COMP3702 Tutorial 6

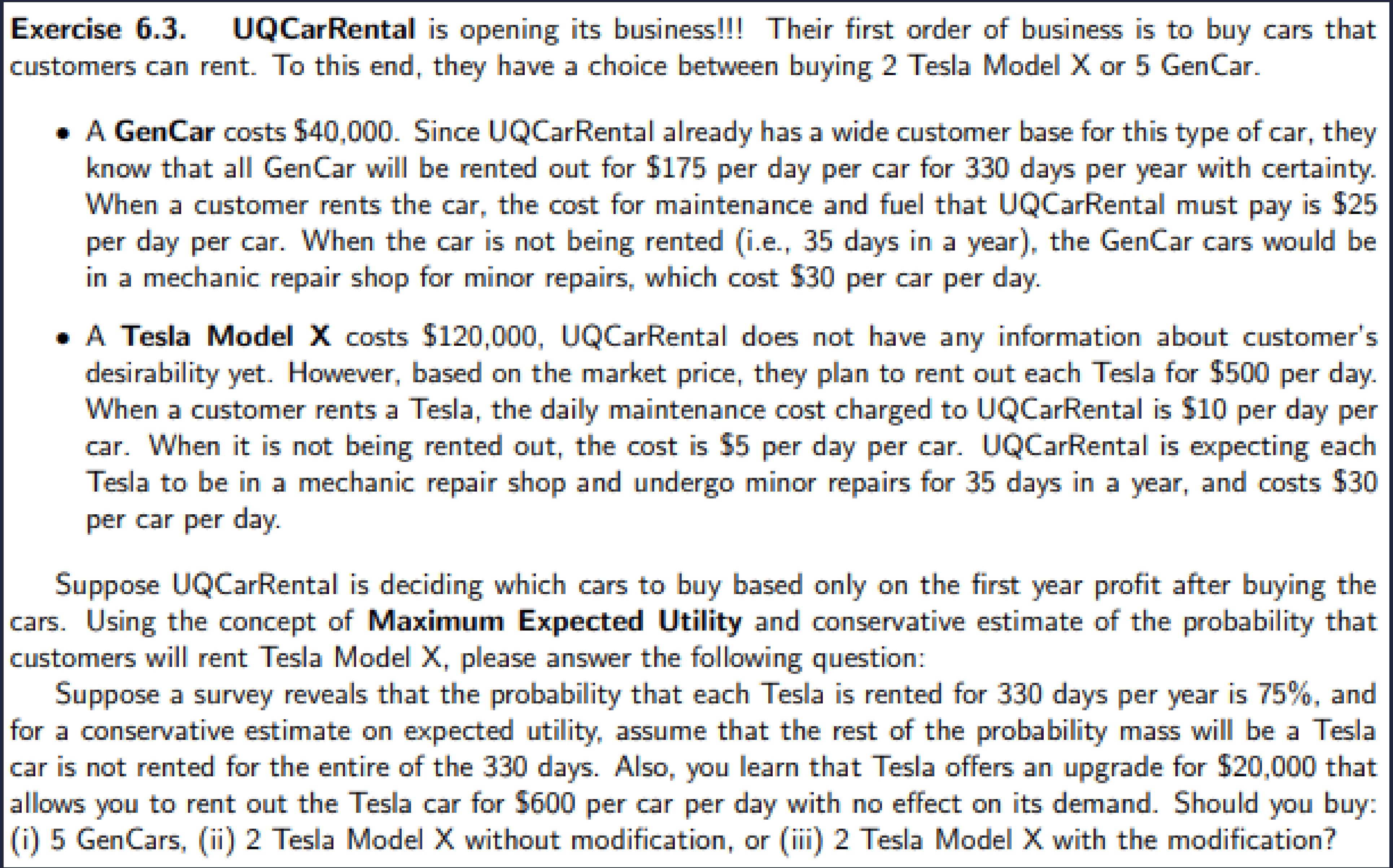
Matt Choy | matthew.choy@uq.edu.au

COMP3702 Tutorial 6

Matt Choy | matthew.choy@uq.edu.au

https://github.com/mattpchoy/comp3702-tutorials

6.3 – UQ Car Rental



Exercise 6.3. **UQCarRental** is opening its business!!! Their first order of business is to buy cars that customers can rent. To this end, they have a choice between buying 2 Tesla Model X or 5 GenCar.

- A **GenCar** costs \$40,000. Since UQCarRental already has a wide customer base for this type of car, they know that all GenCar will be rented out for \$175 per day per car for 330 days per year with certainty. When a customer rents the car, the cost for maintenance and fuel that UQCarRental must pay is \$25 per day per car. When the car is not being rented (i.e., 35 days in a year), the GenCar cars would be in a mechanic repair shop for minor repairs, which cost \$30 per car per day.
- A **Tesla Model X** costs \$120,000, UQCarRental does not have any information about customer's desirability yet. However, based on the market price, they plan to rent out each Tesla for \$500 per day. When a customer rents a Tesla, the daily maintenance cost charged to UQCarRental is \$10 per day per car. When it is not being rented out, the cost is \$5 per day per car. UQCarRental is expecting each Tesla to be in a mechanic repair shop and undergo minor repairs for 35 days in a year, and costs \$30 per car per day.

Suppose UQCarRental is deciding which cars to buy based only on the first year profit after buying the cars. Using the concept of **Maximum Expected Utility** and conservative estimate of the probability that customers will rent Tesla Model X, please answer the following question:

Suppose a survey reveals that the probability that each Tesla is rented for 330 days per year is 75%, and for a conservative estimate on expected utility, assume that the rest of the probability mass will be a Tesla car is not rented for the entire of the 330 days. Also, you learn that Tesla offers an upgrade for \$20,000 that allows you to rent out the Tesla car for \$600 per car per day with no effect on its demand. Should you buy: (i) 5 GenCars, (ii) 2 Tesla Model X without modification, or (iii) 2 Tesla Model X with the modification?

Preferences over Outcomes - Notation

$\succ$ The *is preferred to operator*. $A \succ B$ means that the outcome A is preferred to the outcome B.

$\prec$ The *is preceded by operator*. $A \prec B$ means that the outcome A is preferred less than the outcome B.

$\sim$ The *is indifferent to operator*. $A \sim B$ means that the agent is indifferent to either outcome

Lotteries

- An agent may not know about the outcomes of its actions, but only have a probability distribution of outcomes.
- A **lottery** is a way to describe the probability distribution over an outcome.

$$[p_1 : o_1, p_2 : o_2, \cdots p_k : k]$$

- In a lottery, each of the probabilities p_i should be non-negative, and should sum to 1.
- It specifies that outcome o_i will occur with probability p_i

Axioms of Rational Preferences

1. Completeness / Orderability

Axioms of Rational Preferences

1. Completeness / Orderability

$$(o_1 \succ o_2) \vee (o_1 \prec o_2) \vee (o_1 \sim o_2)$$

An agent must act, and thus must have preferences over their actions

Axioms of Rational Preferences

1. Completeness / Orderability

$$(o_1 \succ o_2) \vee (o_1 \prec o_2) \vee (o_1 \sim o_2)$$

An agent must act, and thus must have preferences over their actions

2. Transitivity

Axioms of Rational Preferences

1. Completeness / Orderability

$$(o_1 \succ o_2) \vee (o_1 \prec o_2) \vee (o_1 \sim o_2)$$

   An agent must act, and thus must have preferences over their actions

2. Transitivity

$$(o_1 \succ o_2) \wedge (o_2 \succ C) \rightarrow (o_1 \succ C)$$

   If an outcome o1 is preferred to o2 and o2 is preferred over C then by transitivity, o1 is preferred to C.

   This is a desirable property, as otherwise, we could have cyclic properties with no solution.

Axioms of Rational Preferences

1. Completeness / Orderability

$$(o_1 \succ o_2) \vee (o_1 \prec o_2) \vee (o_1 \sim o_2)$$

   An agent must act, and thus must have preferences over their actions

2. Transitivity

$$(o_1 \succ o_2) \wedge (o_2 \succ C) \rightarrow (o_1 \succ C)$$

   If an outcome o1 is preferred to o2 and o2 is preferred over C then by transitivity, o1 is preferred to C.

   This is a desirable property, as otherwise, we could have cyclic properties with no solution.

3. Monotonicity

Axioms of Rational Preferences

1. Completeness / Orderability

$$(o_1 \succ o_2) \vee (o_1 \prec o_2) \vee (o_1 \sim o_2)$$

   An agent must act, and thus must have preferences over their actions

2. Transitivity

$$(o_1 \succ o_2) \wedge (o_2 \succ C) \rightarrow (o_1 \succ C)$$

   If an outcome o1 is preferred to o2 and o2 is preferred over C then by transitivity, o1 is preferred to C.

   This is a desirable property, as otherwise, we could have cyclic properties with no solution.

3. Monotonicity

$$o_1 \succ o_2 \rightarrow (p \geq q \Leftrightarrow [p : o_1, 1 - p : o_2] \succ [q : o_1, 1 - q : o_2])$$

   Agent prefers a larger chance of getting a better outcome than a smaller chance

Axioms of Rational Preferences

1. Completeness / Orderability

$$(o_1 \succ o_2) \vee (o_1 \prec o_2) \vee (o_1 \sim o_2)$$

An agent must act, and thus must have preferences over their actions

2. Transitivity

$$(o_1 \succ o_2) \wedge (o_2 \succ C) \rightarrow (o_1 \succ C)$$

If an outcome o1 is preferred to o2 and o2 is preferred over C then by transitivity, o1 is preferred to C.

This is a desirable property, as otherwise, we could have cyclic properties with no solution.

3. Monotonicity

$$o_1 \succ o_2 \rightarrow (p \geq q \Leftrightarrow [p : o_1, 1 - p: o_2] \succ [q : o_1, 1 - q : o_2])$$

Agent prefers a larger chance of getting a better outcome than a smaller chance

4. Continuity

Axioms of Rational Preferences

1. Completeness / Orderability

$$(o_1 \succ o_2) \vee (o_1 \prec o_2) \vee (o_1 \sim o_2)$$

   An agent must act, and thus must have preferences over their actions

2. Transitivity

$$(o_1 \succ o_2) \wedge (o_2 \succ C) \rightarrow (o_1 \succ C)$$

   If an outcome o1 is preferred to o2 and o2 is preferred over C then by transitivity, o1 is preferred to C.

   This is a desirable property, as otherwise, we could have cyclic properties with no solution.

3. Monotonicity

$$o_1 \succ o_2 \rightarrow (p \geq q \Leftrightarrow [p : o_1, 1 - p: o_2] \succ [q : o_1, 1 - q : o_2])$$

   Agent prefers a larger chance of getting a better outcome than a smaller chance

4. Continuity

$$o_1 \succ o_2 \succ C \Rightarrow \exists p \in [0,1][p : o_1, 1 - p : C] \sim o_2$$

Axioms of Rational Preferences

5. Substitutability

Axioms of Rational Preferences

5. Substitutability

$$o_1 \sim o_2$$
$$\Rightarrow [p : o_1, 1-p : C] \sim [p : o_2, 1-p : C]$$

If $o_1 \sim o_2$ then the agent is indifferent between the lotteries that differ by only o1 and o2.

Axioms of Rational Preferences

5. Substitutability

$$o_1 \sim o_2$$
$$\Rightarrow [p : o_1, 1-p : C] \sim [p : o_2, 1-p : C]$$

If $o_1 \sim o_2$ then the agent is indifferent between the lotteries that differ by only o1 and o2.

6. Decomposability

$$[p : o_1, 1-p : [q : o_2, 1-q : o_3]] \sim [p : o_1, (1-p)q : o_2, (1-p)(1-q) : o_3]$$

An agent is indifferent between the same lotteries and the same outcomes.

Really just looking at the final probabilities and possible outcomes in the problem, regardless of how they are structured.

Utility

If we assume that the agent is rational, and observes the rules above, then preferences can be measured by a function of utility.

$$utility : outcomes \rightarrow [0, 1]$$

Such that:

$o_1 \succ o_2$ if and only of $utility(o_1) \geq utility(o_2)$

$o_1 \sim o_2$ if and only if $utility(o_1) = utility(o_2)$

Utility

If we assume that the agent is rational, and observes the rules above, then preferences can be measured by a function of utility.

$$utility : outcomes \rightarrow [0,1]$$

Such that:

$o_1 \succ o_2$ if and only of $utility(o_1) \geq utility(o_2)$

$o_1 \sim o_2$ if and only if $utility(o_1) = utility(o_2)$

We also have the property that utilities are linear with probabilities

$$utility([p_1 : o_1, p_2 : o_2, \cdots, p_k : o_k]) = \sum_{i=1}^{k} (p_i \times utility(o_i))$$

Therefore, we can replace preferences with numbers!

Probabilities and Preferences

$$\$1{,}000{,}000 \qquad or \qquad [0.5 : 0, 0.5 : \$2{,}000{,}000]$$

Probabilities and Preferences

$$\$1{,}000{,}000 \qquad or \qquad [0.5 : 0, 0.5 : \$2{,}000{,}000]$$

$$utility([1.0 : \$1{,}000{,}000]) = \$1{,}000{,}000$$

$$\begin{aligned} utility([0.5 : 0, 0.5 : \$2{,}000{,}000]) &= 0.5 \times \$0 + 0.5 \times \$1{,}000{,}000 \\ &= 0 + \$1{,}000{,}000 = \$1{,}000{,}000 \end{aligned}$$

MEU Example – Car Rental

Suppose our goal is to buy a car, to sell it for profit. We know the following facts:

- A car costs $1,000 and we can sell them for $1,100.
- It costs $40 to repair a good car
- It costs $200 to repair a bad car
- 20% of cars are bad

Determine the best possible outcome using the concept of MEU

MEU Example – Car Rental

Suppose our goal is to buy a car, to sell it for profit. We know the following facts:

- A car costs $1,000 and we can sell them for $1,100. $100 profit from each sale.
- It costs $40 to repair a good car
- It costs $200 to repair a bad car
- 20% of cars are bad

MEU Example – Car Rental

Suppose our goal is to buy a car, to sell it for profit. We know the following facts:

- A car costs $1,000 and we can sell them for $1,100. $100 profit from each sale.
- It costs $40 to repair a good car
- It costs $200 to repair a bad car
- 20% of cars are bad

We know that the possible states are [Good Car, Bad Car] and that we preference having a good car.

MEU Example – Car Rental

Suppose our goal is to buy a car, to sell it for profit. We know the following facts:

- A car costs \$1,000 and we can sell them for \$1,100. \$100 profit from each sale.
- It costs \$40 to repair a good car
- It costs \$200 to repair a bad car
- 20% of cars are bad

State Space: $\{Good\ Car, Bad\ Car\}$

Preference: $Good\ Car \succ Bad\ Car$

MEU Example – Car Rental

Suppose our goal is to buy a car, to sell it for profit. We know the following facts:

- A car costs \$1,000 and we can sell them for \$1,100. \$100 profit from each sale.
- It costs \$40 to repair a good car
- It costs \$200 to repair a bad car
- 20% of cars are bad

State Space: $\{Good\ Car, Bad\ Car\}$

Preference: $Good\ Car \succ Bad\ Car$

Utility Function: $U(Good\ Car) = 1100 - 1000 - 40 = 60$

$U(Bad\ Car) = 1100 - 1000 - 200 = -100$

MEU Example – Car Rental

Suppose our goal is to buy a car, to sell it for profit. We know the following facts:

- A car costs \$1,000 and we can sell them for \$1,100. \$100 profit from each sale.
- It costs \$40 to repair a good car
- It costs \$200 to repair a bad car
- 20% of cars are bad

State Space: $\{Good\ Car, Bad\ Car\}$

Preference: $Good\ Car \succ Bad\ Car$

Utility Function: $U(Good\ Car) = 1100 - 1000 - 40 = 60$

$U(Bad\ Car) = 1100 - 1000 - 200 = -100$

Lottery: $[0.8 : Good\ Car, 0.2 : Bad\ Car]$

MEU Example – Car Rental

Suppose our goal is to buy a car, to sell it for profit. We know the following facts:

- A car costs \$1,000 and we can sell them for \$1,100. \$100 profit from each sale.
- It costs \$40 to repair a good car
- It costs \$200 to repair a bad car
- 20% of cars are bad

State Space: $\{Good\ Car, Bad\ Car\}$

Preference: $Good\ Car \succ Bad\ Car$

Utility Function: $U(Good\ Car) = 1100\ - 1000\ - 40 = 60$

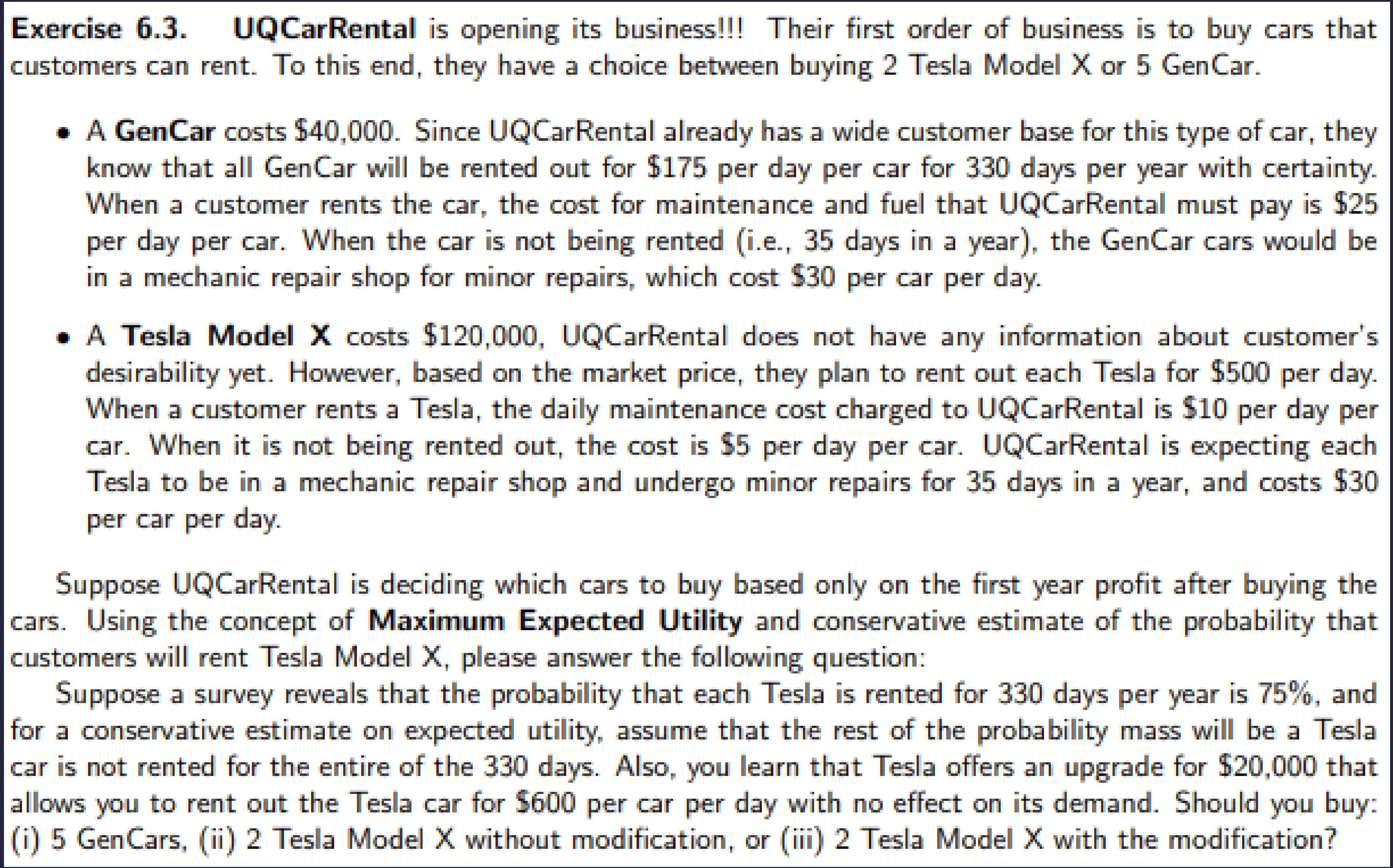
$U(Bad\ Car)\ = 1100\ - 1000\ - 200 = -100$

Lottery: $[0.8 : Good\ Car, 0.2 : Bad\ Car]$

MEU: $P(Good\ Car) \times U(Good\ Car) + P(Bad\ Car) \times U(Bad\ Car)$

$= 0.8 \times 60 + 0.2 \times 100 = 28$

6.3 – UQ Car Rental



Exercise 6.3. **UQCarRental** is opening its business!!! Their first order of business is to buy cars that customers can rent. To this end, they have a choice between buying 2 Tesla Model X or 5 GenCar.

- A **GenCar** costs \$40,000. Since UQCarRental already has a wide customer base for this type of car, they know that all GenCar will be rented out for \$175 per day per car for 330 days per year with certainty. When a customer rents the car, the cost for maintenance and fuel that UQCarRental must pay is \$25 per day per car. When the car is not being rented (i.e., 35 days in a year), the GenCar cars would be in a mechanic repair shop for minor repairs, which cost \$30 per car per day.
- A **Tesla Model X** costs \$120,000, UQCarRental does not have any information about customer's desirability yet. However, based on the market price, they plan to rent out each Tesla for \$500 per day. When a customer rents a Tesla, the daily maintenance cost charged to UQCarRental is \$10 per day per car. When it is not being rented out, the cost is \$5 per day per car. UQCarRental is expecting each Tesla to be in a mechanic repair shop and undergo minor repairs for 35 days in a year, and costs \$30 per car per day.

Suppose UQCarRental is deciding which cars to buy based only on the first year profit after buying the cars. Using the concept of **Maximum Expected Utility** and conservative estimate of the probability that customers will rent Tesla Model X, please answer the following question:

Suppose a survey reveals that the probability that each Tesla is rented for 330 days per year is 75%, and for a conservative estimate on expected utility, assume that the rest of the probability mass will be a Tesla car is not rented for the entire of the 330 days. Also, you learn that Tesla offers an upgrade for \$20,000 that allows you to rent out the Tesla car for \$600 per car per day with no effect on its demand. Should you buy: (i) 5 GenCars, (ii) 2 Tesla Model X without modification, or (iii) 2 Tesla Model X with the modification?

Find the maximum expected utility.

Found by the sum of probability and utility for each possible outcome.

Exercise 6.3

We want to choose the most rational (i.e. highest expected utility) option out of the options presented.

We first compute the expected utility for the 5 GenCars scenario. We know that:

- GenCar costs $40,000
- Rent for $175 per day, with 330 days of certainty.
  - When a customer rents the car, running cost is $25 per day per car
- When not rented, cost $30 per day.

Exercise 6.3

We want to choose the most rational (i.e. highest expected utility) option out of the options presented.

We first compute the expected utility for the 5 GenCars scenario. We know that:

$$\mathbb{E}(5\ GenCar) = 5 \times (330 \times (\$175\ \ - \$25) + 35 \times (-\$30) - \$40{,}000)$$
$$= 5 \times \$8,450 = \$42{,}250$$

- GenCar costs $40,000
- Rent for $175 per day, with 330 days of certainty.
  - When a customer rents the car, running cost is $25 per day per car
- When not rented, cost $30 per day.

Exercise 6.3

We want to choose the most rational (i.e. highest expected utility) option out of the options presented.

We next compute the expected utility of purchasing two Teslas

- Tesla Model X costs $120,000
- Plan to rent out for $500 per day
  - Maintenance cost of $10 per day
- Maintenance cost of $5 per day, when not rented out (35 days)

Exercise 6.3

We want to choose the most rational (i.e. highest expected utility) option out of the options presented.

We next compute the expected utility of purchasing two Teslas

- Tesla Model X costs $120,000
- Plan to rent out for $500 per day -> (330 days)
    - Maintenance cost of $10 per day
- Maintenance cost of $5 per day, when not rented out (35 days)
- Survey says that Tesla will be rented out 75% of the 330 days per year.

Exercise 6.3

We want to choose the most rational (i.e. highest expected utility) option out of the options presented.

We next compute the expected utility of purchasing two Teslas

$$\begin{aligned}\mathbb{E}(2\ Tesla) &= 2 \times (0.75 \times 330 \times (\$500 - \$10) + 0.25 \times 330 \times (-\$5) + 35 \times (\$30 + \$5)) \\ &= \$60{,}712.5 - \$61.473.5 = -\$725\end{aligned}$$

- Tesla Model X costs $120,000
- Plan to rent out for $500 per day -> (330 days)
  - Maintenance cost of $10 per day
- Maintenance cost of $5 per day, when not rented out (35 days)
- Survey says that Tesla will be rented out 75% of the 330 days per year.

Exercise 6.3

We want to choose the most rational (i.e. highest expected utility) option out of the options presented.

Finally, we compute the cost of the upgraded Tesla Model X

$$
\begin{aligned}
U(2\,Tesla\ Upgraded) &= (-725 + 0.75 \times (330 \times \$100 - \$20{,}000) + \\
&\quad +0.25 \times \big(330 \times (-\$5) - 35 \times (\$330 + \$5)\big) - \$12{,}000) \\
&= -725 + \$9{,}500 = \$8770
\end{aligned}
$$

- Tesla Model X costs \$120,000
- Plan to rent out for \$500 per day -> (330 days)
    - Maintenance cost of \$10 per day
- Maintenance cost of \$5 per day, when not rented out (35 days)
- Survey says that Tesla will be rented out 75% of the 330 days per year.
- Upgrade costs \$20,000 but will allow the Teslas to be rented out for \$600 per day.

Exercise 6.3

We want to choose the most rational (i.e. highest expected utility) option out of the options presented.

$$U(5\ GenCars) = \$42{,}250$$

$$U(2\ Teslas) = \$-725$$

$$U(2\ Tesla\ Upgraded) = \$8770$$

Markov Decision Processes

Markov Decision Processes

Markov Decision Process is a framework we can use to determine the best action that an agent should perform in a environment that is:

- Stochastic (non-deterministic)
- Discrete-time
- Discrete-transition (well defined transitions between states)

MDPs are the basis for Reinforcement Learning, and can be used to solve games like Tic Tac Toe, Chess, Go and other games in which we consider the other player.

MDP Components

A set of states (S)

A set of actions (A)

Transition Function $P(S_{t+1} | S_t, A_t)$

Reward Function $R(S_t, A_t, S_{t+1})$

Discount Factor γ

MDP Components

A set of states (S)

A set of actions (A)

Transition Function $P(S_{t+1}\,|S_t, A_t)$ - Changed definition

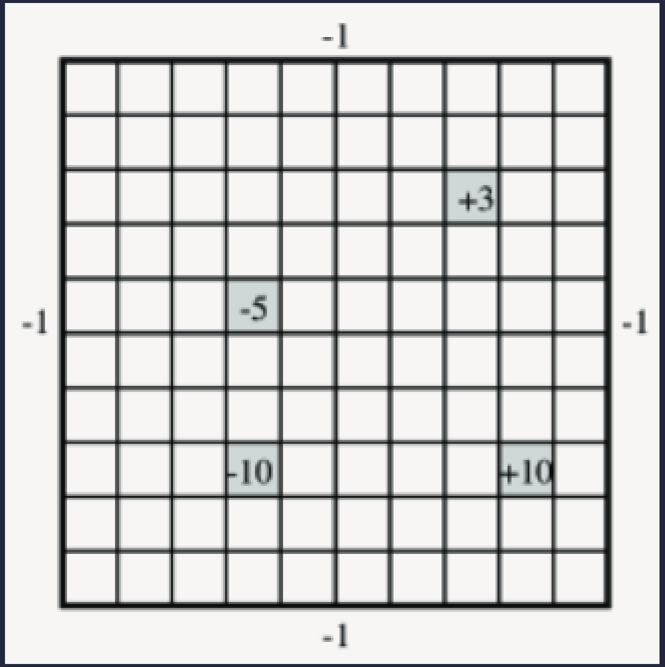
Reward Function $R(S_t, A_t, S_{t+1})$ - Reward at some time instant t

Discount Factor γ

MDP Example

Consider a simple, non-deterministic Grid world, with the following rules:

- The agent can be in any grid cell
- In certain cells, the agent is given positive or negative rewards.
- If you collide with the walls, there is a reward of -1.



MDP Example

Consider a simple, non-deterministic Grid world, with the following rules:

- The agent can be in any grid cell
- In certain cells, the agent is given positive or negative rewards.
- If you collide with the walls, there is a reward of -1.

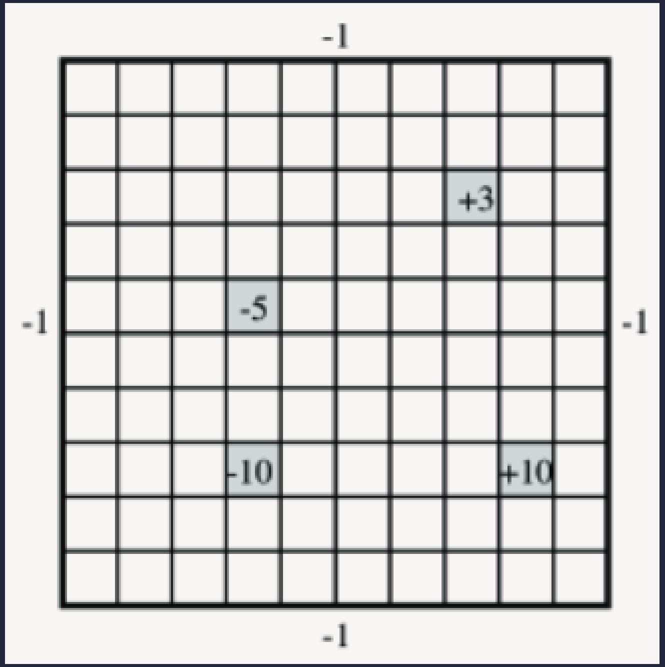
States: 100 states, corresponding to the (r, c) of the robot.

Actions: Move up, down, left, right

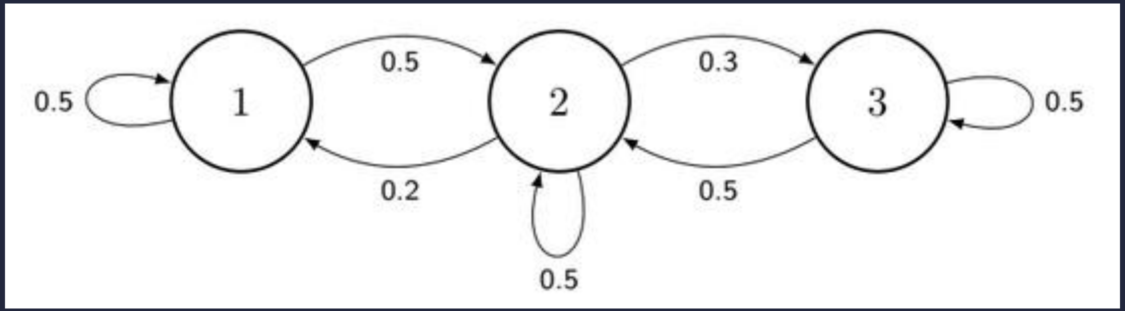
Transition: Robot moves in the commanded direction with probability 0.7

One of the other directions with probability 0.1

Rewards: If the agent crashes into an outside wall, it remains in its current position, and has a reward of -1. For special reward states, the agent gets the reward when leaving the state.



Markov Chains



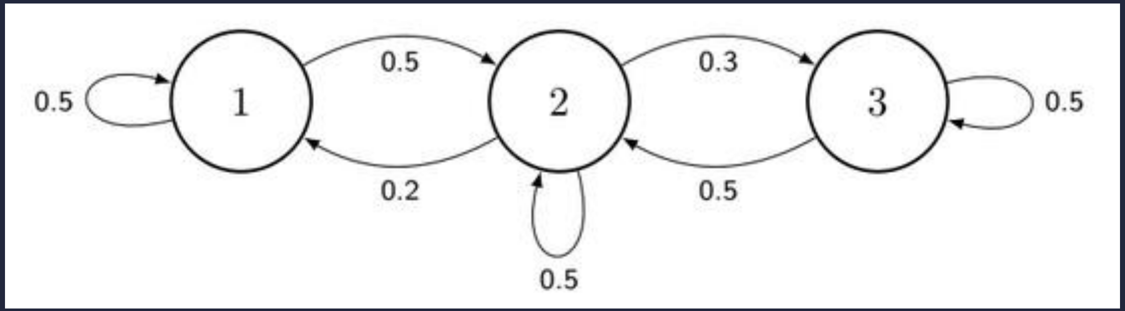
A Markov chain is a system with:

- Discrete time steps
- Discrete states
- Predictable, stochastic state transitions underlying probability distributions that govern how the agent transitions from one state to another.

- Conventional notation for Markov Chains describe that states are circles, and transitions are directed edges leaving states.
- The weights of those edges are the probabilities of the transition occurring.

- No decisions are made, purely probabilistic chances of ending up in another state.
- Transitions only depend on the current states, and not the full history
  (Markovian Property; Memoryless)

Markov Chains

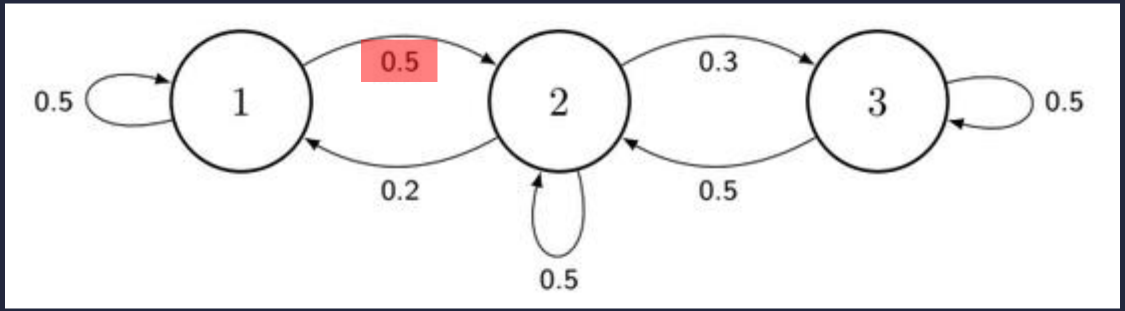


We can describe the state transition probabilities as matrices – we call this the state transition probability matrix, denoted P

- The size of the state transition matrix is |S| x |S|, where |S| is the number of states in the state space.

$$\begin{array}{cccc} & & \textbf{Destination} & \\ & \mathbf{0.5} & \mathbf{0.5} & \mathbf{0} \\ \textbf{Source} & \mathbf{0.2} & \mathbf{0.5} & \mathbf{0.3} \\ & \mathbf{0} & \mathbf{0.5} & \mathbf{0.5} \end{array}$$

Markov Chains

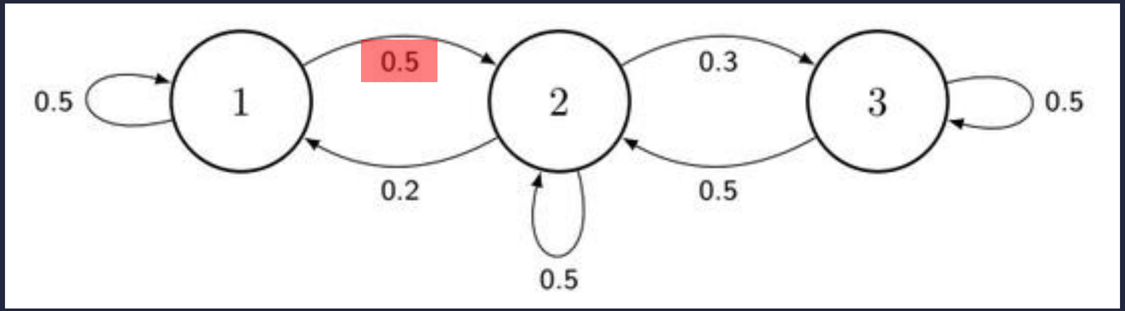


We can describe the state transition probabilities as matrices – we call this the state transition probability matrix, denoted P

- The size of the state transition matrix is |S| x |S|, where |S| is the number of states in the state space.

$$\begin{array}{cccc} & & \textbf{Destination} & \\ \textbf{Source} & 0.5 & 0.5 & 0 \\ & 0.2 & 0.5 & 0.3 \\ & 0 & 0.5 & 0.5 \end{array}$$

Markov Chains

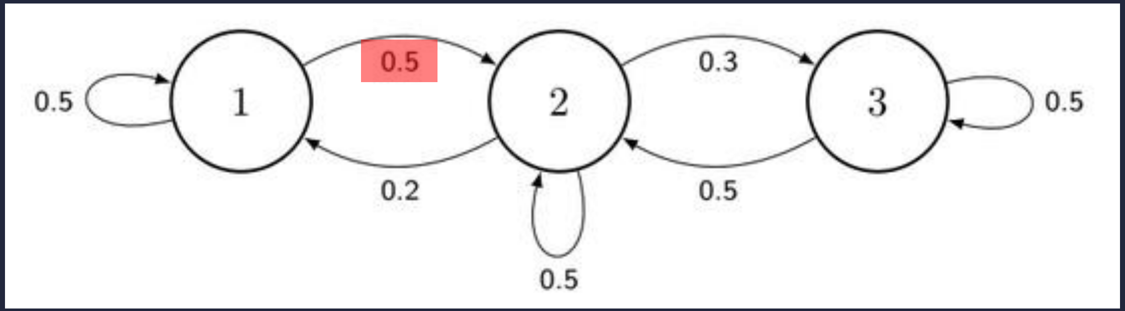


We can describe the state transition probabilities as matrices – we call this the state transition probability matrix, denoted P

- The size of the state transition matrix is |S| x |S|, where |S| is the number of states in the state space.
- The state vector of a given state is a one-hot vector, containing all zeros, except for a single 1 – this is the important piece of data in the vector.
  - For example, the state vector for state 1 in this example is $x_1 = [1,0,0]$

$$\begin{array}{c|ccc} & & \textbf{Destination} & \\ \hline \textbf{Source} & \textbf{0.5} & \textbf{0.5} & \textbf{0} \\ & \textbf{0.2} & \textbf{0.5} & \textbf{0.3} \\ & \textbf{0} & \textbf{0.5} & \textbf{0.5} \end{array}$$

Markov Chains



We can describe the state transition probabilities as matrices – we call this the state transition probability matrix, denoted P

- The size of the state transition matrix is |S| x |S|, where |S| is the number of states in the state space.
- The state vector of a given state is a one-hot vector, containing all zeros, except for a single 1 – this is the important piece of data in the vector.
  - For example, the state vector for state 1 in this example is $x_1 = [1,0,0]$

$$\begin{array}{c|ccc} & & \textbf{Destination} & \\ \textbf{Source} & 0.5 & 0.5 & 0 \\ & 0.2 & 0.5 & 0.3 \\ & 0 & 0.5 & 0.5 \end{array}$$

- The distribution of states is given by the following formula:

$$x_k = x_0 P^k$$

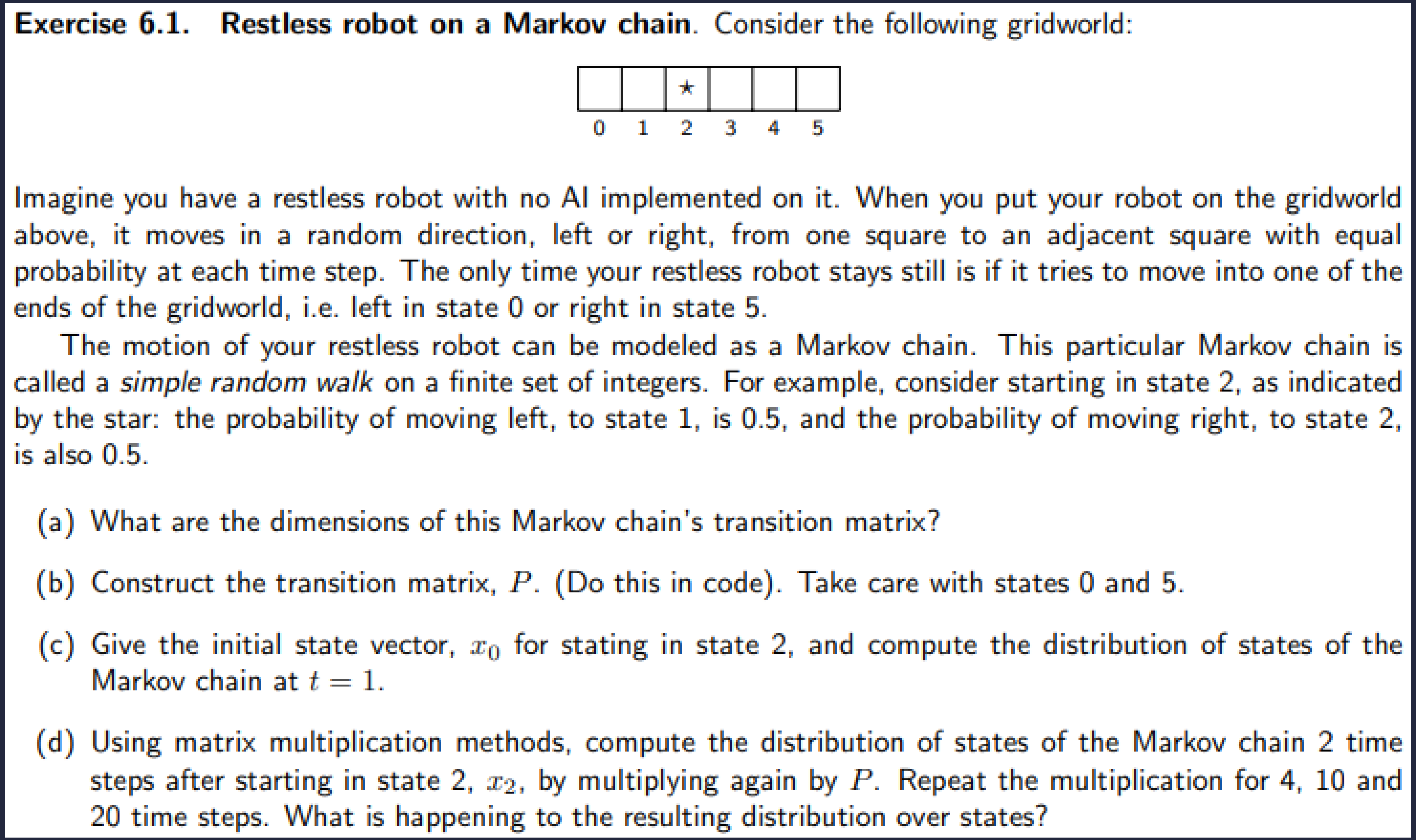
where k is the current time step

x_0 is the state vector

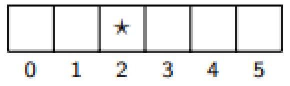
P is the state transition probability matrix

- In short, x_k can be found by performing matrix multiplication

Exercise 6.1



Exercise 6.1. Restless robot on a Markov chain. Consider the following gridworld:



Imagine you have a restless robot with no AI implemented on it. When you put your robot on the gridworld above, it moves in a random direction, left or right, from one square to an adjacent square with equal probability at each time step. The only time your restless robot stays still is if it tries to move into one of the ends of the gridworld, i.e. left in state 0 or right in state 5.

The motion of your restless robot can be modeled as a Markov chain. This particular Markov chain is called a *simple random walk* on a finite set of integers. For example, consider starting in state 2, as indicated by the star: the probability of moving left, to state 1, is 0.5, and the probability of moving right, to state 2, is also 0.5.

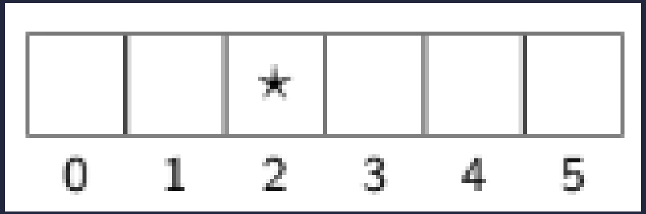
(a) What are the dimensions of this Markov chain's transition matrix?

(b) Construct the transition matrix, P. (Do this in code). Take care with states 0 and 5.

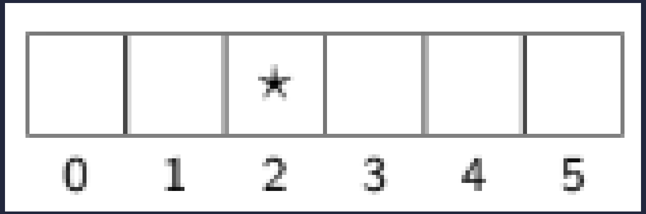
(c) Give the initial state vector, x_0 for stating in state 2, and compute the distribution of states of the Markov chain at $t = 1$.

(d) Using matrix multiplication methods, compute the distribution of states of the Markov chain 2 time steps after starting in state 2, x_2, by multiplying again by P. Repeat the multiplication for 4, 10 and 20 time steps. What is happening to the resulting distribution over states?

Exercise 6.1

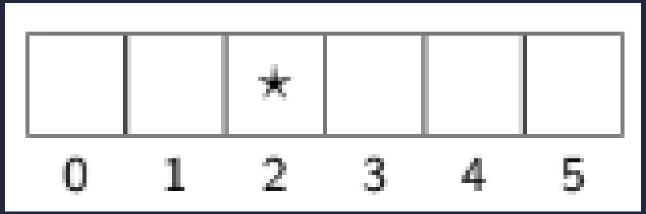


Exercise 6.1a



The transition matrix is of size |S| x |S|, where |S| is the size of the state space (i.e. the number of states in the problem).

Exercise 6.1a

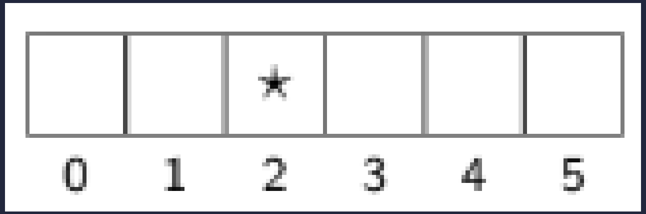


The transition matrix is of size |S| x |S|, where |S| is the size of the state space (i.e. the number of states in the problem).

We have that:

$$s \in \{0,1,2,3,4,5\}$$
$$s' \in \{0,1,2,3,4,5\}$$

Exercise 6.1b



The transition matrix is of size |S| x |S|, where |S| is the size of the state space (i.e. the number of states in the problem).

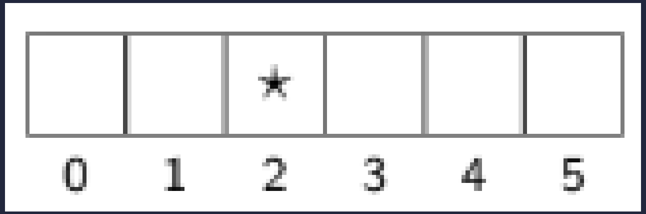
We have that:

$$s \in \{0,1,2,3,4,5\}$$
$$s' \in \{0,1,2,3,4,5\}$$

Since we have |S|= 6, our transition matrix has size 6x6

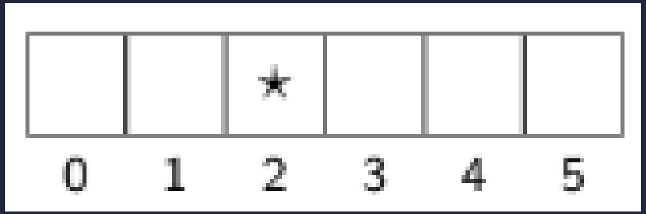
```
p = np.array([
    [0.5, 0.5, 0.0, 0.0, 0.0, 0.0],
    [0.5, 0.0, 0.5, 0.0, 0.0, 0.0],
    [0.0, 0.5, 0.0, 0.5, 0.0, 0.0],
    [0.0, 0.0, 0.5, 0.0, 0.5, 0.0],
    [0.0, 0.0, 0.0, 0.5, 0.0, 0.5],
    [0.0, 0.0, 0.0, 0.0, 0.5, 0.5]
])
```

Exercise 6.1c



The initial state vector is a one-hot vector representing the current state.

Exercise 6.1c

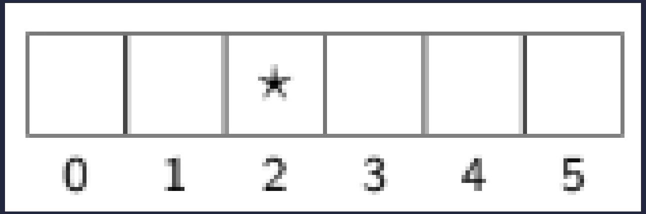


The initial state vector is a one-hot vector representing the current state.

We know that the initial state is state 2, and thus, the state distribution vector is:

$$x_0 = [0, 0, 1, 0, 0, 0]$$

Exercise 6.1c



The initial state vector is a one-hot vector representing the current state.

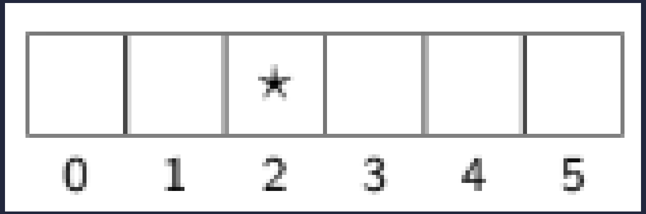
We know that the initial state is state 2, and thus, the state distribution vector is:

$$x_0 = [0, 0, 1, 0, 0, 0]$$

We can represent this in code as follows:

```
x0 = np.array([0, 0, 1, 0, 0, 0])
```

Exercise 6.1c



The initial state vector is a one-hot vector representing the current state.

We know that the initial state is state 2, and thus, the state distribution vector is:

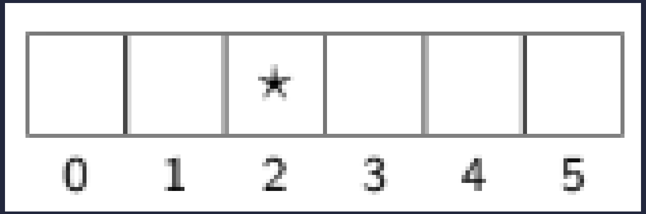
$$x_0 = [0, 0, 1, 0, 0, 0]$$

We can represent this in code as follows:

```
x0 = np.array([0, 0, 1, 0, 0, 0])
```

We next want to find the state distribution at $t = 1$

Exercise 6.1c



The initial state vector is a one-hot vector representing the current state.

We know that the initial state is state 2, and thus, the state distribution vector is:

$$x_0 = [0, 0, 1, 0, 0, 0]$$

We can represent this in code as follows:

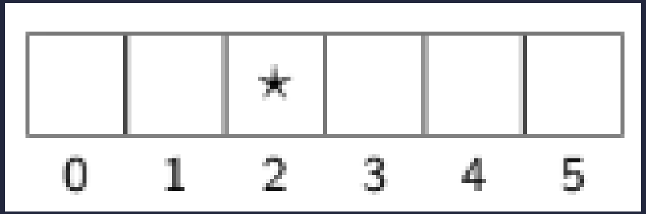
```
x0 = np.array([0, 0, 1, 0, 0, 0])
```

We next want to find the state distribution at $t = 1$

That is, we want to compute $x_0 \cdot P^1$. We can do this in code as follows

```
x1 = np.matmul(x0, p)
[0.0, 0.5, 0.0, 0.5, 0.0, 0.0]
```

Exercise 6.1d

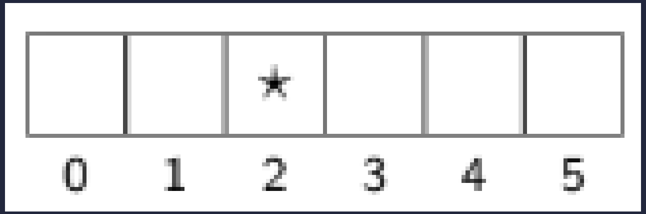


We can compute the state distribution for other time instances, using a modified version of the syntax before.

Remember that the state distribution is given by the formula:

$$x_t = x_0 P^t$$

Exercise 6.1d



We can compute the state distribution for other time instances, using a modified version of the syntax before.

Remember that the state distribution is given by the formula:

$$x_t = x_0 P^t$$

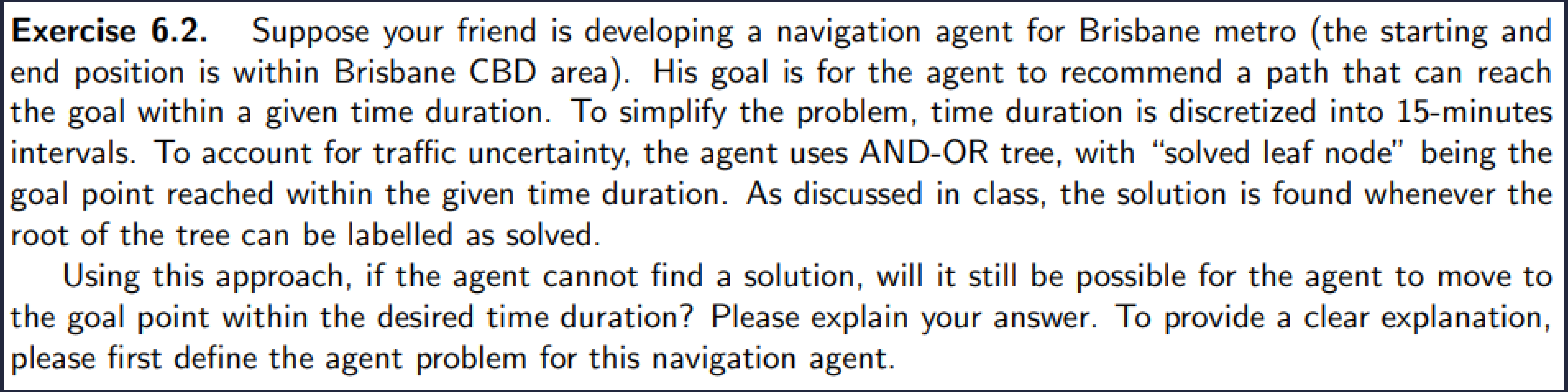
```
x2 = np.matmul(x0, np.linalg.matrix_power(p, 2))
[0.25, 0.00, 0.50, 0.00, 0.25, 0.00]

x4 = np.matmul(x0, np.linalg.matrix_power(p, 4))
[0.2500, 0.0625, 0.3750, 0.00, 0.250, 0.0625]

x10 = np.matmul(x0, np.linalg.matrix_power(p, 10))
[0.2060, 0.1269, 0.2460, 0.878, 0.2060, 0.1269]

x20 = np.matmul(x0, np.linalg.matrix_power(p, 20))
[0.1760, 0.1572, 0.1854, 0.1478, 0.1760, 0.1572]
```

Exercise 6.2



Exercise 6.2. Suppose your friend is developing a navigation agent for Brisbane metro (the starting and end position is within Brisbane CBD area). His goal is for the agent to recommend a path that can reach the goal within a given time duration. To simplify the problem, time duration is discretized into 15-minutes intervals. To account for traffic uncertainty, the agent uses AND-OR tree, with "solved leaf node" being the goal point reached within the given time duration. As discussed in class, the solution is found whenever the root of the tree can be labelled as solved.

Using this approach, if the agent cannot find a solution, will it still be possible for the agent to move to the goal point within the desired time duration? Please explain your answer. To provide a clear explanation, please first define the agent problem for this navigation agent.

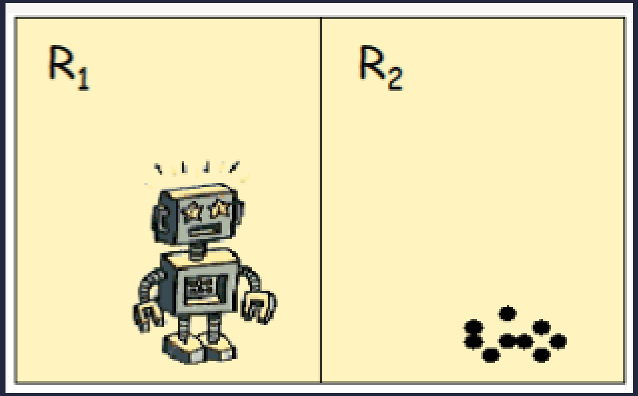
AND / OR Trees

AND / OR Trees

- Alternating (interleaved) AND and OR levels.
- At each node of a OR level, branching is introduced by the agent’s own choice
- At each node of an AND level, branching is introduced by the environment.

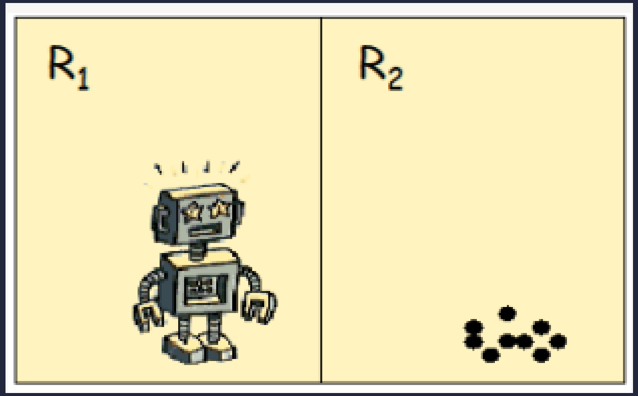
Slippery Vacuum Robot

- The state of the robot can be described as follows:
  - The robot’s position is either {in R1, in R2}
  - R1 = {clean, dirty}
  - R2 = {clean, dirty}

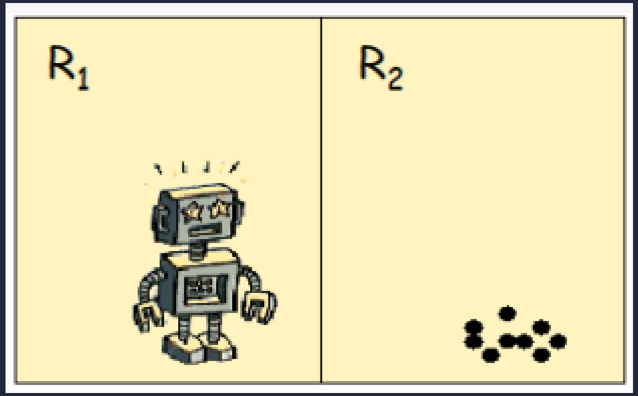


Slippery Vacuum Robot

- The state of the robot can be described as follows:
  - The robot’s position is either {in R1, in R2}
  - R1 = {clean, dirty}
  - R2 = {clean, dirty}
- The robot can perform the following actions:
  - {Left, Right, Suck(R1), Suck(R2)

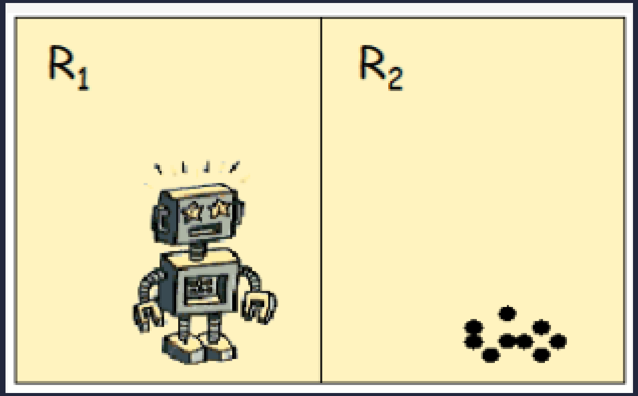


Slippery Vacuum Robot



- The state of the robot can be described as follows:
  - The robot’s position is either {in R1, in R2}
  - R1 = {clean, dirty}
  - R2 = {clean, dirty}
- The robot can perform the following actions:
  - {Left, Right, Suck(R1), Suck(R2)
- The world is non-deterministic – the robot will sometimes perform the desired movement action

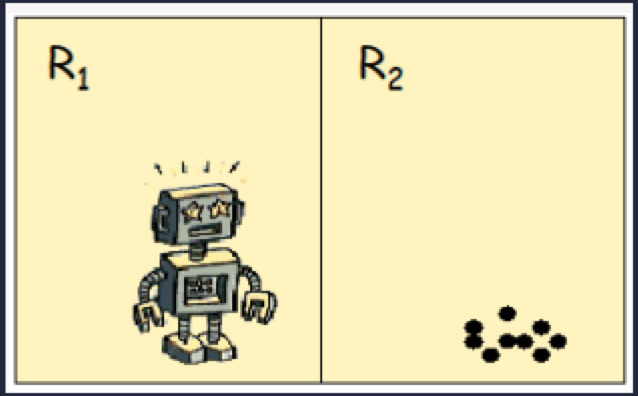
Slippery Vacuum Robot



- The state of the robot can be described as follows:
  - The robot's position is either {in R1, in R2}
  - R1 = {clean, dirty}
  - R2 = {clean, dirty}
- The robot can perform the following actions:
  - {Left, Right, Suck(R1), Suck(R2)}
- The world is non-deterministic – the robot will sometimes perform the desired movement action
- The initial state of the robot can be described as follows:

$$(Robot\ in\ R_1) \land (R_1\ is\ clean) \land (R_2\ is\ dirty)$$

Slippery Vacuum Robot



- The state of the robot can be described as follows:
  - The robot's position is either {in R1, in R2}
  - R1 = {clean, dirty}
  - R2 = {clean, dirty}
- The robot can perform the following actions:
  - {Left, Right, Suck(R1), Suck(R2)}
- The world is non-deterministic – the robot will sometimes perform the desired movement action
- The initial state of the robot can be described as follows:

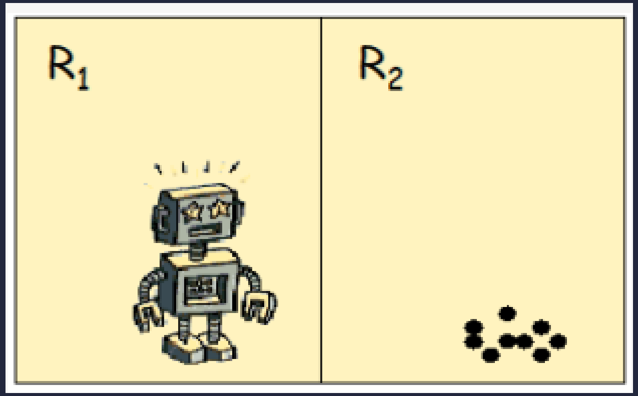
$$(Robot\ in\ R_1) \wedge (R_1\ is\ clean) \wedge (R_2\ is\ dirty)$$

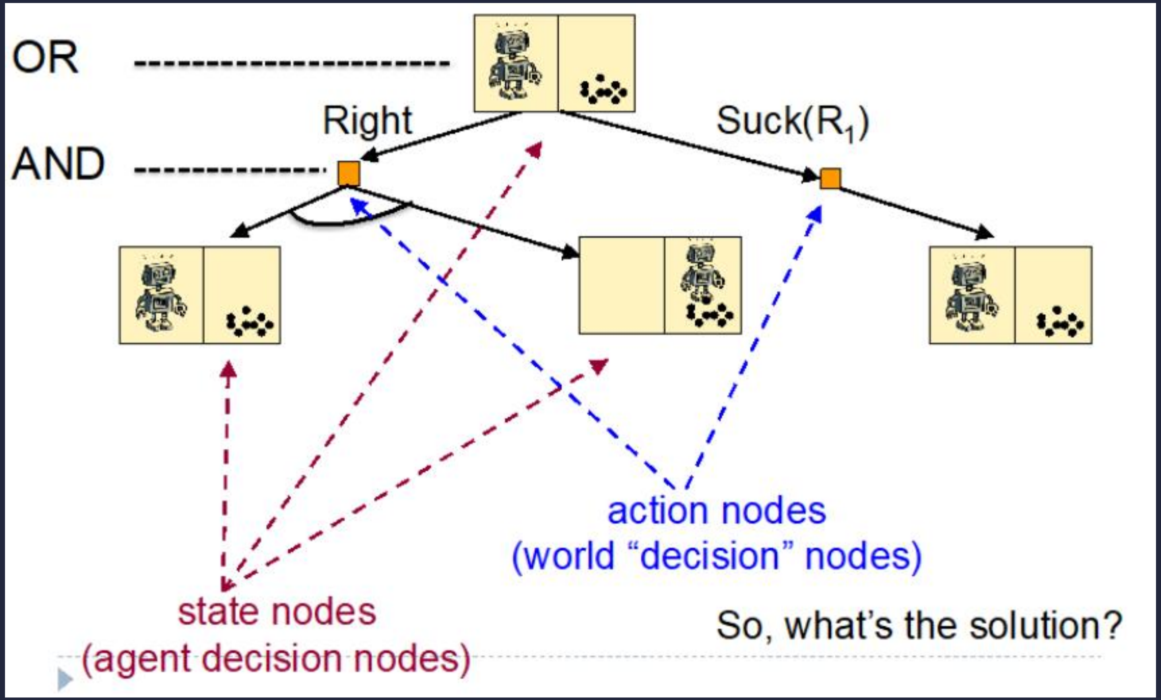
- The goal state is as follows:

$$(R_1\ is\ clean) \wedge (R_2\ is\ clean)$$

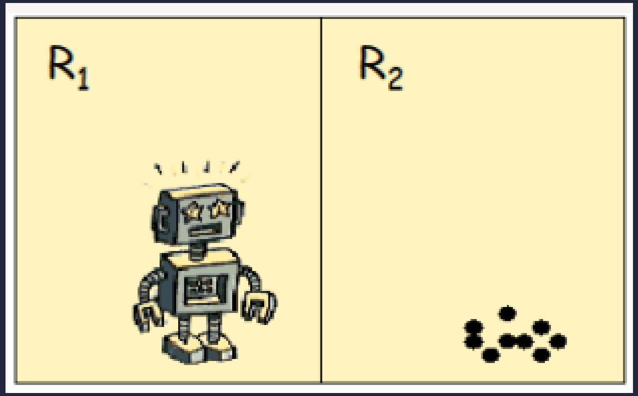
Slippery Vacuum Robot

- We can generate the following AND/OR tree, describing the context that the robot is in.
- At the OR level, the robot is able to make a choice.
- At the AND level, the environment acts upon the robot

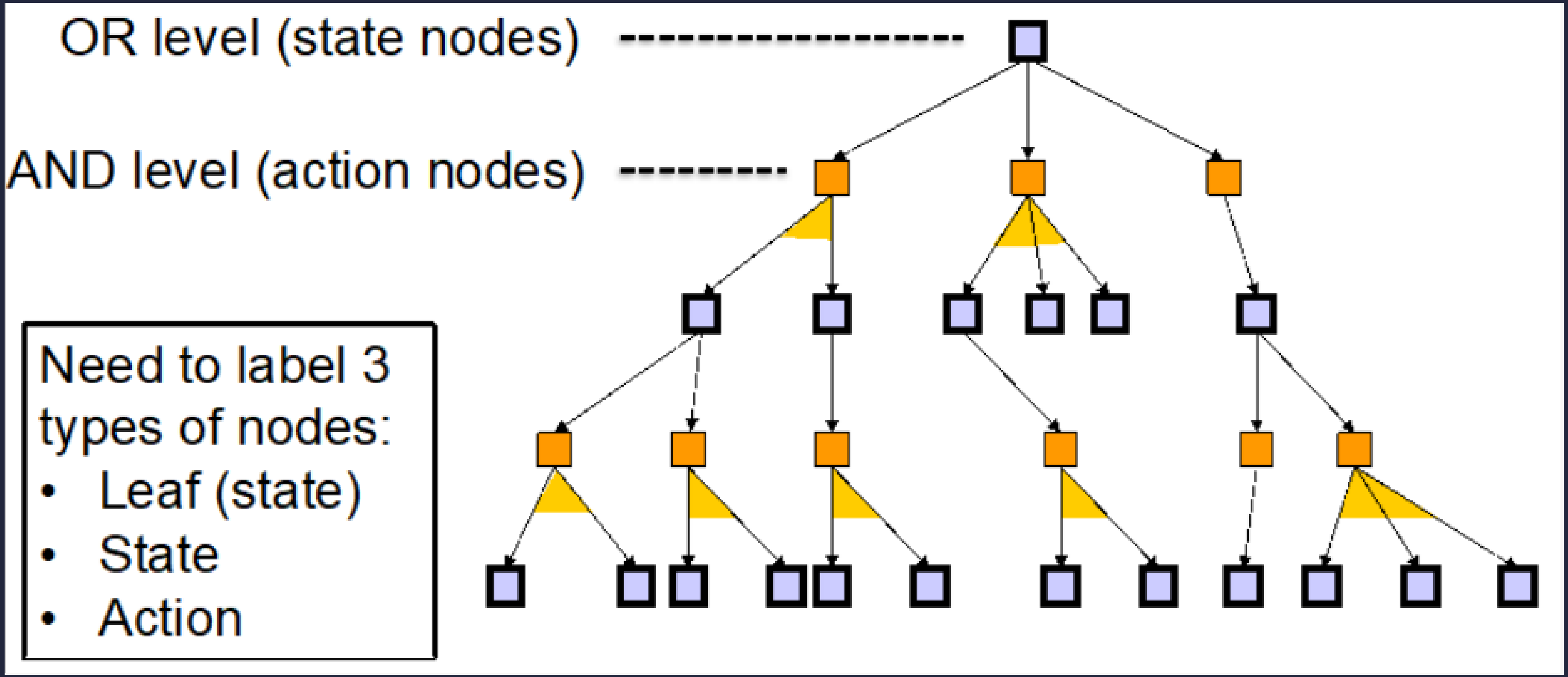




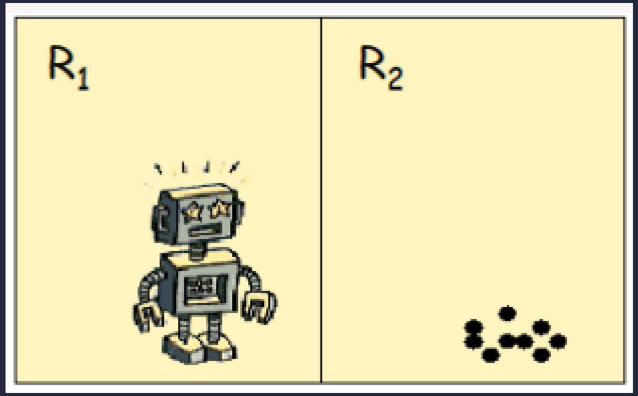
Solutions of AND/OR Trees



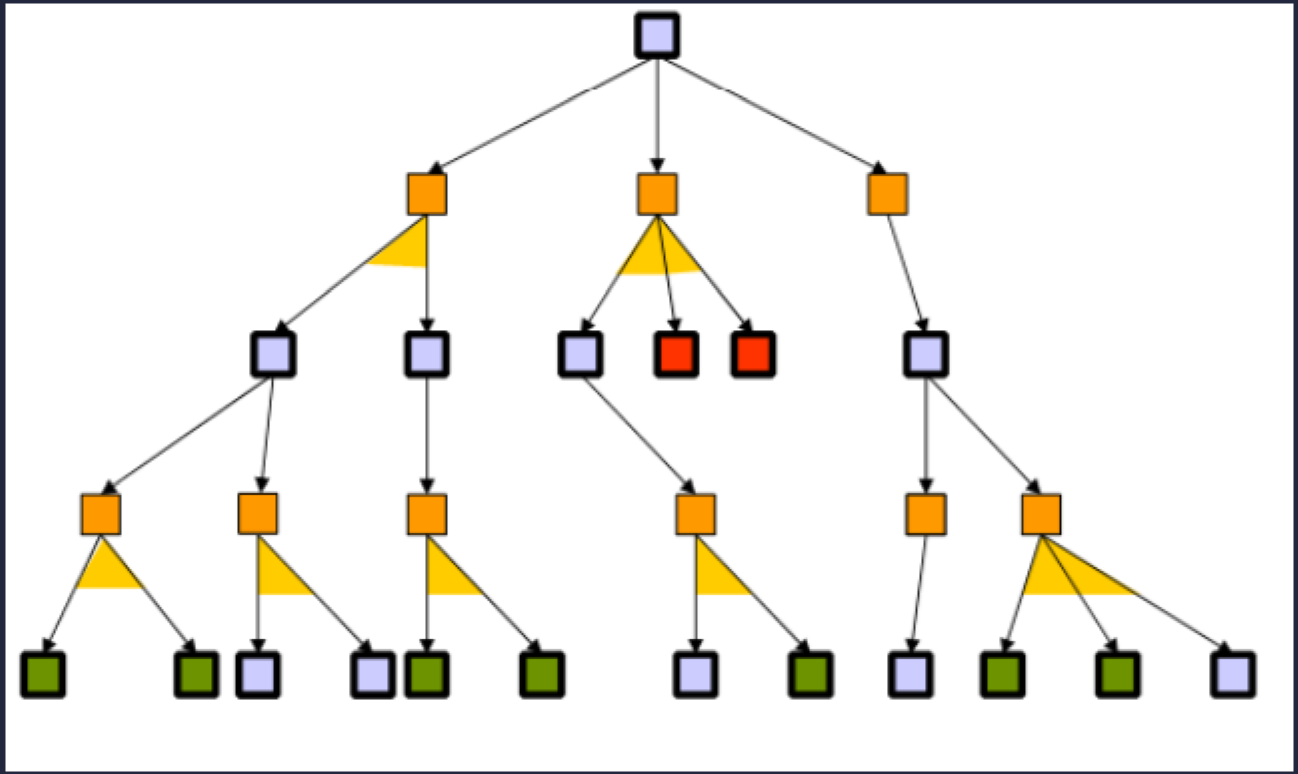
- A solution of an AND/OR tree isa sub-tree that meets 3 conditions:
  - Has a goal node at every leaf
  - Specifies one action at each node of an OR level
  - Includes every outcome branch at each node of an AND level.



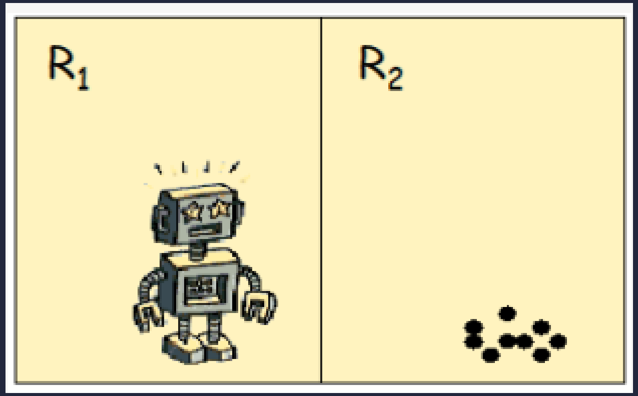
Solutions of AND/OR Trees



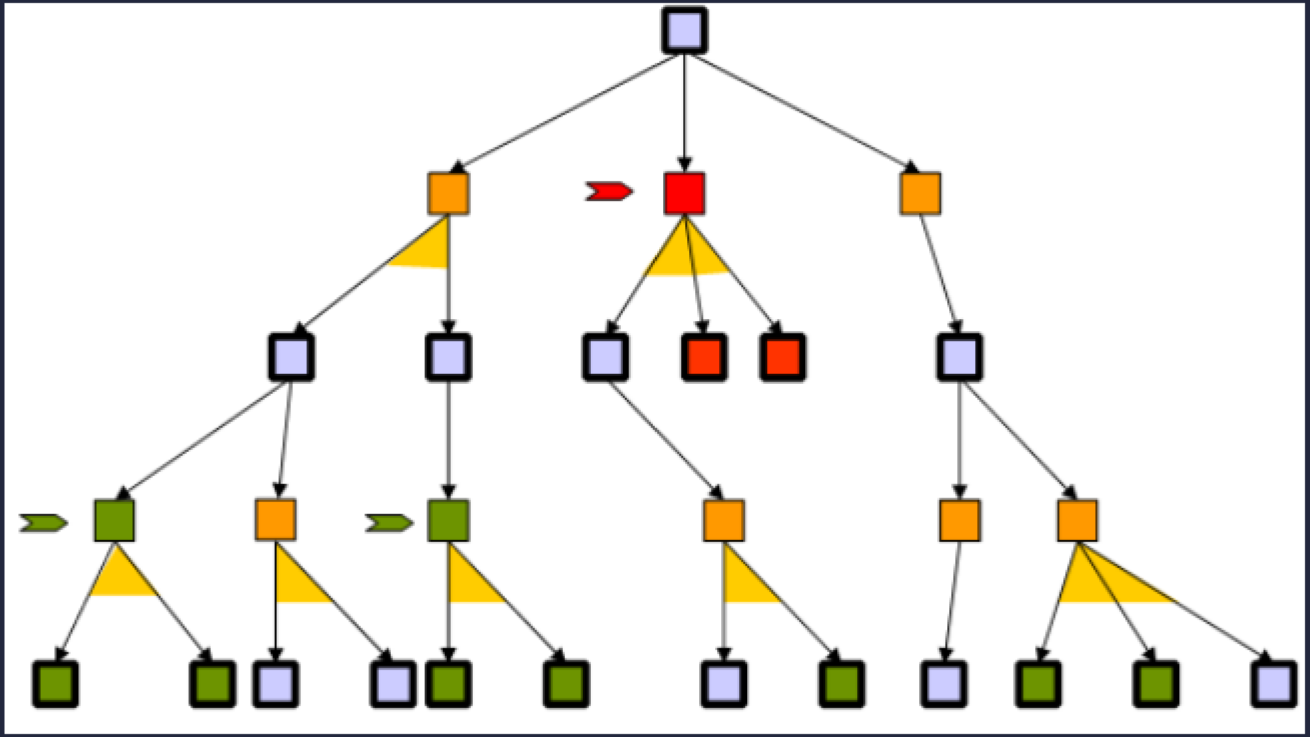
- A solution of an AND/OR tree isa sub-tree that meets 3 conditions:
  - Has a goal node at every leaf
  - Specifies one action at each node of an OR level
  - Includes every outcome branch at each node of an AND level.



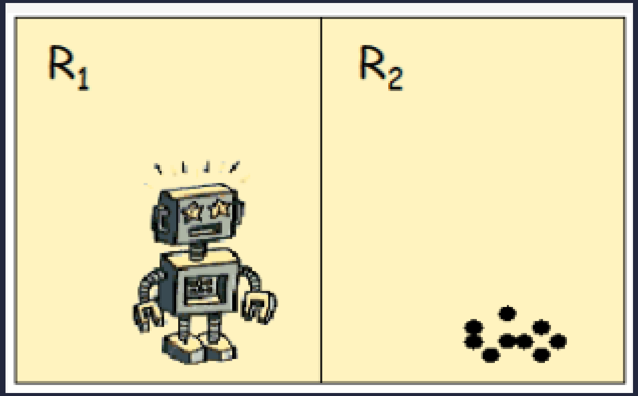
Solutions of AND/OR Trees



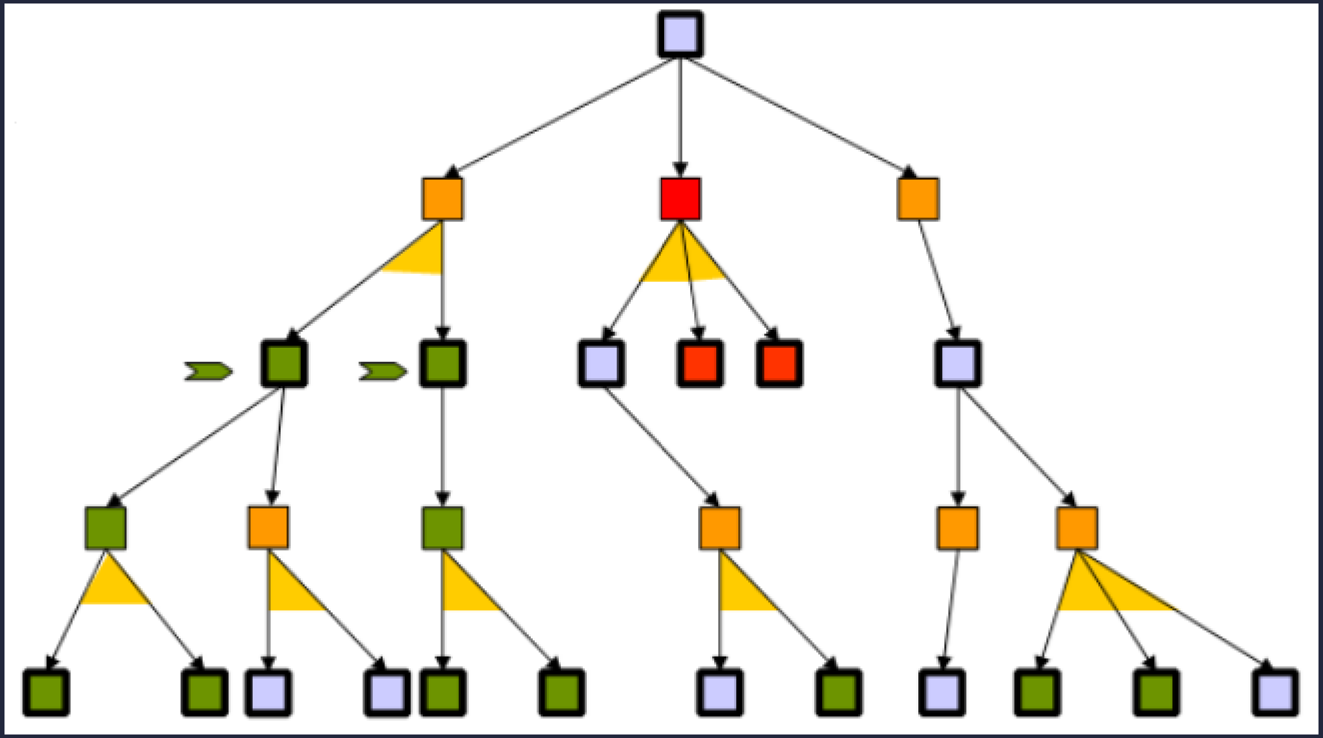
- A solution of an AND/OR tree isa sub-tree that meets 3 conditions:
  - Has a goal node at every leaf
  - Specifies one action at each node of an OR level
  - Includes every outcome branch at each node of an AND level.



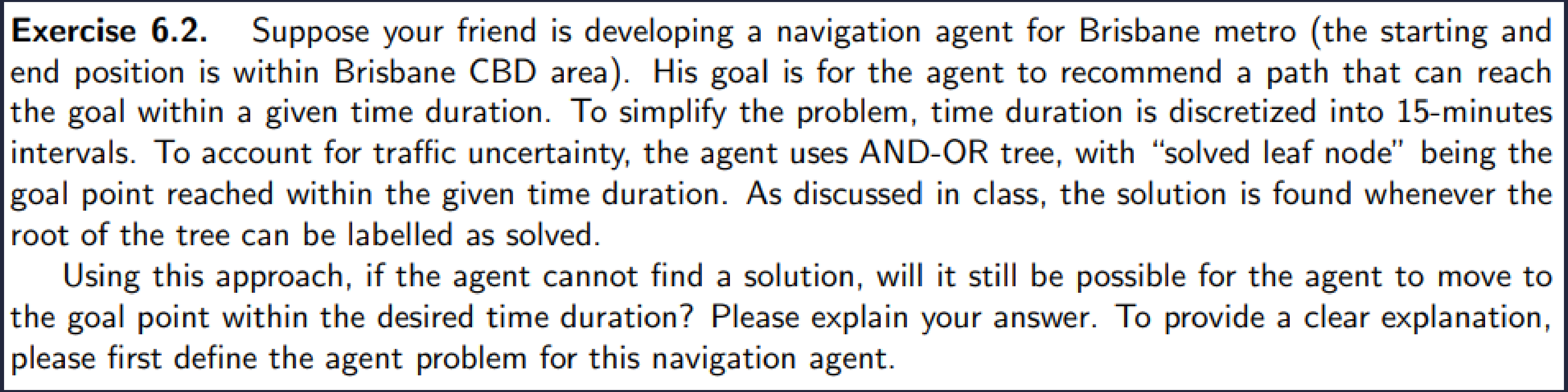
Solutions of AND/OR Trees



- A solution of an AND/OR tree isa sub-tree that meets 3 conditions:
  - Has a goal node at every leaf
  - Specifies one action at each node of an OR level
  - Includes every outcome branch at each node of an AND level.



Exercise 6.2



Exercise 6.2. Suppose your friend is developing a navigation agent for Brisbane metro (the starting and end position is within Brisbane CBD area). His goal is for the agent to recommend a path that can reach the goal within a given time duration. To simplify the problem, time duration is discretized into 15-minutes intervals. To account for traffic uncertainty, the agent uses AND-OR tree, with "solved leaf node" being the goal point reached within the given time duration. As discussed in class, the solution is found whenever the root of the tree can be labelled as solved.

Using this approach, if the agent cannot find a solution, will it still be possible for the agent to move to the goal point within the desired time duration? Please explain your answer. To provide a clear explanation, please first define the agent problem for this navigation agent.

Exercise 6.2

Use AND/OR trees to find solutions in non-deterministic search problems.

Exercise 6.2

Use AND/OR trees to find solutions in non-deterministic search problems.

- We know that the agent is able to control the actions that it takes (where it goes)
- We also know that the agent is unable to control certain things (e.g. traffic)

Exercise 6.2

Use AND/OR trees to find solutions in non-deterministic search problems.

- We know that the agent is able to control the actions that it takes (where it goes)
- We also know that the agent is unable to control certain things (e.g. traffic)

Exercise 6.2

Use AND/OR trees to find solutions in non-deterministic search problems.

- We know that the agent is able to control the actions that it takes (where it goes)
- We also know that the agent is unable to control certain things (e.g. traffic)

If the agent is unable to find a solution, it only means that there is no guaranteed solution – the agent could still move to the goal point in time if the traffic happens to be good.

COMP3702 Tutorial 6

Matt Choy | matthew.choy@uq.edu.au

https://github.com/mattpchoy/comp3702-tutorials