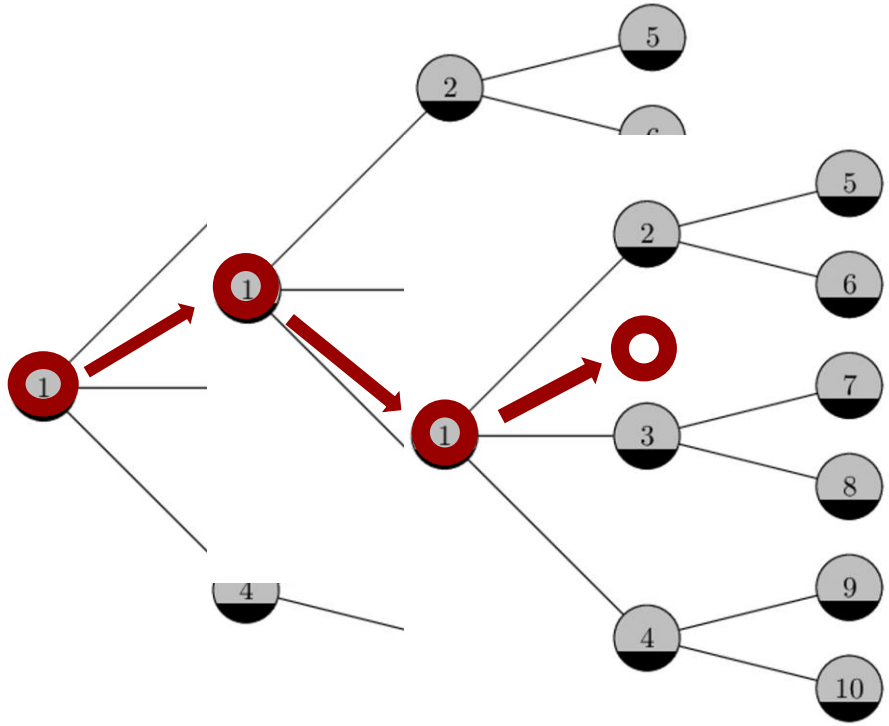


Stages: 1 2 3 4 5

Stochastic Lookahead Policy

5-stage problem

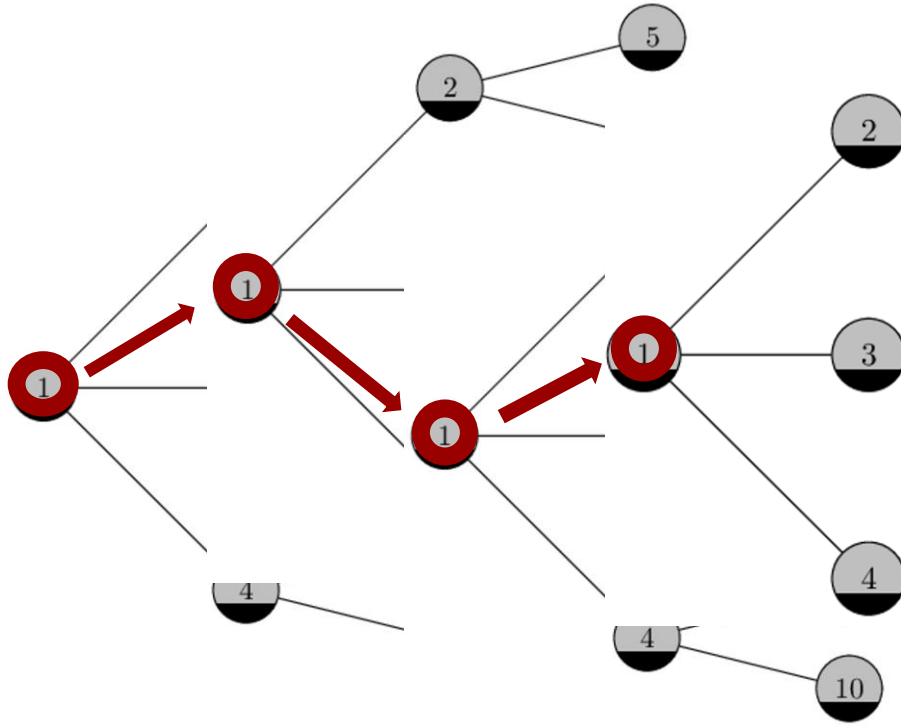
Policy: Lookahead horizon of 3 stages with 6 scenarios



Stochastic Lookahead Policy

5-stage problem

Policy: Lookahead horizon of 3 stages with 6 scenarios



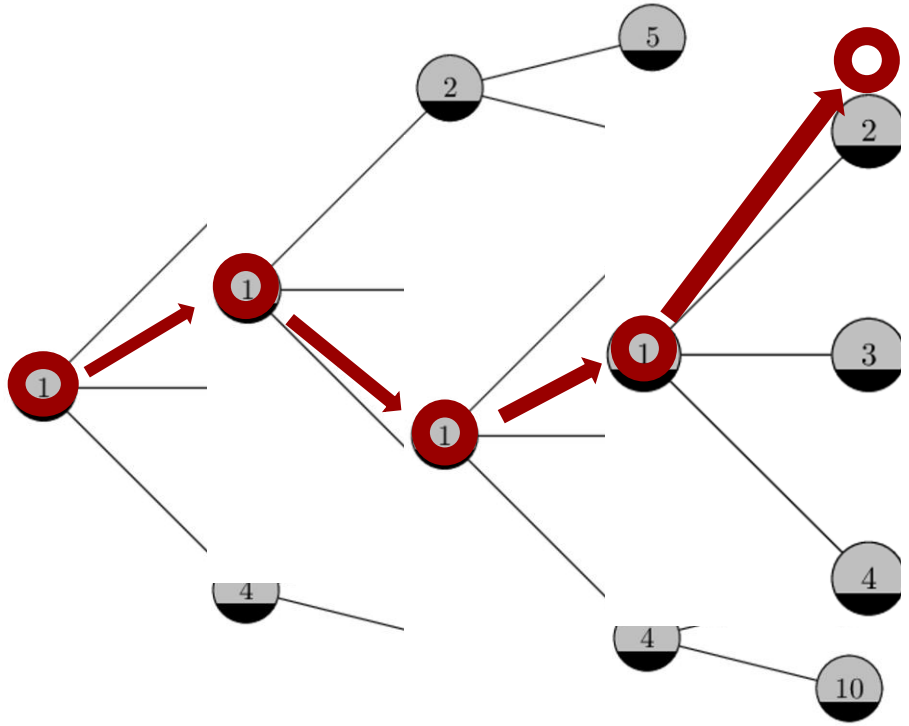
Stages: 1 2 3 4 5

we only have one stage ahead, we shrink we only look at the very last stage

Stochastic Lookahead Policy

5-stage problem

Policy: Lookahead horizon of 3 stages with 6 scenarios

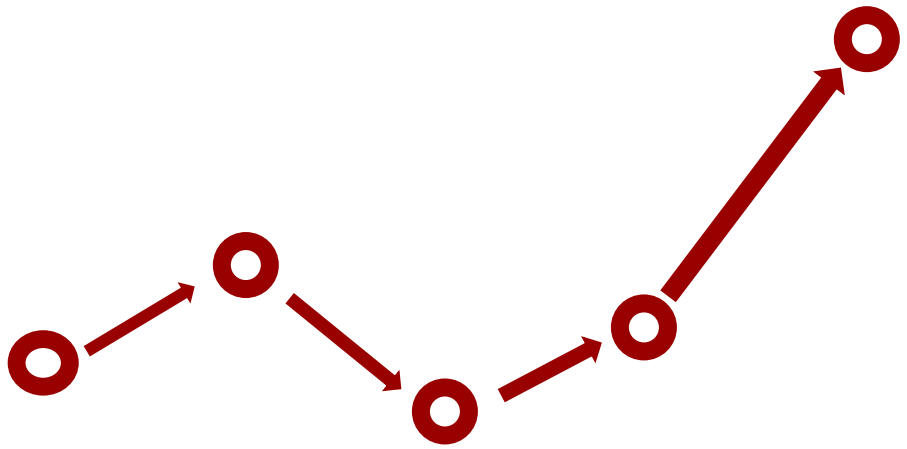


Stages: 1 2 3 4 5

Stochastic Lookahead Policy

5-stage problem

Policy: Lookahead horizon of 3 stages with 6 scenarios

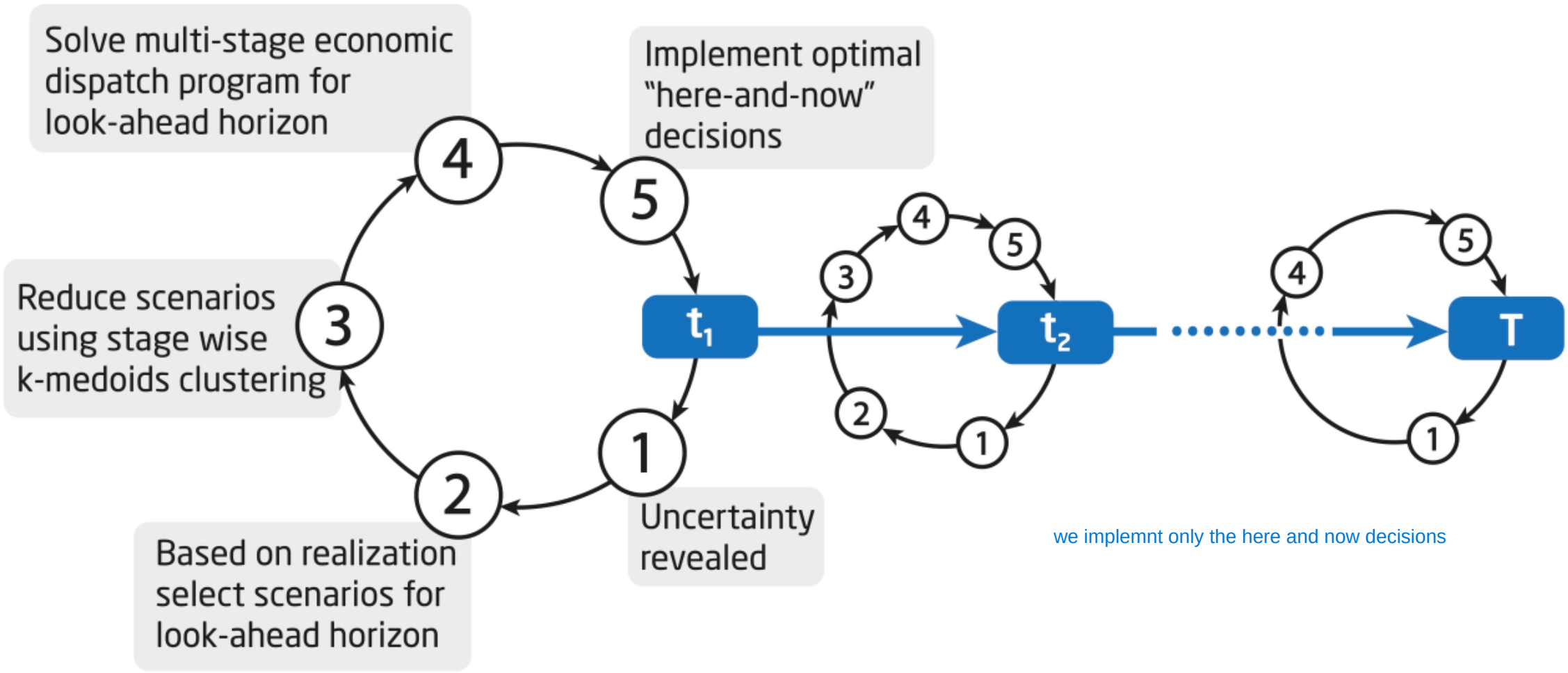


theer is anything to look aghead for

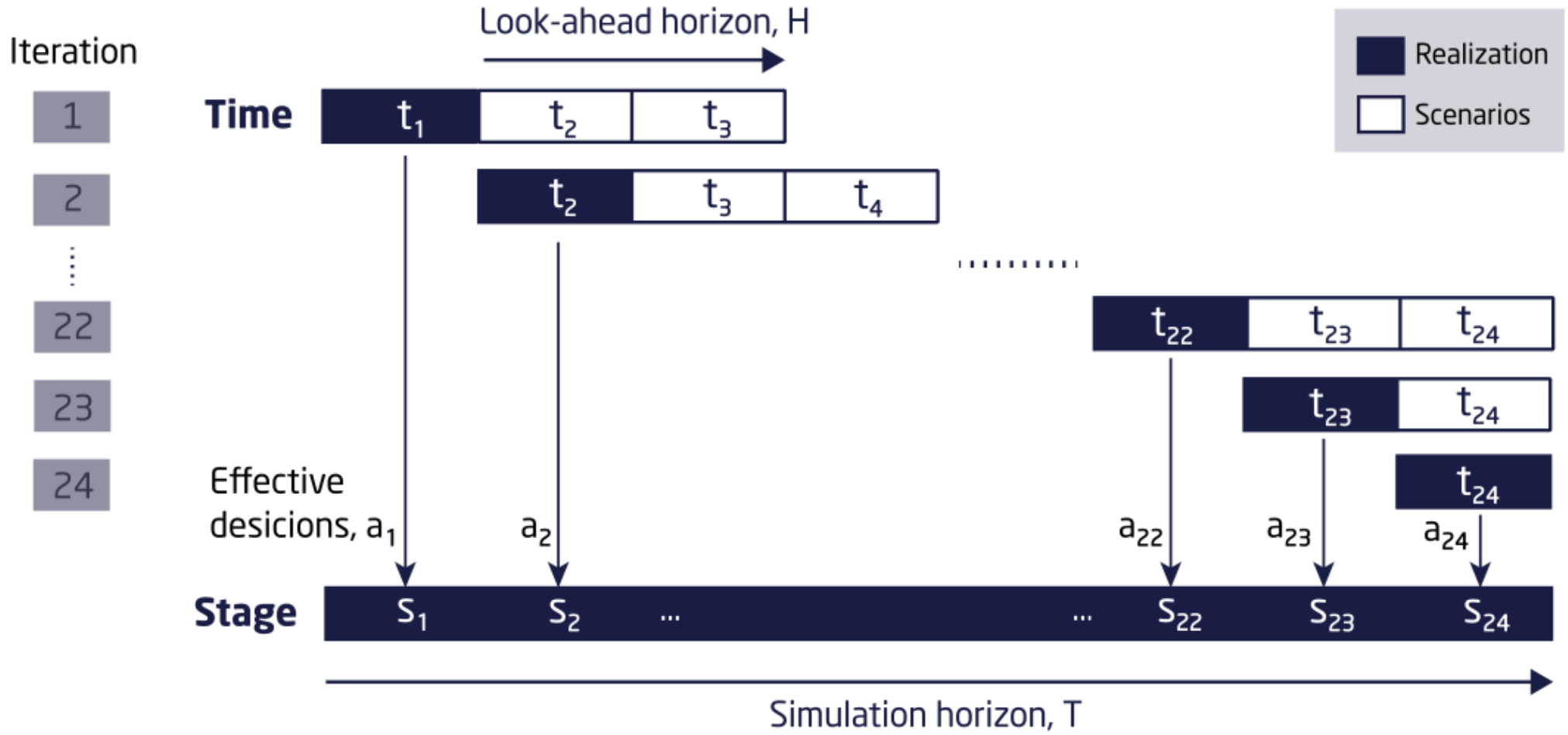
we only count the cost on this that actually happened

Stages: 1 2 3 4 5

Evaluating a Multi-Stage SP policy

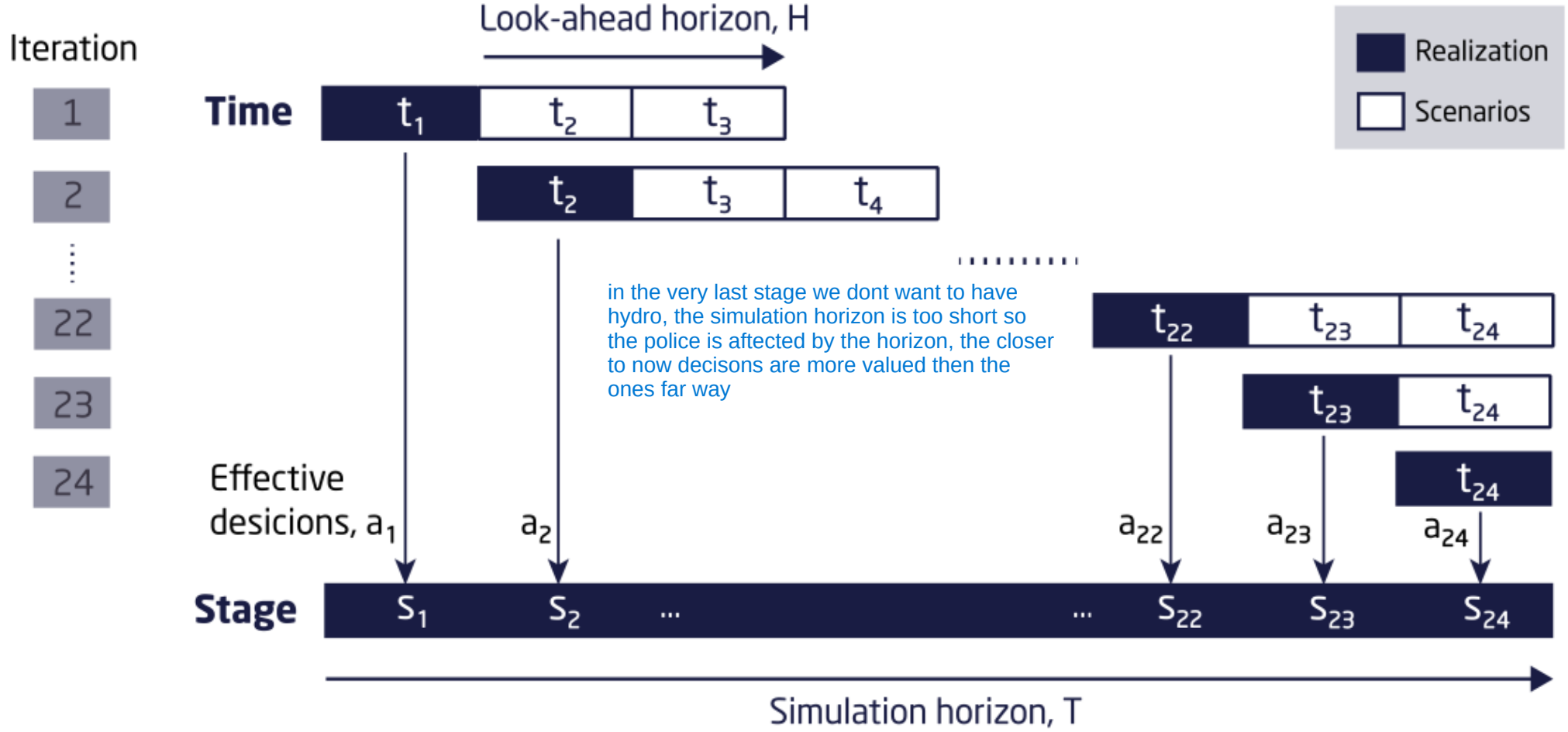


Evaluating a Multi-Stage SP policy



Evaluating a Multi-Stage SP policy

how can we choose this itme horizon, we need to validate



Questions and Survey

Join at:
vevox.app

ID:
188-460-106

Game Quiz

Join at:
vevox.app

ID:
188-974-928