A tutorial of Markov Decision Process starting from the perspective of Stochastic Programming

Yixin Ye
Department of Chemical Engineering, Carnegie Mellon University

Feb 13, 2020

Markov?

A. A. Марков. "Распространение закона больших чисел на величины, зависящие друг от друга". "Известия Физико-математического общества при Казанском университете", 2-я серия, том 15, ст. 135–156, 1906

A. A. Markov. "Spreading the law of large numbers to quantities that depend on each other." "Izvestiya of the Physico-Mathematical Society at the Kazan University", 2-nd series, volume 15, art. 135-156, 1906

Andrey Andreyevich Markov

Why – Wide applications

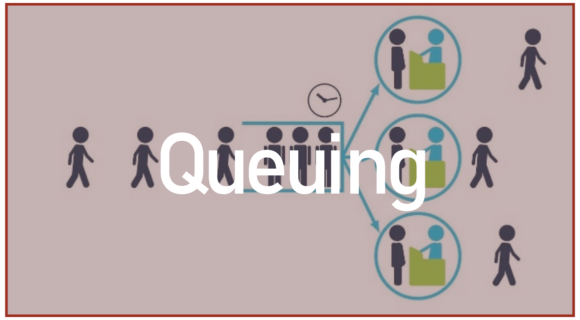
- White, Douglas J. "A survey of applications of Markov decision processes." *Journal of the operational research society* 44.11 (1993): 1073-1096.

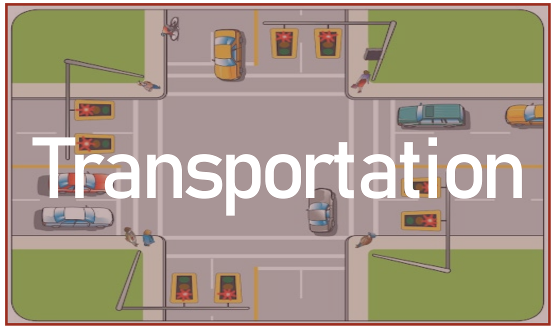
TABLE 1. *Application areas*

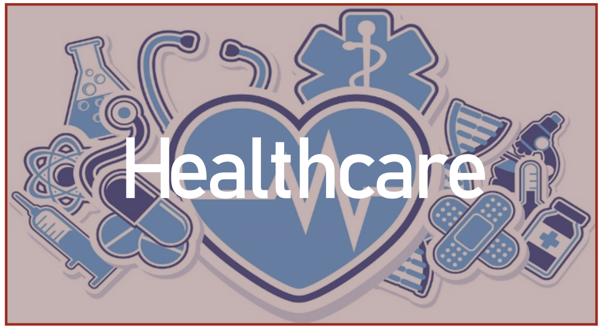
1	Population harvesting	(5)
2	Agriculture	(5)
3	Water resources	(15)
4	Inspection, maintenance and repair	(18)
5	Purchasing, inventory and production	(14)
6	Finance and investment	(9)
7	Queues	(6)
8	Sales promotion	(4)
9	Search	(3)
10	Motor insurance claims	(2)
11	Overbooking	(5)
12	Epidemics	(2)
13	Credit	(2)
14	Sports	(2)
15	Patient admissions	(1)
16	Location	(1)
17	Design of experiments	(1)
18	General	(5)

- Boucherie, Richard J., and Nico M. Van Dijk, eds. *Markov decision processes in practice*. Springer International Publishing, 2017.

Part I: General Theory
Part II: Healthcare
Part III: Transportation
Part IV: Production
Part V: Communications
Part VI: Financial Modeling







MDP x PSE

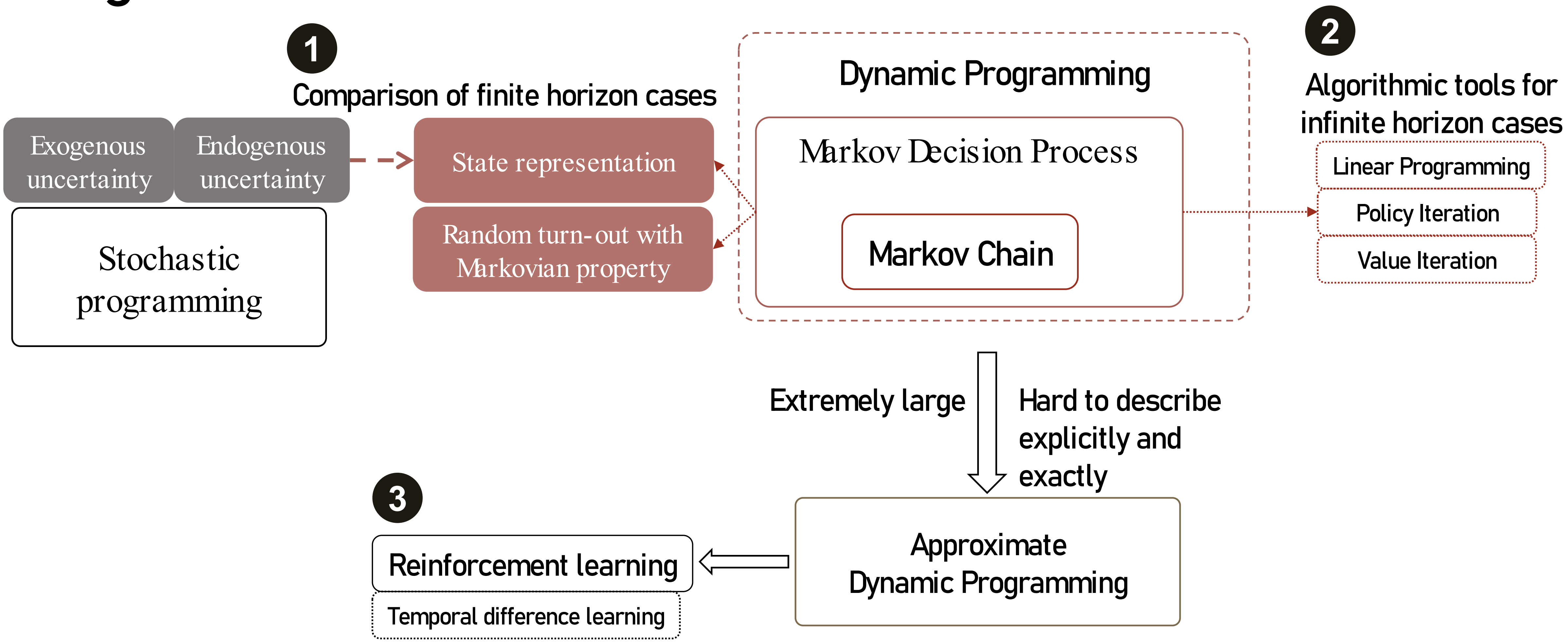
- Saucedo, Victor M., and M. Nazmul Karim. "On-line optimization of stochastic processes using Markov Decision Processes." *Computers & chemical engineering* 20 (1996): S701-S706.
- Tamir, Abraham. *Applications of Markov chains in chemical engineering*. Elsevier, 1998.
- Wongthatsanekorn, Wuthichai & Realff, Matthew J. & Ammons, Jane C., 2010. "Multi-time scale Markov decision process approach to strategic network growth of reverse supply chains," Omega, Elsevier, vol. 38(1-2), pages 20-32, February.
- Wong, Wee Chin, and Jay H. Lee. "Fault detection and diagnosis using hidden Markov disturbance models." Industrial & Engineering Chemistry Research 49.17 (2010): 7901-7908.
- Martagan, Tugce, and Ananth Krishnamurthy. "Control and Optimization of Bioprocesses Using Markov Decision Process." *IIE Annual Conference. Proceedings*. Institute of Industrial and Systems Engineers (IISE), 2012.
- Goel, Vikas, and Kevin C. Furman. "Markov decision process-based support tool for reservoir development planning." U.S. Patent No. 8,775,347. 8 Jul. 2014.
- Kim, Jong Woo, et al. "Optimal scheduling of the maintenance and improvement for water main system using Markov decision process." *IFAC-Papers OnLine* 48.8 (2015): 379-384.

How – Comparative demonstration

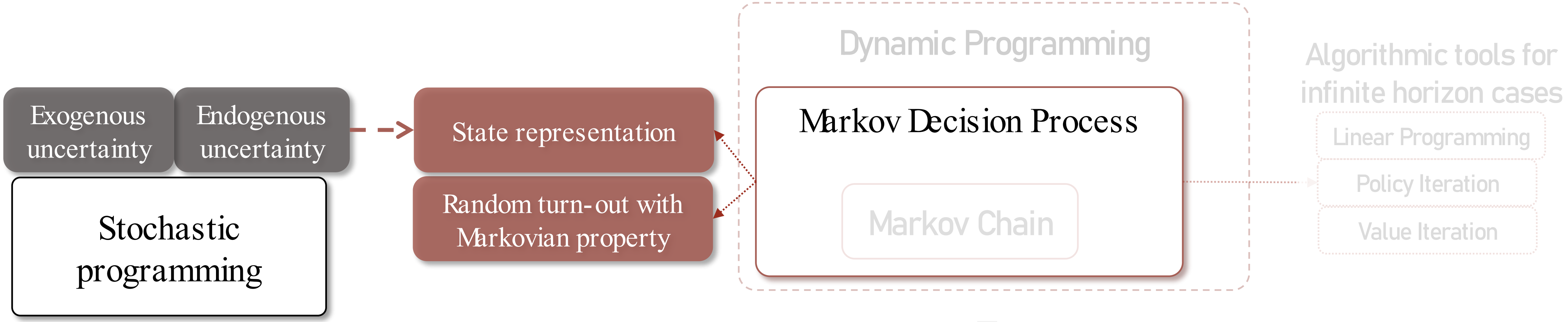
- Markov Decision Process is a less familiar tool to the PSE community for decision-making under uncertainty.
- Stochastic programming is a more familiar tool to the PSE community for decision-making under uncertainty.
- This talk will start from a comparative demonstration of these two, as a perspective to introduce Markov Decision Process.

- Dupačová, J., & Sladký, K. (2002). Comparison of multistage stochastic programs with recourse and stochastic dynamic programs with discrete time. *ZAMM-Journal of Applied Mathematics and Mechanics/Zeitschrift für Angewandte Mathematik und Mechanik: Applied Mathematics and Mechanics*, *82*(11-12), 753-765.
- Cheng, L., Subrahmanian, E., & Westerberg, A. W. (2004). A comparison of optimal control and stochastic programming from a formulation and computation perspective. *Computers & Chemical Engineering*, *29*(1), 149-164.
- Powell, W. B. (2019). A unified framework for stochastic optimization. European Journal of Operational Research, 275(3), 795-821.

Things to cover



Things to cover

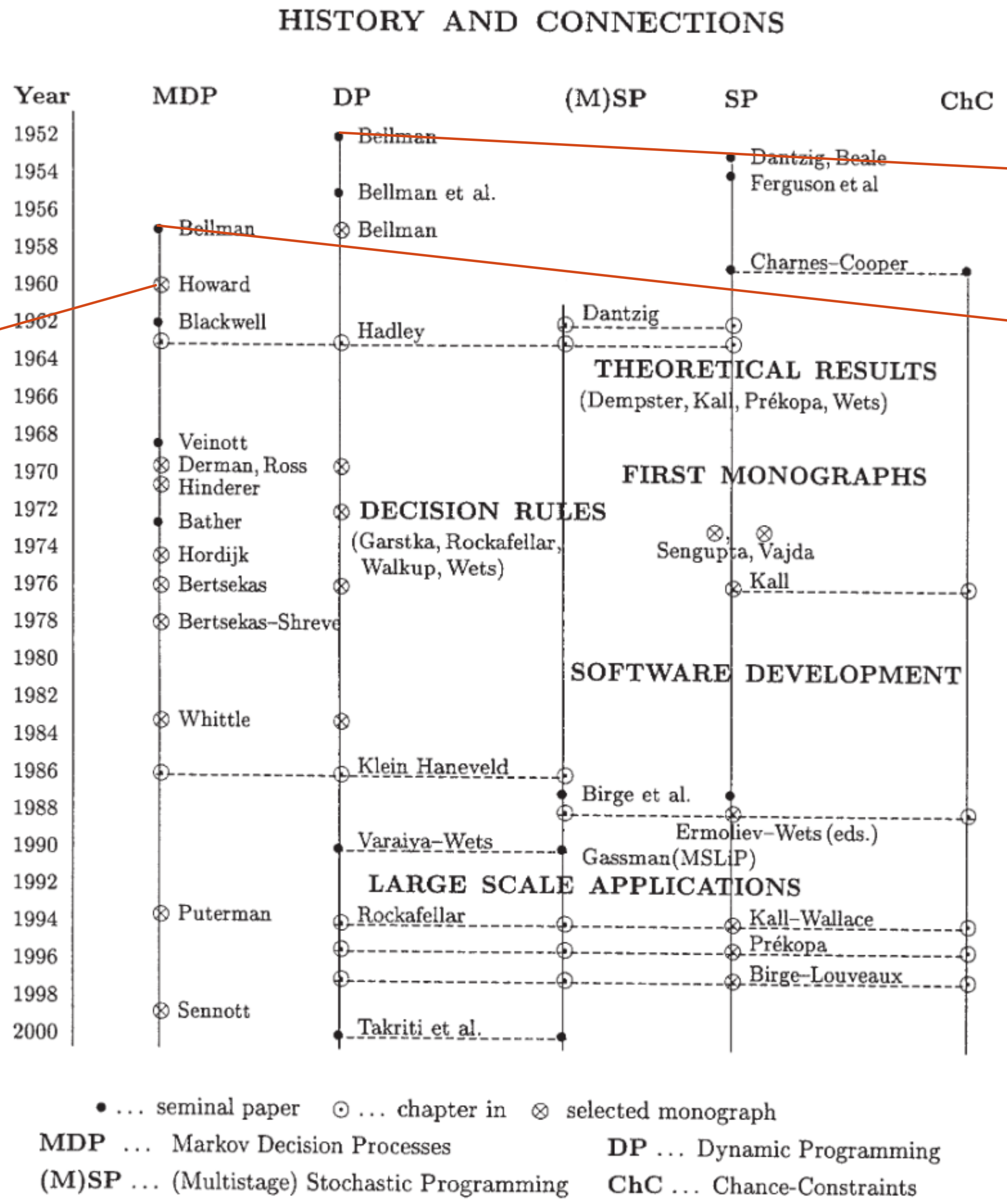


Multi-stage stochastic programming VS Finite-horizon Markov Decision Process

- Special properties, general formulations and applicable areas
- Intersection at an example problem

Book: Dynamic Programming and Markov Processes, 1960

Ronald A. Howard



Richard Bellman

On the theory of dynamic programming. *Proceedings of the National Academy of Sciences of the United States of America* 38.8 (1952): 716.

A Markovian Decision Process, Indiana Univ. Math. J. 6 No. 4 (1957), 679–684

Dupačová, J., & Sladký, K. (2002). Comparison of multistage stochastic programs with recourse and stochastic dynamic programs with discrete time. *ZAMM-Journal of Applied Mathematics and Mechanics/Zeitschrift für Angewandte Mathematik und Mechanik: Applied Mathematics and Mechanics*, *82*(11-12), 753-765.

Stochastic Programming

Exogenous uncertainty

- Uncertainty parameter realizations are independent of decisions:

Eg. Stock prices for individual investors, Oil/gas reserve amount of wells to be drilled, Product demands for small business owners

Endogenous uncertainty

- Uncertainty parameter realizations are influenced by decisions:
  - Type I: Decisions impact the probability distributions.

  Eg. Block trades by institutional investors causing stock price changes
  - Type II: Decisions impact the observations.

  Eg. Shale gas reserve amount revealed upon drilling

Goel, Vikas, and Ignacio E. Grossmann. "A class of stochastic programs with decision dependent uncertainty." Mathematical programming 108.2-3 (2006): 355-394.

Stochastic Programming - Static & Exhaustive

Exogenous uncertainty

- Uncertainty parameter realizations are independent of decisions:

E.g. Stock prices, Oil/gas reserve amount, Product demands

Endogenous uncertainty

- Uncertainty parameter realizations are influenced by decisions:
  - ~~Type I: Decisions impact the probability distributions.~~
  - Type II: Decisions impact the observations.

General form of multistage stochastic programming:

$$\min_{x,y,z} \sum_{w \in W} p_w \cdot \sum_{t=1}^{T} f_{w,t}(x_{w,t}, y_{w,t})$$

w: Scenarios; t: Stages

$$\text{s. t. } g_{w,t}(x_w, y_w) \le 0, \;\; w \in W, t = 0, 1, \dots, T$$

Endogenous non-anticipativity disjunctions:

$$Z_{w,w',t} \Leftrightarrow H_t(Y_w), \;\; (w, w', t) \in SP_N$$

$$\begin{pmatrix} Z_{w,w',t} \\ x_{w,t} = x_{w',t} \\ y_{w,t} = y_{w',t} \end{pmatrix} \vee \left(\neg Z_{w,w',t}\right), \;\; (w, w', t) \in SP_N$$

Exogenous non-anticipativity:

$$x_{w,t} = x_{w',t}, y_{w,t} = y_{w',t}, (w, w', t) \in SP_x$$

x: Continuous decision variables

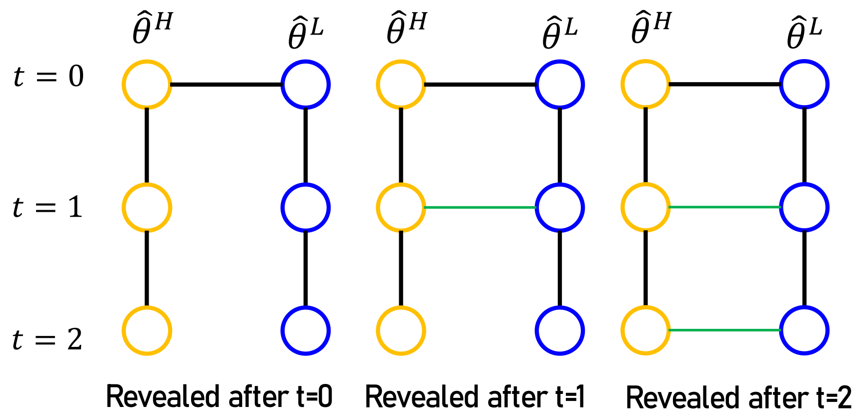
y: Binary decision variables;

z: Binary indicating variables of scenario revelation

Non-anticipativity: Consistent decision down to the last shared time point of two scenarios.

Scenario tree

θ: Endogenous uncertainty



Apap, Robert M., and Ignacio E. Grossmann. "Models and computational strategies for multistage stochastic programming under endogenous and exogenous uncertainties." Computers & Chemical Engineering 103 (2017): 233-274.

Markov Decision Process – Dynamic & Recursive

State representation

- The system (the entity to model) transitions among a set of finite states

E.g. A machine working, or broken

Random turn-out with Markovian property

- Probability distributions only depend on the current state

General form of finite horizon MDP optimal condition

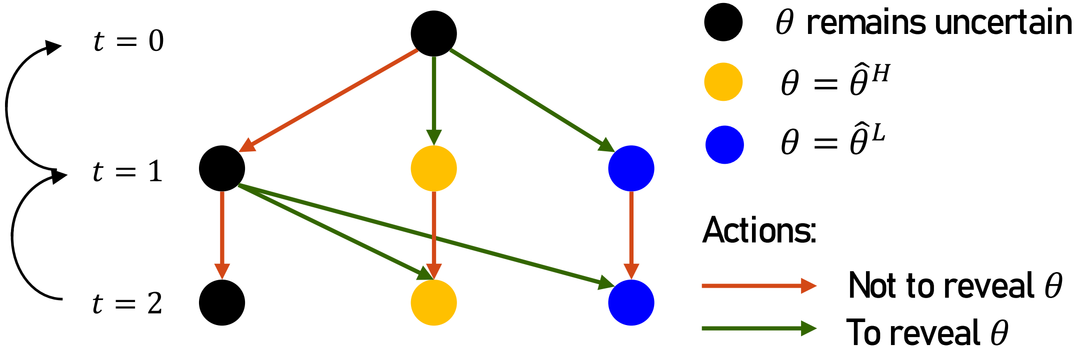
For $t = 0, \ldots, T$

$$v(s_t) = \max_{a_t}[f(s_t, a_t) + \gamma \sum_{s_{t+1}} P(s_{t+1}|s_t, a_t) v(s_{t+1})]$$

$$v(s_T) = V_T(s_T)$$

t: Stages
s: States

Transition diagram



Stochastic Programming

- Look ahead into future uncertainty with flexible form:
  - Relationship between current stage decision and next stage behaviors can be described with constraints
- Reasonable number of stages (scenarios)
- Finite horizon

$$\min_{x,y,z} \sum_{w\in W} p_w \cdot \sum_{t=1}^{T} f_{w,t}(x_{w,t}, y_{w,t})$$

$$\text{s.t. } g_{w,t}(x_w, y_w) \le 0,\ \ w \in W, t = 0,1,\dots,T$$

$$Z_{w,w',t} \Leftrightarrow H_t(Y_w),\ \ (w,w',t) \in SP_N$$

$$\begin{pmatrix} Z_{w,w',t} \\ x_{w,t} = x_{w',t} \\ y_{w,t} = y_{w',t} \end{pmatrix} \vee \left(Z_{w,w',t}\right),\ \ (w,w',t) \in SP_N$$

$$x_{w,t} = x_{w',t}, y_{w,t} = y_{w',t}, (w,w',t) \in SP_x$$

Markov Decision Process

- Look ahead into future uncertainty with recursive structure:
  - Has state representation and corresponding Markovian behavior
- Reasonable number of states
- Can deal with infinite horizon dynamics

For $t = 1, \dots, T$

$$v(s_t) = \max_{a_t} f(s_t, a_t) + \gamma \sum_{s_{t+1}\in S} P(s_{t+1}|s_t, a_t) v(s_{t+1})$$

$$v(s_T) = V_T$$

t: Stages
s: States

Finite horizon

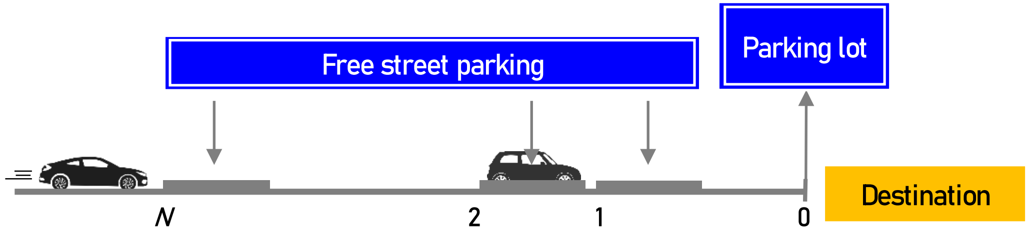
Static
Exhaustive

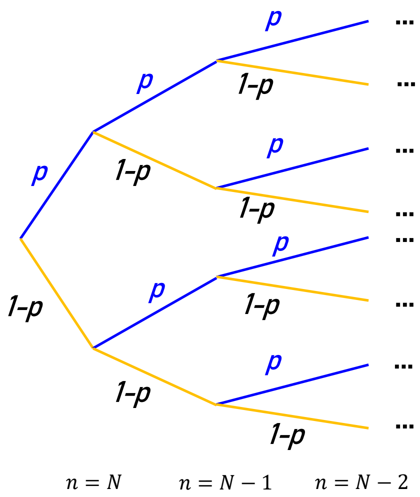
$$\min_{\substack{\text{action}_0 \\ + \\ action_t}} payback_0 + E\left(\sum_{t=1}^{T} payback_t(action_t, uncertainty_t)\right)$$

Dynamic
Recursive

Solve a problem with both tools – parking problem

- You are driving to the destination from street parking spot *N*, and you can observe whether parking spot *n* is empty only when arriving at the spot.
- By probability *p*, a spot is empty; By probability *1-p*, a spot is occupied.
- The parking lot is always available with fee ***c*** (>1).
- The inconvenience penalty of parking at street parking spot *n* is ***n***.





Decision to make at spot *n=1,…,N*:

Park if possible OR Keep looking for closer spot

Stochastic Programming

- Indicating parameter $\delta_{n,s}$
  - $\delta_{n,s} = 0$: Spot n is occupied in scenario s;
  - $\delta_{n,s} = 1$: Spot n is empty in scenario s;
- Binary variable $y_{n,s}$
  - $y_{n,s} = 0$: In scenario s, do not park in spot n;
  - $y_{n,s} = 1$: In scenario s, park in spot n;
- Variable c_s: Cost of scenario s
- Variable p_s: Probability of scenario s. $p_s = \prod_{n=1}^{N}(p \cdot \delta_{n,s} + (1-p)(1-\delta_{n,s}))$
- MILP model:

$$\min \sum_s c_s \cdot p_s$$

s. t.

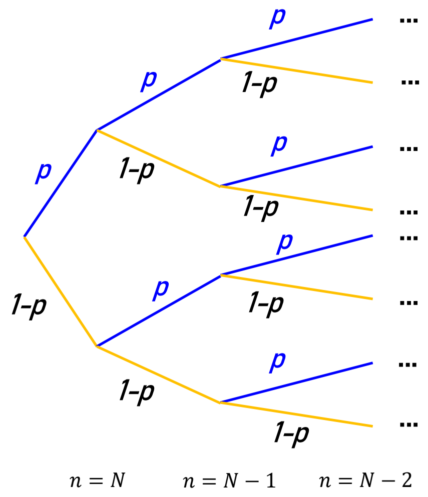
$c_s =$ cost of street parking $+$ cost of lot parking, $\forall s \in S$

Park only when empty

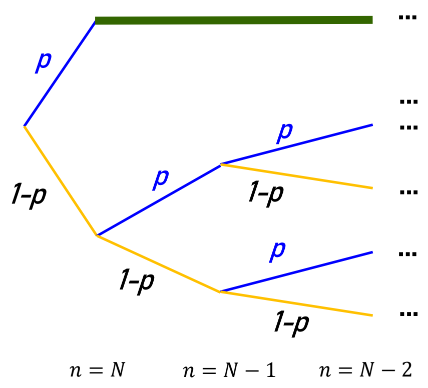
Each scenario park at most one spot

Non-anticipativity constraints

"Keep driving anyway"



"Park when the farthest spot is available"



Stochastic Programming

- Indicating parameter $\delta_{n,s}$
  - $\delta_{n,s} = 0$: Spot n is occupied in scenario s;
  - $\delta_{n,s} = 1$: Spot n is empty in scenario s;
- Binary variable $y_{n,s}$
  - $y_{n,s} = 0$: In scenario s, do not park in spot n;
  - $y_{n,s} = 1$: In scenario s, park in spot n;
- Variable c_s: Cost of scenario s
- Variable p_s: Probability of scenario s. $p_s = \prod_{n=1}^{N}(p \cdot \delta_{n,s} + (1-p)(1-\delta_{n,s}))$
- MILP model:

$$\min \sum_s c_s \cdot p_s$$

s. t.

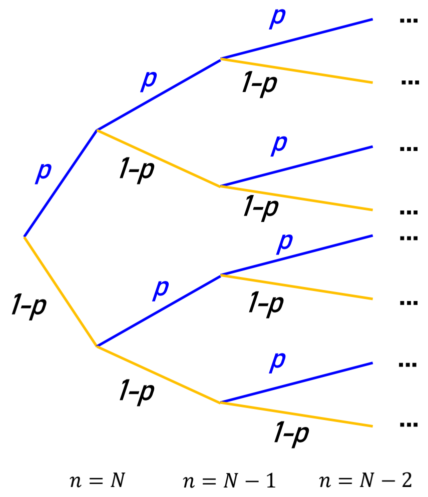
$$c_s = \sum_{n=1}^{N} y_{n,s} \cdot n + \left(1 - \sum_{n=1}^{N} y_{n,s}\right) \cdot c, \qquad \forall s \in S$$

$$y_{n,s} \le \delta_{n,s}, \forall 1 \le n \le N, s \in S$$

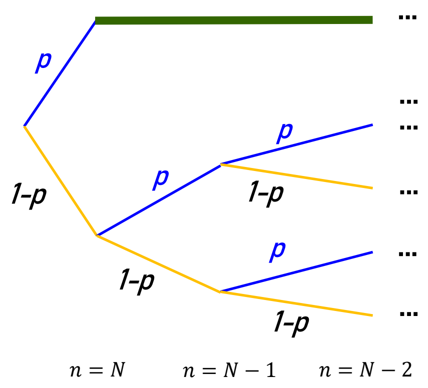
$$\sum_{n=1}^{N} y_{n,s} \le 1, \qquad \forall s \in S$$

$$y_{n,s} = y_{n,s'}, \qquad \forall 1 \le n \le N_{s,s'}^{parent}, s, s' \in S$$

"Keep driving anyway"



"Park when the farthest spot is available"



Stochastic Programming

$$\min \sum_{s} c_s \cdot p_s$$

s.t.

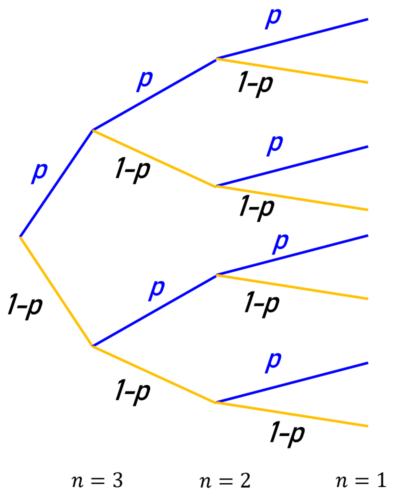
$$c_s = \sum_{n=1}^{N} y_{n,s} \cdot n + \left(1 - \sum_{n=1}^{N} y_{n,s}\right) \cdot c, \qquad \forall s \in S$$

$$y_{n,s} \le \delta_{n,s}, \forall 1 \le n \le N, s \in S$$

$$\sum_{n} y_{n,s} \le 1, \qquad \forall s \in S$$

$$y_{n,s} = y_{n,s'}, \qquad \forall 1 \le n \le N_{s,s'}^{parent}, s, s' \in S$$

$\boldsymbol{N = 3,\ p = 0.6,\ c = 4}$



Result

	$s=1$		$s=2$		$s=3$		$s=4$		$s=5$		$s=6$		$s=7$		$s=8$	
	$\delta_{n,s}$	$y_{n,s}$	$\delta_{n,s}$	$y_{n,s}$	$\delta_{n,s}$	$y_{n,s}$	$\delta_{n,s}$	$y_{n,s}$	$\delta_{n,s}$	$y_{n,s}$	$\delta_{n,s}$	$y_{n,s}$	$\delta_{n,s}$	$y_{n,s}$	$\delta_{n,s}$	$y_{n,s}$
$n=1$	0	0	1	1	0	0	1	0	0	0	1	1	0	0	1	0
$n=2$	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
$n=3$	0	0	0	0	0	0	0	0	1	0	1	0	1	0	1	0

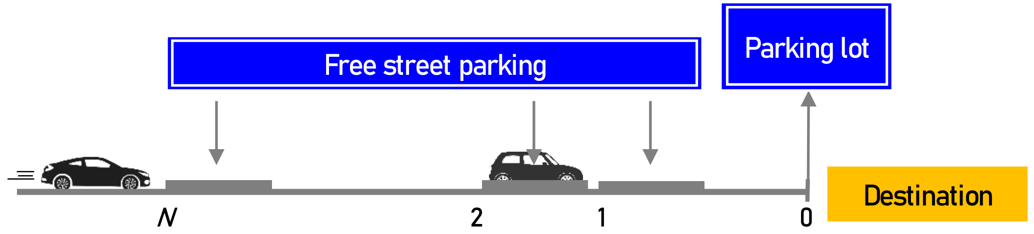
$y_{n,s} = 0$: In scenario s, do not park in spot n;
$y_{n,s} = 1$: In scenario s, park in spot n;

```
├--- VAR y

        LOWER     LEVEL     UPPER    MARGINAL

1.1       .         .       1.000    -0.192
1.2       .       1.000     1.000    -0.288
1.3       .         .       1.000    -0.288
1.4       .         .       1.000    -0.432
1.5       .         .       1.000    -0.288
1.6       .       1.000     1.000    -0.432
1.7       .         .       1.000    -0.432
1.8       .         .       1.000    -0.648
2.1       .         .       1.000    -0.128
2.2       .         .       1.000    -0.192
2.3       .       1.000     1.000    -0.192
2.4       .       1.000     1.000    -0.288
2.5       .         .       1.000    -0.192
2.6       .         .       1.000    -0.288
2.7       .       1.000     1.000    -0.288
2.8       .       1.000     1.000    -0.432
3.1       .         .       1.000    -0.064
3.2       .         .       1.000    -0.096
3.3       .         .       1.000    -0.096
3.4       .         .       1.000    -0.144
3.5       .         .       1.000    -0.096
3.6       .         .       1.000    -0.144
3.7       .         .       1.000    -0.144
3.8       .         .       1.000    -0.216
```

Markov Decision Process



- Recursive backtracking

- <u>State space</u>:

  $\{(n,i)|1 \leq n \leq N, i \in \{0,1\}\} + \{(0,1)\} + \{\text{Parked}\}$

  $i = 0$: Cannot park :, $i = 1$: Can park

- <u>Action space</u>:

  $A_{n,0} = \{\text{keep looking}\}$, $A_{n,1} = \{\text{Park, Keep looking}\}$,
  $A_{0,1} = \{\text{Park}\}$

- <u>Transition probabilities</u>:

  $P\big((n,0),\text{keep looking},(n-1,1)\big) = p,$

  $P\big((n,0),\text{keep looking},(n-1,0)\big) = 1-p,$

  $P\big((n,1),\text{keep looking},(n-1,1)\big) = p,$

  $P\big((n,1),\text{keep looking},(n-1,0)\big) = 1-p,$
  $P\big((n,1),\text{park},\text{parked}\big) = 1$

  $P\big((1,1),\text{keep looking},(0,1)\big) = 1$
  $P\big((1,0),\text{keep looking},(0,1)\big) = 1$

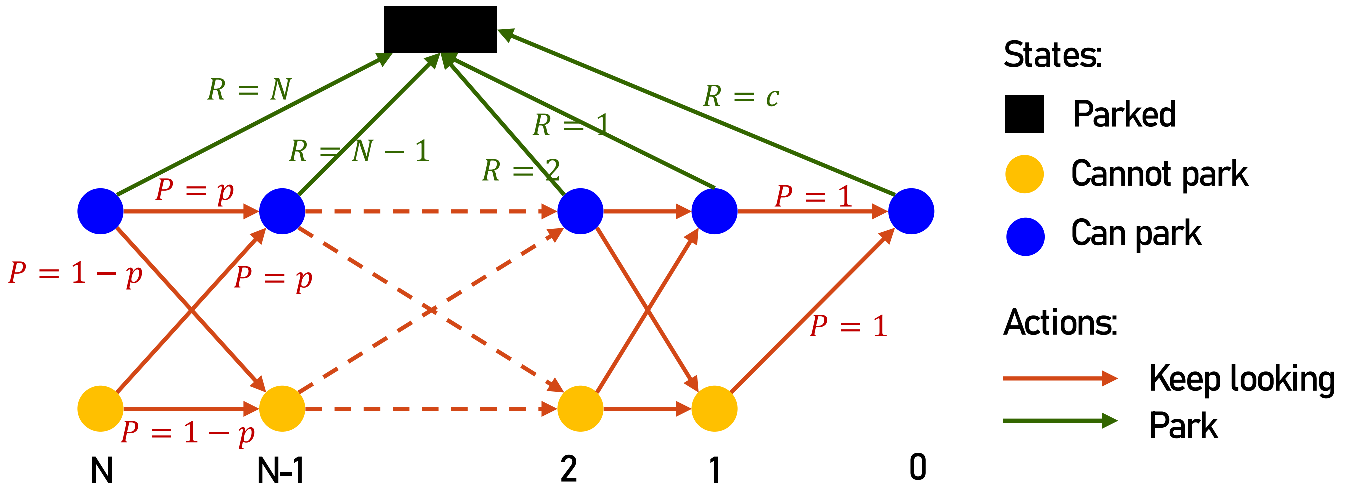
- <u>Direct cost</u>: $R\big((n,1),\text{Park},\text{Parked}\big) = n$, $R\big((0,1),\text{Park},\text{Parked}\big) = c$

- $f_{n,i}$: Optimal expected cost starting from state (n,i)

- Boundary condition: $f_0 = c$

- Recursive optimality condition:

$$f_{n,0} = p \cdot f_{n-1,1} + (1-p) \cdot f_{n-1,0}$$
$$f_{n,1} = \min\{n, p \cdot f_{n-1,1} + (1-p) \cdot f_{n-1,0}\}$$

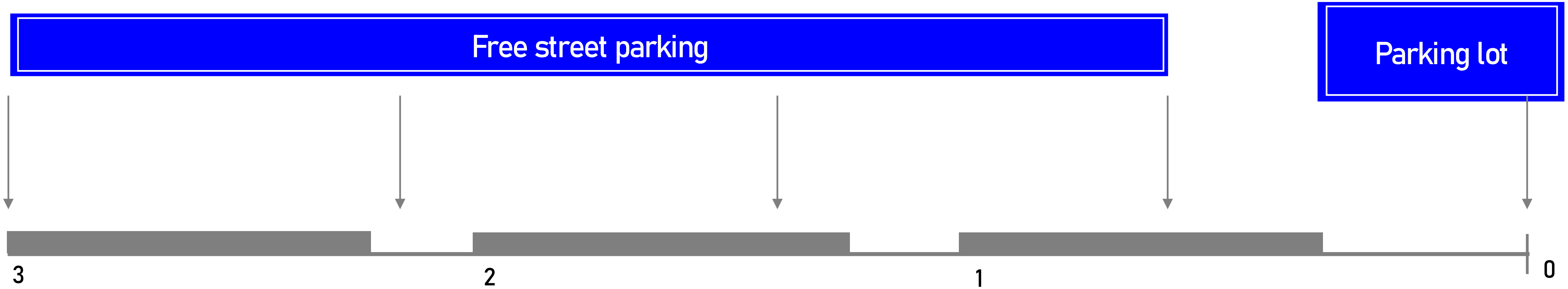


Markov Decision Process

Optimal cost at parking spot n with the spot occupied - $f_{n,0} = p \cdot f_{n-1,1} + (1-p) \cdot f_{n-1,0}$

Optimal cost at parking spot n with the spot empty - $f_{n,1} = \min\{ \underbrace{n,}_{\text{To park}} \quad \underbrace{p \cdot f_{n-1,1} + (1-p) \cdot f_{n-1,0}}_{\text{To keep looking}} \}$

$$\boldsymbol{N = 3,\ p = 0.6,\ c = 4}$$



$f_{3,0} = 0.6 \times f_{2,1} + 0.4 \times f_{2,0} = 0.6 \times 2 + 0.4 \times 2.2 = 2.08$

$f_{3,1} = \min\{3, 2.08\} = 2.08$

To keep looking

$f_{2,0} = 0.6 \times f_{1,1} + 0.4 \times f_{1,0} = 0.6 \times 1 + 0.4 \times 4 = 2.2$

$f_{2,1} = \min\{2, 2.2\} = 2$

To park

$f_{1,0} = c = 4$

$f_{1,1} = \min\{1, 4\} = 1$

To park

Destination

Stochastic Programming

- Look ahead into future uncertainty with flexible form:
  - Relationship between current stage decision and next stage behaviors can be described with polytopes
- Reasonable number of stages (scenarios)
- Finite horizon

$$\min_{x,y,z} \sum_{w\in W} p_w \cdot \sum_{t=1}^{T} f_{w,t}(x_{w,t}, y_{w,t})$$

$$\text{s. t. } g_{w,t}(x_w, y_w) \le 0,\ w\in W, t = 0,1,\dots,T$$

$$Z_{w,w',t} \Leftrightarrow H_t(Y_w),\ (w,w',t)\in SP_N$$

$$\begin{pmatrix} Z_{w,w',t} \\ x_{w,t} = x_{w',t} \\ y_{w,t} = y_{w',t} \end{pmatrix} \vee \left(Z_{w,w',t}\right),\ (w,w',t)\in SP_N$$

$$x_{w,t} = x_{w',t}, y_{w,t} = y_{w',t}, (w,w',t)\in SP_x$$

Markov Decision Process

- Look ahead into future uncertainty with recursive structure:
  - Has state representation and corresponding Markovian behavior
- Reasonable number of states
- Can deal with infinite horizon dynamics

For $t = 1,\dots,T$

$$v(s_t) = \max_{a_t} f(s_t, a_t) + \gamma \sum_{s_{t+1}\in S} P(s_{t+1}|s_t, a_t) v(s_{t+1})$$

$$v(s_T) = V_T$$

t: Stages
s: States

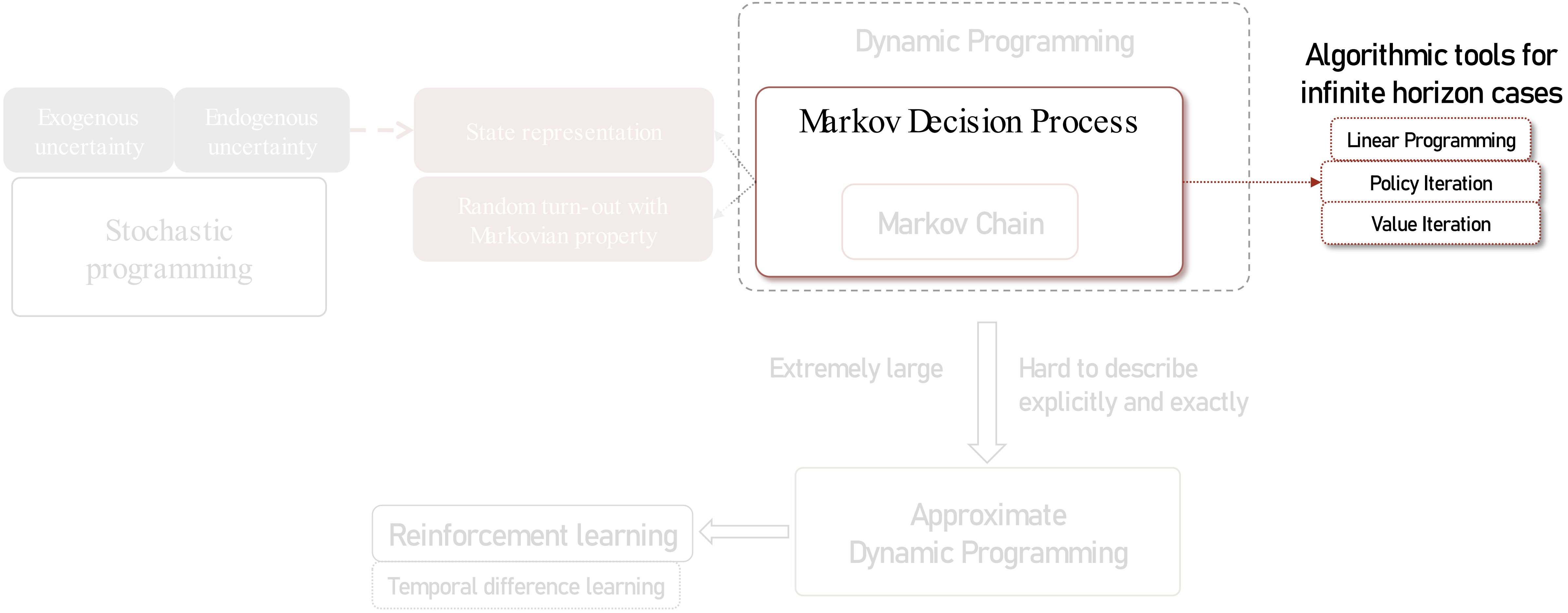
Finite horizon

Static
Exhaustive

$$\min_{\substack{action_0 \\ + \\ action_t}} payback_0 + E\left(\sum_{t=1}^{T} payback_t(action_t, uncertainty_t)\right)$$

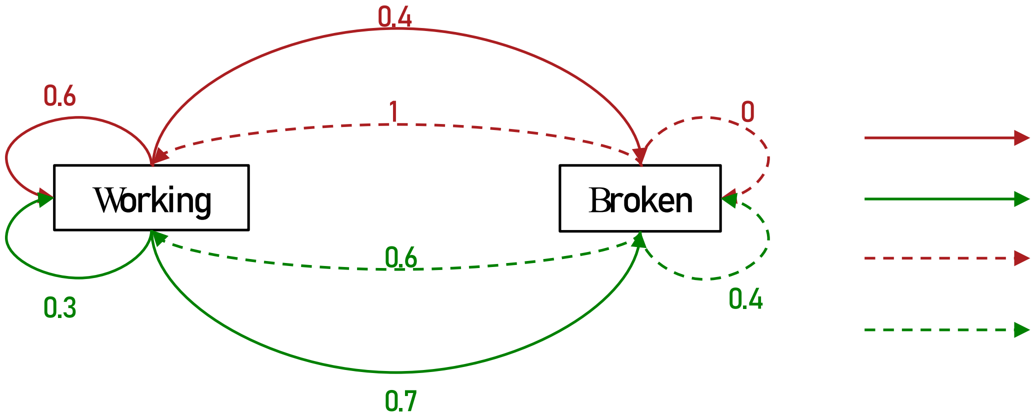
Dynamic
Recursive

Things to cover



Infinite horizon Markov Decision Process – to make a maintenance decision

- Consider a machine that is either running or is broken.
- If it runs throughout one week, it makes a gross profit of $100. If it fails during the week, gross profit is 0.

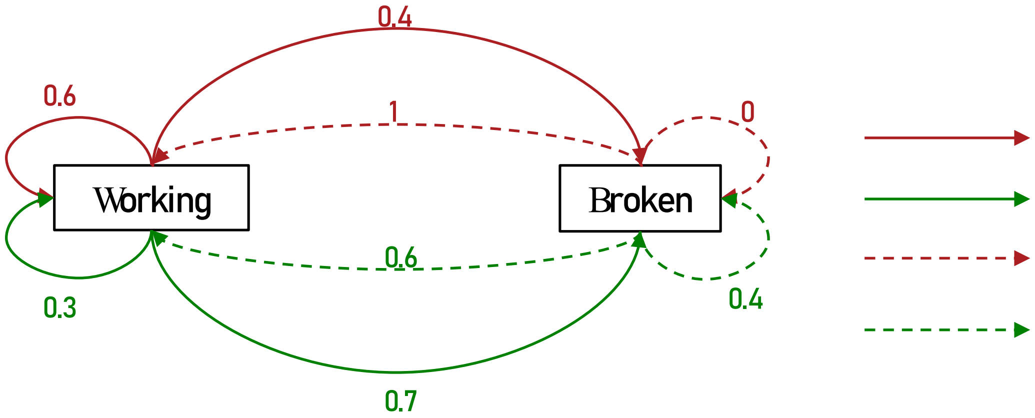


States	Actions	C(action)
Working	Maintenance	20
	Wait	0
Broken	Replace	150
	Repair	40

- Purpose: find the best action for each state
- <u>State space</u> $S = \{\text{Working}, \text{Broken}\}$
- <u>Action space</u> $A(\text{current state})$: $A(\text{Working}) = \{\text{Maintenance}, \text{Wait}\}$, $A(\text{Broken}) = \{\text{Replace}, \text{Repair}\}$
- <u>Transition probabilities</u> $P(\text{current state}, \text{action}, \text{next state})$: as shown in the graph
- <u>Direct reward</u> $R(\text{current state}, \text{action})$: $-C(\text{action}) + \mathrm{E}(\text{gross profit}|\text{action})$
- Discount factor $\gamma = 0.8$

Infinite horizon Markov Decision Process – to make a maintenance decision

- Consider a machine that is either running or is broken.
- If it runs throughout one week, it makes a gross profit of \$100. If it fails during the week, gross profit is 0.



States	Actions	C(action)
Working	Maintenance	20
	Wait	0
Broken	Replace	150
	Repair	40

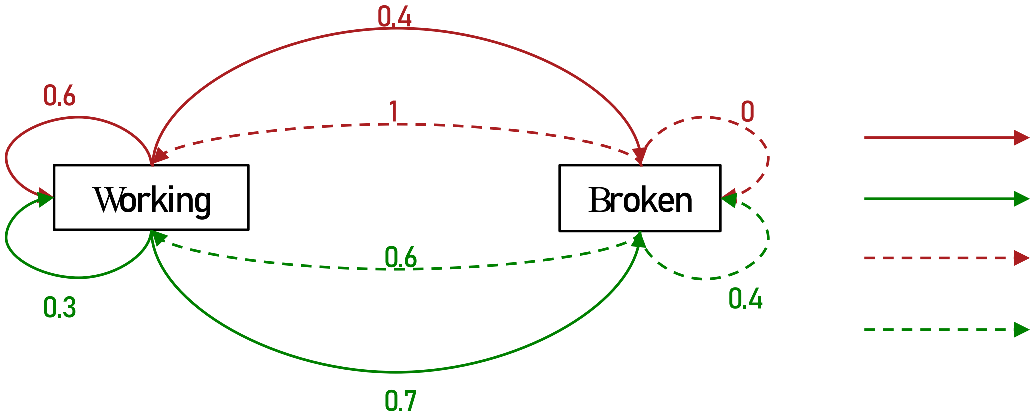
- Purpose: find the best action for each state

$$v(\text{current state}) = \max_{\text{action}\in\{\text{Actions}\}} \{-C(\text{action}) + \text{E}(\text{gross profit}|\text{action}) + \gamma \text{E}(v(\text{next state})|\text{action})\}$$

Optimality condition

Infinite horizon Markov Decision Process – to make a maintenance decision

- Consider a machine that is either running or is broken.
- If it runs throughout one week, it makes a **gross profit** of $100. If it fails during the week, **gross profit** is 0.

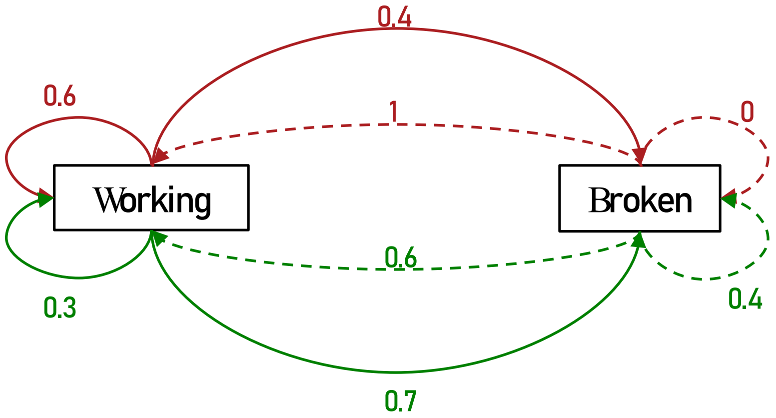


States	Actions	C(action)
Working	Maintenance	20
	Wait	0
Broken	Replace	150
	Repair	40

- Let the optimal value of **Working** and **Broken** be $v(W)$ and $v(B)$.
- $v(W) = \max\left\{-20 + \left(0.6(0.8v(W) + 100) + 0.4\left(0.8v(B)\right)\right), \left(0.3(0.8v(W) + 100) + 0.7\left(0.8v(B)\right)\right)\right\}$
- $v(B) = \max\{-150 + \left((0.8v(W) + 100)\right), -40 + \left(0.6(0.8v(W) + 100) + 0.4\left(0.8v(B)\right)\right)\}$

Infinite horizon Markov Decision Process – to make a maintenance decision

- Consider a machine that is either running or is broken.
- If it runs throughout one week, it makes a **gross profit** of $100. If it fails during the week, **gross profit** is 0.



States	Actions	C(action)
Working	Maintenance	20
	Wait	0
Broken	Replace	150
	Repair	40

- Let the optimal value of **Working** and **Broken** be $v(W)$ and $v(B)$.
- $v(W) = \max\left\{-20 + \left(0.6(0.8v(W) + 100) + 0.4\left(0.8v(B)\right)\right), \left(0.3(0.8v(W) + 100) + 0.7\left(0.8v(B)\right)\right)\right\}$

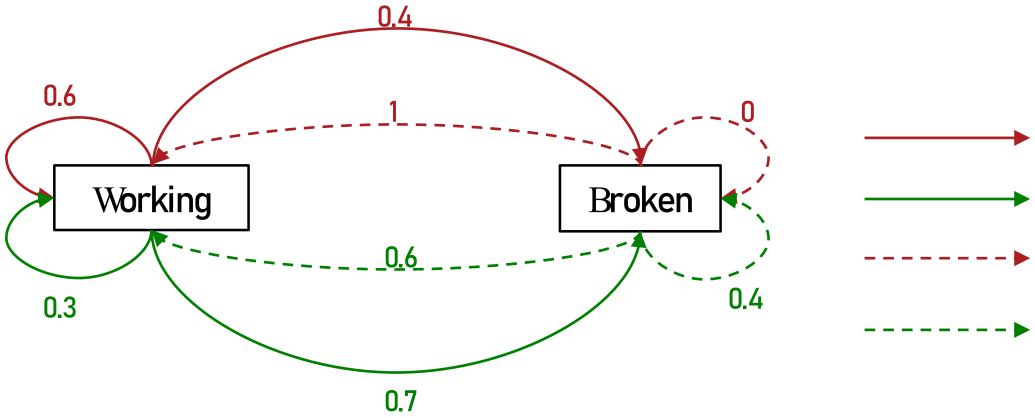
  $\rightarrow$ $v(W) = \max\{0.32v(B) + 0.48v(W) + 40, 0.56v(B) + 0.24v(W) + 30\}$

- $v(B) = \max\left\{-150 + \left((0.8v(W) + 100)\right), -40 + \left(0.6(0.8v(W) + 100) + 0.4\left(0.8v(B)\right)\right)\right\}$

  $\rightarrow$ $v(B) = \max\{0.8v(W) - 50, 0.32v(B) + 0.48v(W) + 20\}$

Infinite horizon Markov Decision Process – to make a maintenance decision

- Consider a machine that is either running or is broken.
- If it runs throughout one week, it makes a **gross profit** of $100. If it fails during the week, **gross profit** is 0.



States	Actions	C(action)
Working	Maintenance	20
	Wait	0
Broken	Replace	150
	Repair	40

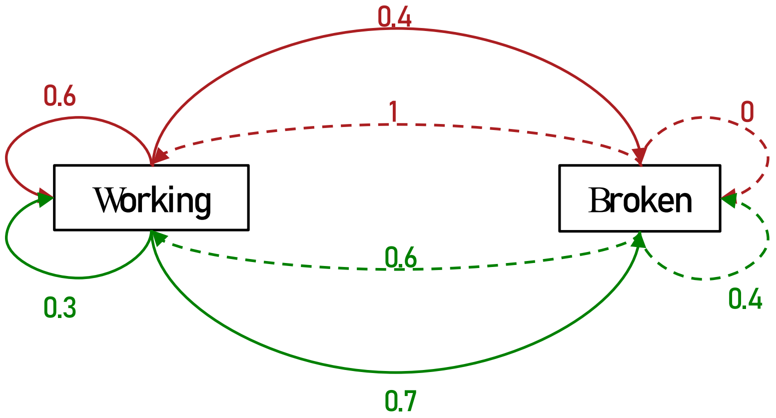
- Let the optimal value of **Working** and **Broken** be $v(W)$ and $v(B)$.
- $v(W) = \max\{0.32v(B) + 0.48v(W) + 40, 0.56v(B) + 0.24v(W) + 30\}$ → $v(W) \geq 0.32v(B) + 0.48v(W) + 40, v(W) \geq 0.56v(B) + 0.24v(W) + 30\}$
- $v(B) = \max\{0.8v(W) - 50, 0.32v(B) + 0.48v(W) + 20\}$ → $v(B) \geq 0.8v(W) - 50, v(B) \geq 0.32v(B) + 0.48v(W) + 20\}$

$$\min v(W) + v(B)$$

Linear Programming

Infinite horizon Markov Decision Process – to make a maintenance decision

- Consider a machine that is either running or is broken.
- If it runs throughout one week, it makes a **gross profit** of $100. If it fails during the week, **gross profit** is 0.



States	Actions	C(action)
Working	Maintenance	20
	Wait	0
Broken	Replace	150
	Repair	40

- Let the optimal value of **Working** and **Broken** be $v(W)$ and $v(B)$.
- $v(W) = \max\{0.32v(B) + 0.48v(W) + 40, 0.56v(B) + 0.24v(W) + 30\}$
- $v(B) = \max\{0.8v(W) - 50, 0.32v(B) + 0.48v(W) + 20\}$

$\min v(W) + v(B)$

→ $v(W) \geq 0.32v(B) + 0.48v(W) + 40, v(W) \geq 0.56v(B) + 0.24v(W) + 30$

→ $v(B) \geq 0.8v(W) - 50, v(B) \geq 0.32v(B) + 0.48v(W) + 20$

LP special property

General form of the previous problem:

$$\boldsymbol{v}^* = \arg\min_{\boldsymbol{v}} \mathbf{1}^{\mathrm{T}}\boldsymbol{v}$$

s.t. $\boldsymbol{v} \succcurlyeq H_\delta \boldsymbol{v}, \quad \forall \boldsymbol{\delta} \in \Delta$ $\quad\quad$ $\Delta = A(1) \times \cdots \times A(|S|)$ is policy space

For $\boldsymbol{v}$ that satisfy $\boldsymbol{v} \succcurlyeq H_\delta \boldsymbol{v}$, applying the operator H_δ again gives $H_\delta \boldsymbol{v} \succcurlyeq H_\delta(H_\delta \boldsymbol{v})$, therefore $\boldsymbol{v} \succcurlyeq H_\delta \boldsymbol{v} \succcurlyeq \cdots \succcurlyeq \lim_{n\to\infty} H_\delta^n \boldsymbol{v} = \boldsymbol{v}^*$ → $\boldsymbol{v}^*$ is the element-wise minimum

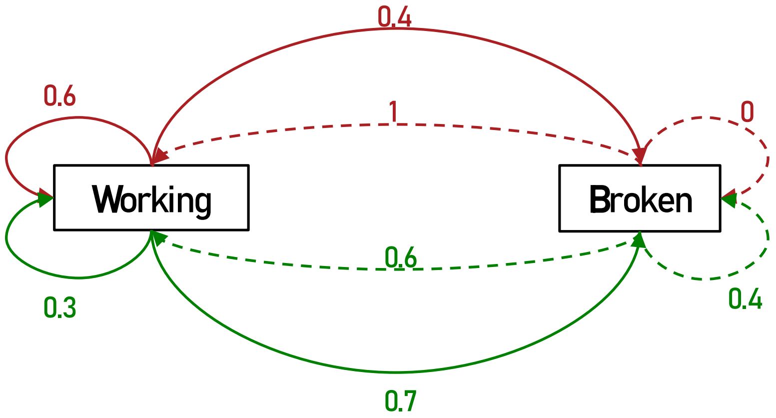
$$\boldsymbol{v}^* = \arg\min_{\boldsymbol{v}} \boldsymbol{w}^{\mathrm{T}}\boldsymbol{v}$$

where $\boldsymbol{w}$ is any positive weight vector

s.t. $\boldsymbol{v} \succcurlyeq H_\delta \boldsymbol{v}, \quad \forall \boldsymbol{\delta} \in \Delta$ $\quad\quad$ $\Delta = A(1) \times \cdots \times A(|S|)$ is policy space

Infinite horizon Markov Decision Process – to make a maintenance decision

- Consider a machine that is either running or is broken.
- If it runs throughout one week, it makes a **gross profit** of $100. If it fails during the week, **gross profit** is 0.



States	Actions	C(action)
Working	Maintenance	20
	Wait	0
Broken	Replace	150
	Repair	40

- Let the optimal value of **Working** and **Broken** be $v(W)$ and $v(B)$.
- $v(W) = \max\{0.32v(B) + 0.48v(W) + 40, 0.56v(B) + 0.24v(W) + 30\}$
- $v(B) = \max\{0.8v(W) - 50, 0.32v(B) + 0.48v(W) + 20\}$

$$\min 40\alpha(W,Mt) + 30\alpha(W,Wt) - 50\alpha(B,Re) + 20\alpha(B,Rr)$$

$$\text{s.t. } 0.52\alpha(W,Mt) + 0.76\alpha(W,Wt) - 0.8\alpha(B,Re) - 0.48\alpha(B,Rr) = 1$$

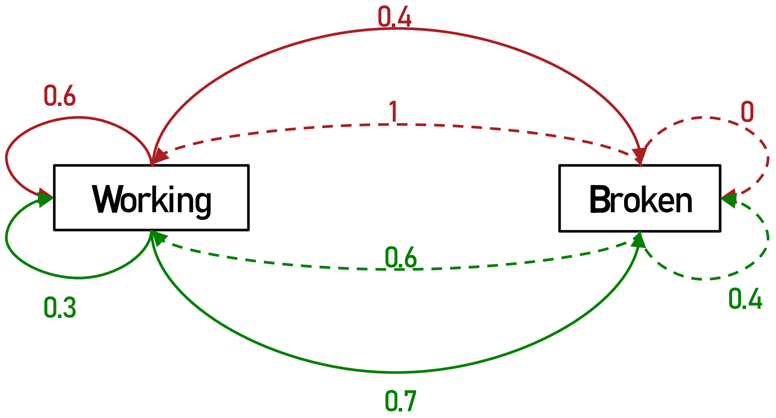
$$-0.32\alpha(W,Mt) - 0.56\alpha(W,Wt) + \alpha(B,Re) + 0.68\alpha(B,Rr) = 1$$

LP dual

$\alpha(current\ state, action) > 0$: The action is chosen for the state
$\alpha(current\ state, action) = 0$: The action is not chosen for the state

Infinite horizon Markov Decision Process – to make a maintenance decision

- Consider a machine that is either running or is broken.
- If it runs throughout one week, it makes a **gross profit** of $100. If it fails during the week, **gross profit** is 0.



States	Actions	C(action)
Working	Maintenance	20
	Wait	0
Broken	Replace	150
	Repair	40

- Let the optimal value of **Working** and **Broken** be $v(W)$ and $v(B)$.
- $v(W) = \max\{0.32v(B) + 0.48v(W) + 40, 0.56v(B) + 0.24v(W) + 30\}$
- $v(B) = \max\{0.8v(W) - 50, 0.32v(B) + 0.48v(W) + 20\}$

$$\min 40\alpha(W,Mt) + 30\alpha(W,Wt) - 50\alpha(B,Re) + 20\alpha(B,Rr)$$

$$\text{s.t. } 0.52\alpha(W,Mt) + 0.76\alpha(W,Wt) - 0.8\alpha(B,Re) - 0.48\alpha(B,Rr) = 1$$

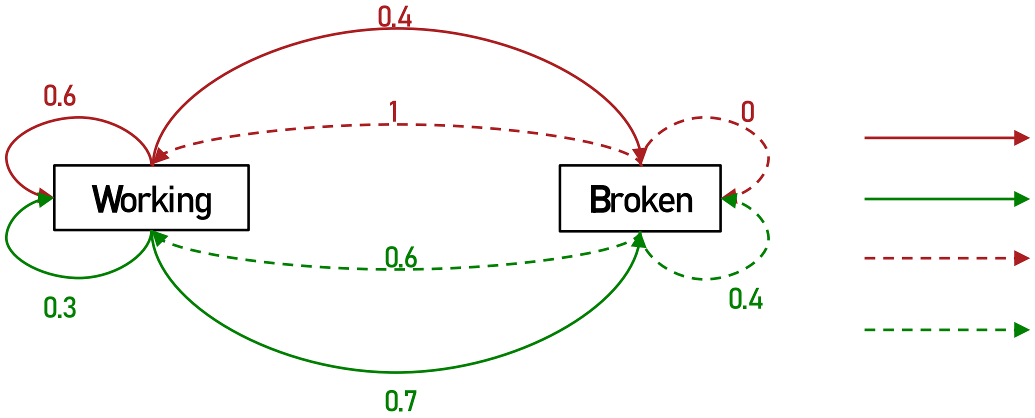
$$-0.32\alpha(W,Mt) - 0.56\alpha(W,Wt) + \alpha(B,Re) + 0.68\alpha(B,Rr) = 1$$

$\alpha(current\ state, action) > 0$: The action is chosen for the state

$\alpha(current\ state, action) = 0$: The action is not chosen for the state

Infinite horizon Markov Decision Process – to make a maintenance decision

- Consider a machine that is either running or is broken.
- If it runs throughout one week, it makes a **gross profit** of $100. If it fails during the week, **gross profit** is 0.

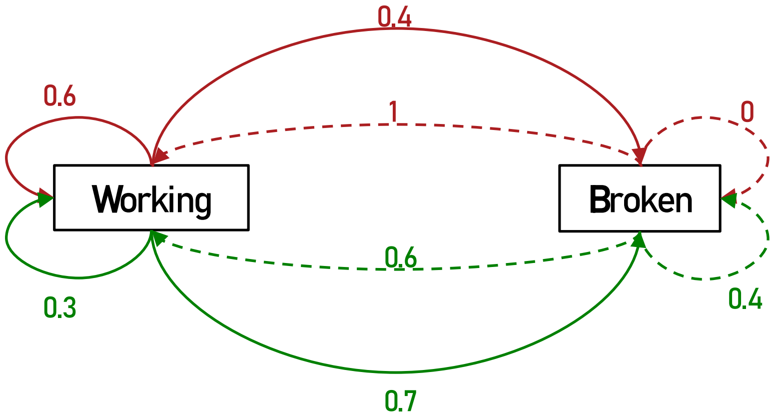


States	Actions	C(action)
Working	Maintenance	20
	Wait	0
Broken	Replace	150
	Repair	40

- Let the optimal value of **Working** and **Broken** be $v(W)$ and $v(B)$.
- $v(W) = \max\{0.32v(B) + 0.48v(W) + 40, 0.56v(B) + 0.24v(W) + 30\}$
- $v(B) = \max\{0.8v(W) - 50, 0.32v(B) + 0.48v(W) + 20\}$

Infinite horizon Markov Decision Process – to make a maintenance decision

- Consider a machine that is either running or is broken.
- If it runs throughout one week, it makes a **gross profit** of $100. If it fails during the week, **gross profit** is 0.



States	Actions	C(action)
Working	Maintenance	20
	Wait	0
Broken	Replace	150
	Repair	40

- Let the optimal value of **Working** and **Broken** be $v(W)$ and $v(B)$.
- $v(W) = \max\{0.32v(B) + 0.48v(W) + 40, 0.56v(B) + 0.24v(W) + 30\}$
- $v(B) = \max\{0.8v(W) - 50, 0.32v(B) + 0.48v(W) + 20\}$

Initialize

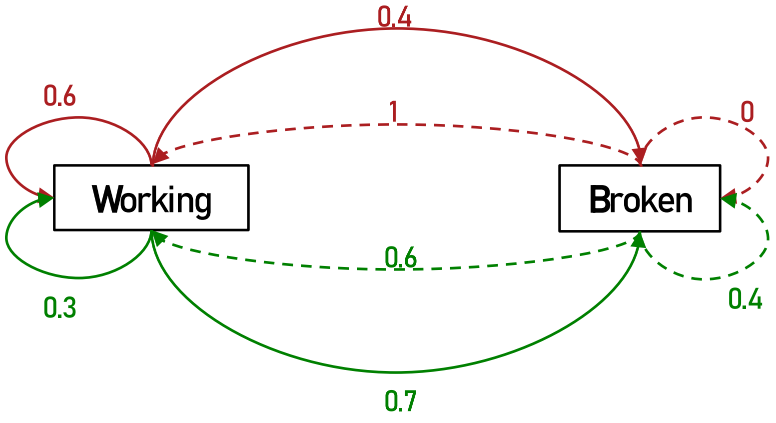
$v(B), v(W)$

Value Iteration

$v(W) \leftarrow \max\{0.32v(B) + 0.48v(W) + 40, 0.56v(B) + 0.24v(W) + 30\}$
$v(B) \leftarrow \max\{0.8v(W) - 50, 0.32v(B) + 0.48v(W) + 20\}$

Infinite horizon Markov Decision Process – to make a maintenance decision

- Consider a machine that is either running or is broken.
- If it runs throughout one week, it makes a **gross profit** of $100. If it fails during the week, **gross profit** is 0.



States	Actions	C(action)
Working	Maintenance	20
	Wait	0
Broken	Replace	150
	Repair	40

- Let the optimal value of **Working** and **Broken** be $v(W)$ and $v(B)$.
- $v(W) = \max\{0.32v(B) + 0.48v(W) + 40, 0.56v(B) + 0.24v(W) + 30\}$
- $v(B) = \max\{0.8v(W) - 50, 0.32v(B) + 0.48v(W) + 20\}$

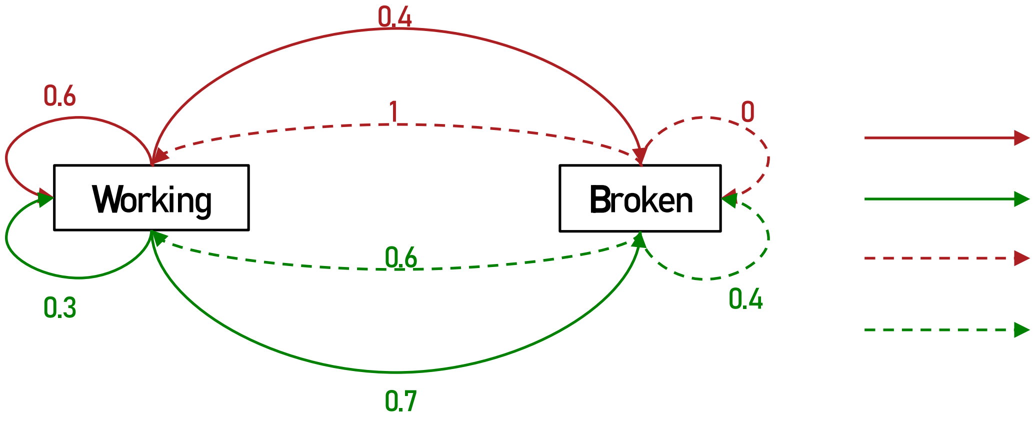
Initialize

$v(B), v(W)$

$$v(W) \leftarrow \max\{0.32v(B) + 0.48v(W) + 40, 0.56v(B) + 0.24v(W) + 30\}$$
$$v(B) \leftarrow \max\{0.8v(W) - 50, 0.32v(B) + 0.48v(W) + 20\}$$

Infinite horizon Markov Decision Process – to make a maintenance decision

- Consider a machine that is either running or is broken.
- If it runs throughout one week, it makes a **gross profit** of $100. If it fails during the week, **gross profit** is 0.

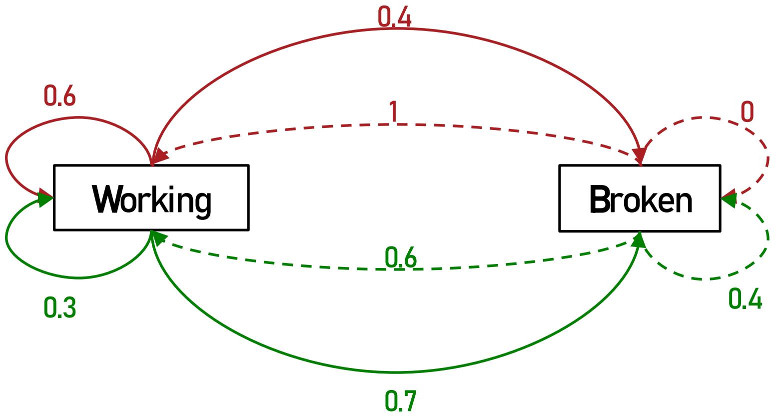


States	Actions	C(action)
Working	Maintenance	20
	Wait	0
Broken	Replace	150
	Repair	40

- Let the optimal value of **Working** and **Broken** be $v(W)$ and $v(B)$.
- $v(W) = \max\{0.32v(B) + 0.48v(W) + 40, 0.56v(B) + 0.24v(W) + 30\}$
- $v(B) = \max\{0.8v(W) - 50, 0.32v(B) + 0.48v(W) + 20\}$

Infinite horizon Markov Decision Process – to make a maintenance decision

- Consider a machine that is either running or is broken.
- If it runs throughout one week, it makes a gross profit of \$100. If it fails during the week, gross profit is 0.



States	Actions	C(action)
Working	Maintenance	20
	Wait	0
Broken	Replace	150
	Repair	40

$v(W) = 0.32v(B) + 0.48v(W) + 40,$ [redacted]

$v(B) =$ [redacted] $0.32v(B) + 0.48v(W) + 20$

Policy:
(Maintenance, Repair)

Policy Iteration

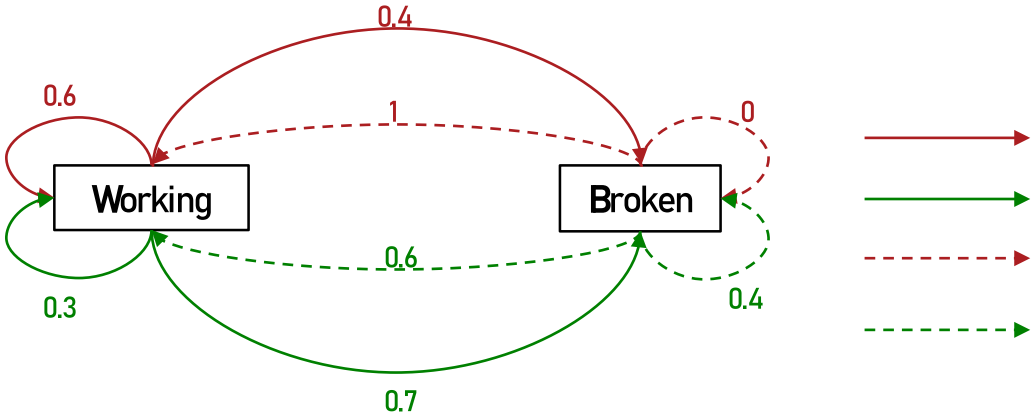
$v(B) = 148, v(W) = 168$

$v(W) \leftarrow \max\{\boxed{0.32v(B) + 0.48v(W) + 40 = 168}, 0.56v(B) + 0.24v(W) + 30 = 153.2\}$

$v(B) \leftarrow \max\{0.8v(W) - 50 = 84.4, \boxed{0.32v(B) + 0.48v(W) + 20 = 148}\}$

Infinite horizon Markov Decision Process – to make a maintenance decision

- Consider a machine that is either running or is broken.
- If it runs throughout one week, it makes a gross profit of $100. If it fails during the week, gross profit is 0.



States	Actions	*C*(action)
Working	Maintenance	20
	Wait	0
Broken	Replace	150
	Repair	40

$$v(W) = 0.32v(B) + 0.48v(W) + 40,$$ [redacted]

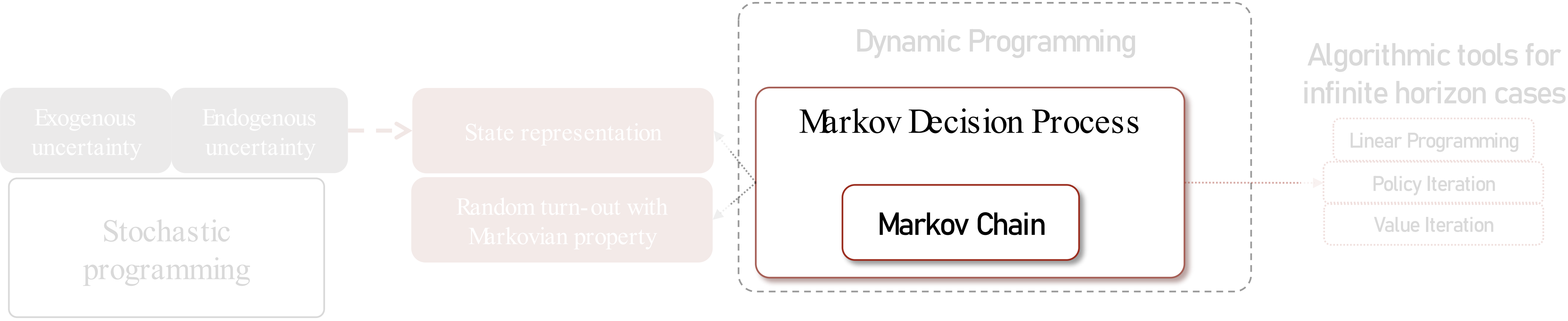
$$v(B) =$$ [redacted] $$, 0.32v(B) + 0.48v(W) + 20$$

Policy:
(Maintenance, Repair)

$$v(B) = 148, v(W) = 168$$

$$v(W) \leftarrow \max\{\boxed{0.32v(B) + 0.48v(W) + 40 = 168}, 0.56v(B) + 0.24v(W) + 30 = 153.2\}$$
$$v(B) \leftarrow \max\{0.8v(W) - 50 = 84.4, \boxed{0.32v(B) + 0.48v(W) + 20 = 148}\}$$

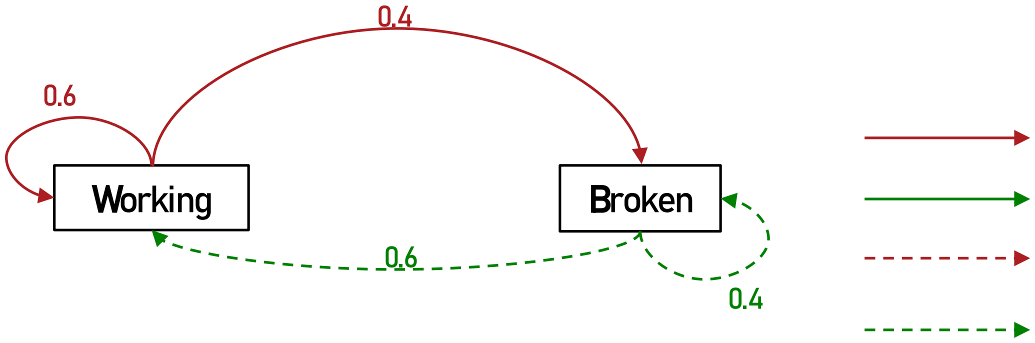
Things to cover



- Markov Decision Process is the superstructure of Markov Chains on action space;
- Markov Decision Process reduces to Markov Chain when the actions are fixed.

Infinite horizon Markov Decision Process – to make a maintenance decision

- Consider a machine that is either running or is broken.
- If it runs throughout one week, it makes a gross profit of $100. If it fails during the week, gross profit is 0.



States	Actions	C(action)
Working	Maintenance	20
	~~Wait~~	0
Broken	~~Replace~~	150
	Repair	40

Transition probability matrix

	Working	Broken
Working	0.6	0.4
Broken	0.6	0.4

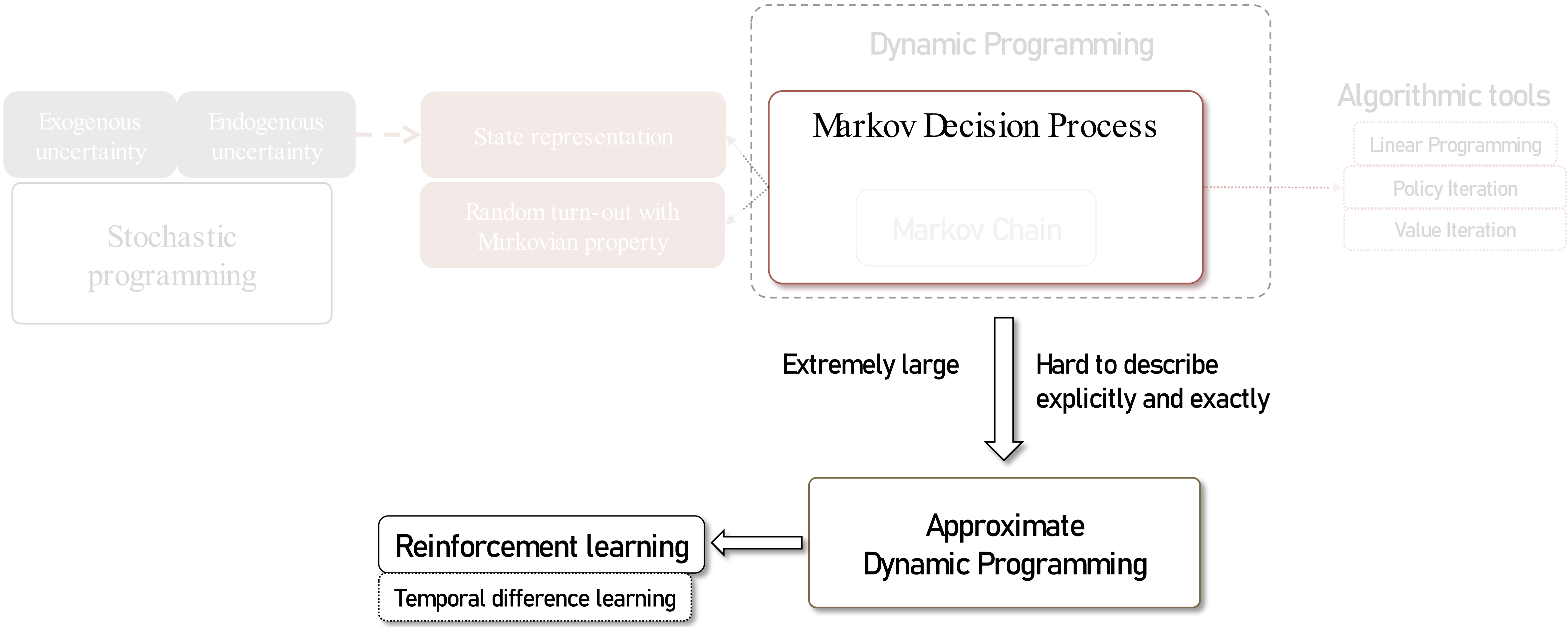
Stationary probability:

$$[\Pr(Working), \Pr(Broken)] \cdot \begin{bmatrix} 0.6, 0.4 \\ 0.6, 0.4 \end{bmatrix} = [\Pr(Working), \Pr(Broken)]$$

$$\Pr(Working) + \Pr(Broken) = 1$$

$$\Pr(Working) = 0.6, \Pr(Broken) = 0.4$$

Things to cover



Reinforcement learning – simulation based optimization

Temporal difference learning: update state-action value function after every interaction with the environment.

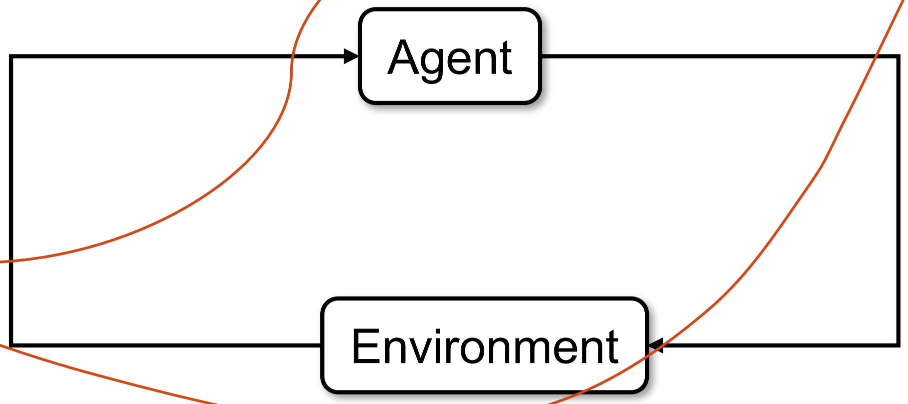
Recall: Optimal condition $v(s) = \max_{a \in A_s}\{\mathrm{E}_{s'}(R(s,a,s')|a) + \gamma \mathrm{E}_{s'}(v(s')|a)\}, \forall s \in S$

<u>Q learning:</u>

- Look-up table of $\boldsymbol{Q}(\boldsymbol{s_t}, \boldsymbol{a_t})$
- Parameterize $\boldsymbol{Q}(\boldsymbol{s_t}, \boldsymbol{a_t})$ with basis functions and learn the parameters via neural networks

3. Update belief

$$\boldsymbol{Q^{new}}(\boldsymbol{s_t}, \boldsymbol{a_t}) \leftarrow (1 - \underset{\substack{\uparrow \\ \text{Learning rate}}}{\alpha}) \cdot Q^{old}(s_t, a_t) + \underset{\substack{\uparrow \\ \text{Learning rate}}}{\alpha} \cdot (R_{t+1} + \underset{\substack{\uparrow \\ \text{Discount factor}}}{\gamma} \cdot \max_{a \in A_{s_{t+1}}} (Q^{old}(s_{t+1}, a_{t+1})))$$



0. Initialized the belief of Q

1. Take action based on current belief of Q

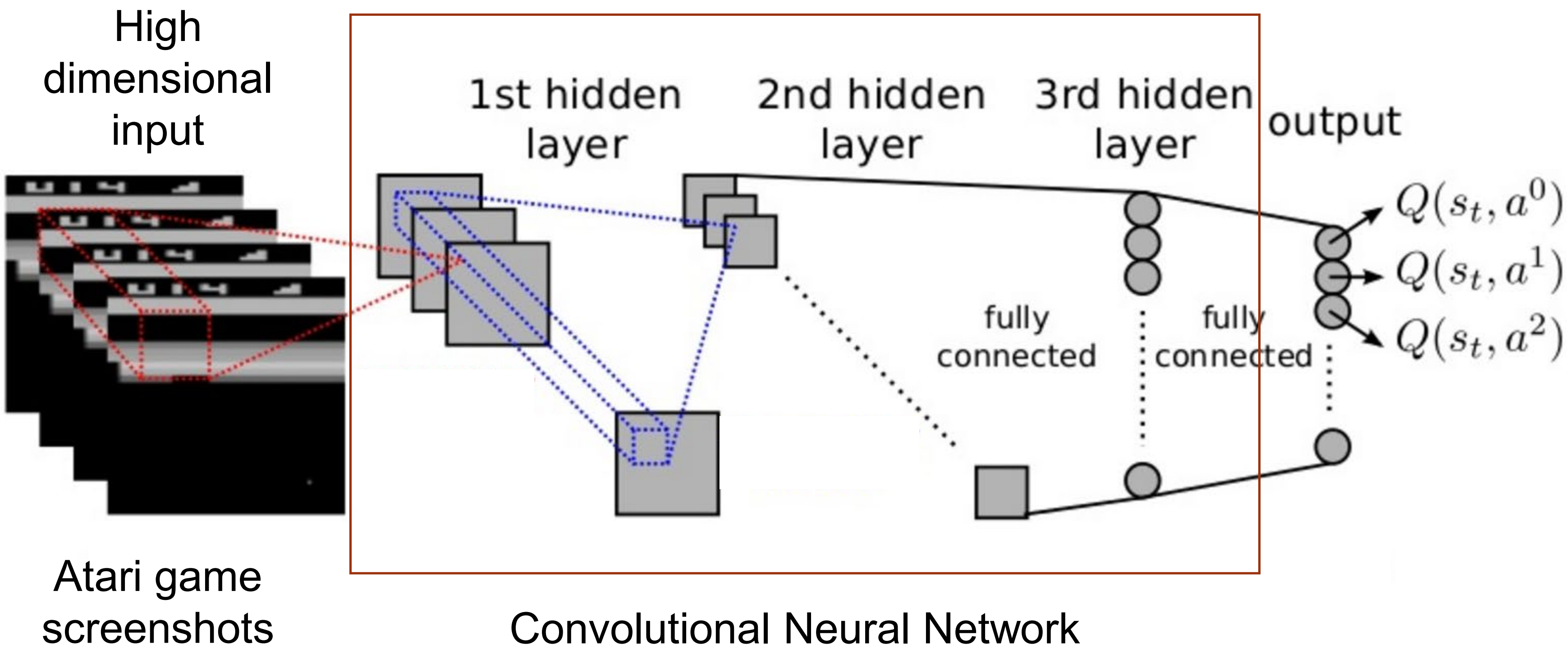
$$\boldsymbol{a_t} = \max_{a \in A_{s_t}} Q(s_t, a_t)$$

2. Feedback

$\boldsymbol{s_{t+1}}, \boldsymbol{R_{t+1}}$

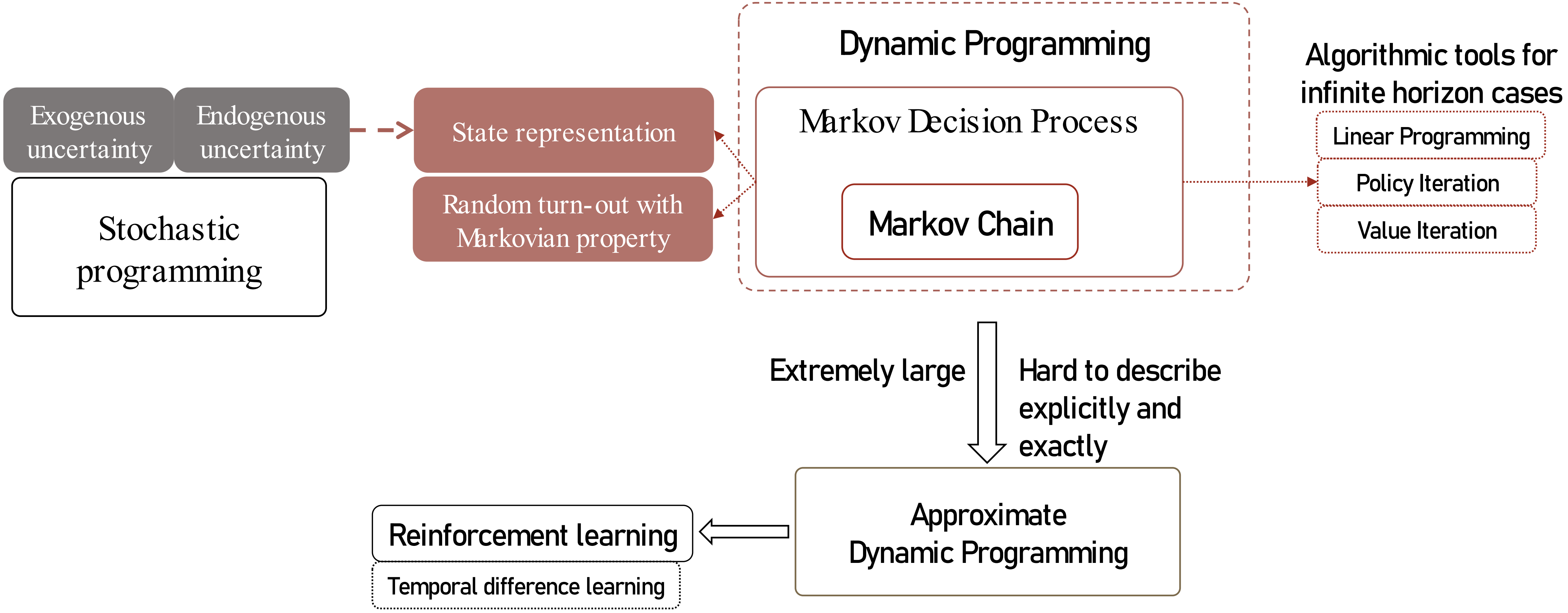
Deep Q Networks

Mnih et al., 2013



Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., & Riedmiller, M. (2013). Playing atari with deep reinforcement learning. arXiv preprint arXiv:1312.5602.

Summary


Exogenous uncertainty
Endogenous uncertainty
Stochastic programming
State representation
Random turn-out with Markovian property
Dynamic Programming
Markov Decision Process
Markov Chain
Algorithmic tools for infinite horizon cases
Linear Programming
Policy Iteration
Value Iteration
Extremely large
Hard to describe explicitly and exactly
Approximate Dynamic Programming
Reinforcement learning
Temporal difference learning

Further extension and recommended resources

- Semi-Markov Decision Process - Continuous Markov Chain
- Partially observed Markov Decision Process - Hidden Markov Chain
- Time-inhomogeneous behaviors

Puterman, Martin L. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

Bertsekas, Dimitri P., et al. Dynamic programming and optimal control. Vol. 1. No. 2. Belmont, MA: Athena scientific, 1995.

Boucherie, Richard J., and Nico M. Van Dijk, eds. *Markov decision processes in practice*. Springer International Publishing, 2017.

Alfa, Attahiru Sule, and Barbara Haas Margolius. "Two classes of time-inhomogeneous Markov chains: Analysis of the periodic case." Annals of Operations Research 160.1 (2008): 121-137.