
CIS 471 / 571 (Fall 2020): Introduction to Artificial Intelligence

Assignment Project Exam Help

Lecture 7: Expectimax, Utilities

<https://tutorcs.com>

WeChat: cstutorcs

Thanh H. Nguyen

Source: <http://ai.berkeley.edu/home.html>



Reminders

- Project 2: Multi-agent Search

- Deadline: Oct 27th, 2020

Assignment Project Exam Help

- Homework 2: CSPs and Games

- Deadline: Oct 24th, 2020

<https://tutorcs.com>

Wechat: cstutorcs

Today

- Expectimax Search

Assignment Project Exam Help

- Utilities

<https://tutorcs.com>

WeChat: cstutorcs

Uncertain Outcomes

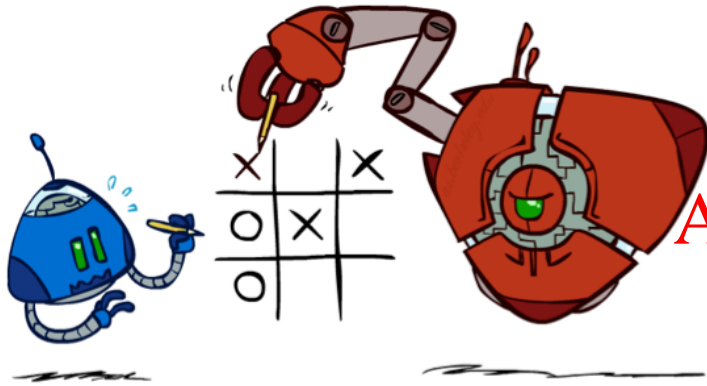
Assignment Project Exam Help

<https://tutorcs.com>

WeChat: cstutorcs



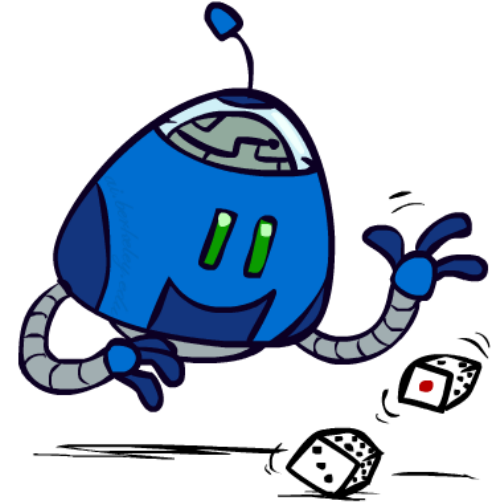
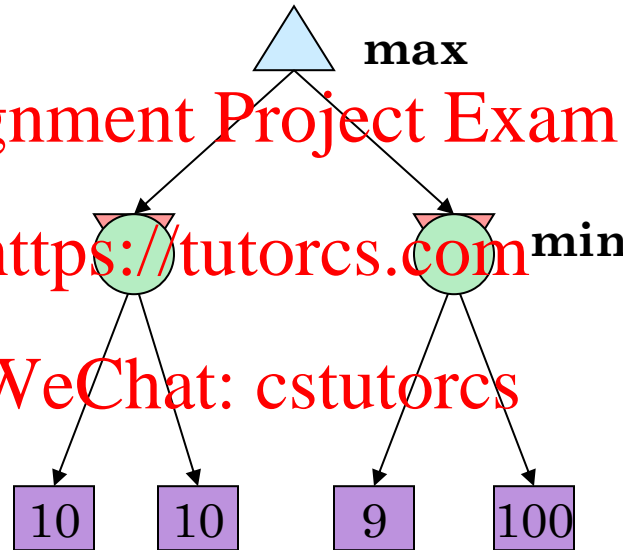
Worst-Case vs. Average Case



Assignment Project Exam Help

<https://tutorcs.com>

WeChat: cstutorcs



Idea: Uncertain outcomes controlled by chance, not an adversary!



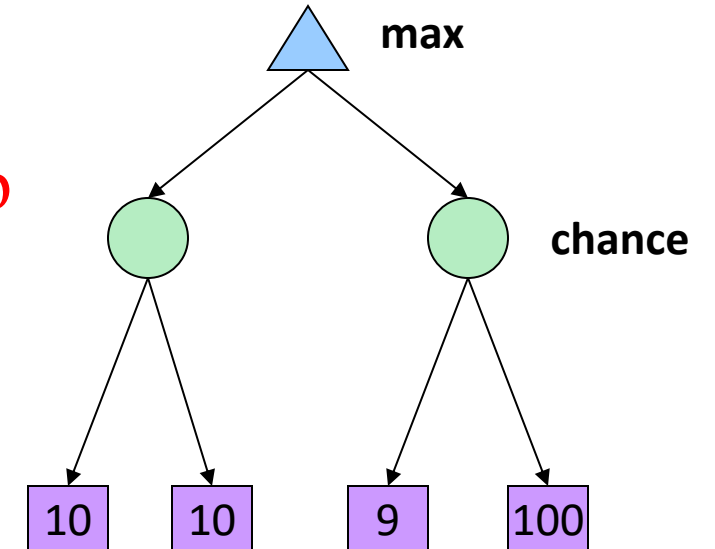
Expectimax Search

- Why wouldn't we know what the result of an action will be?
 - Explicit randomness: rolling dice
 - Unpredictable opponents: the ghosts respond randomly
 - Actions can fail: when moving a robot, wheels might slip

- Values should now reflect average-case (expectimax) outcomes, not worst-case (minimax) outcomes

- Expectimax search:** compute the average score under optimal play
 - Max nodes as in minimax search
 - Chance nodes are like min nodes but the outcome is uncertain
 - Calculate their **expected utilities**
 - I.e. take weighted average (expectation) of children

- Later, we'll learn how to formalize the underlying uncertain-result problems as **Markov Decision Processes**



Expectimax Pseudocode

```
def value(state):
```

```
    if the state is a terminal state: return the state's utility
```

```
    if the next agent is MAX: return max-value(state)
```

```
    if the next agent is EXP: return exp-value(state)
```

<https://tutorcs.com>

WeChat: cstutorcs

```
def max-value(state):
```

```
    initialize v =  $-\infty$ 
```

```
    for each successor of state:
```

```
        v = max(v, value(successor))
```

```
    return v
```

```
def exp-value(state):
```

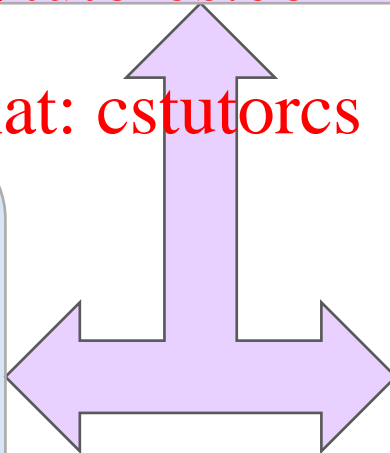
```
    initialize v = 0
```

```
    for each successor of state:
```

```
        p = probability(successor)
```

```
        v += p * value(successor)
```

```
    return v
```



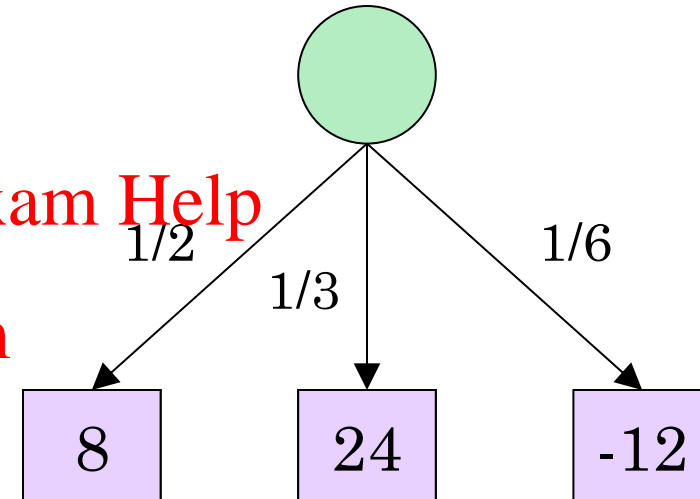
Expectimax Pseudocode

```
def exp-value(state):  
    initialize v = 0  
    for each successor of state:  
        p = probability(successor)  
        v += p * value(successor)  
    return v
```

Assignment Project Exam Help

<https://tutorcs.com>

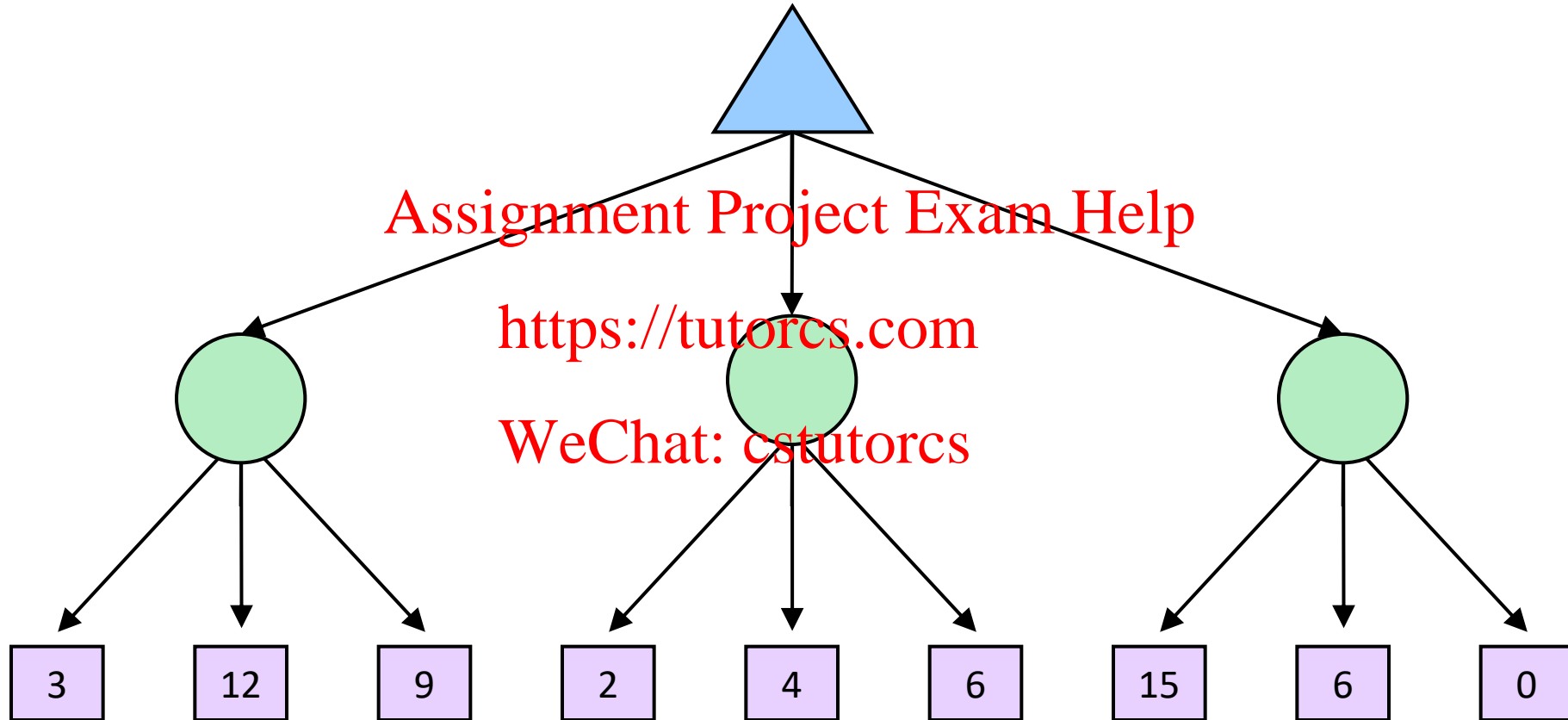
WeChat: cstutorcs



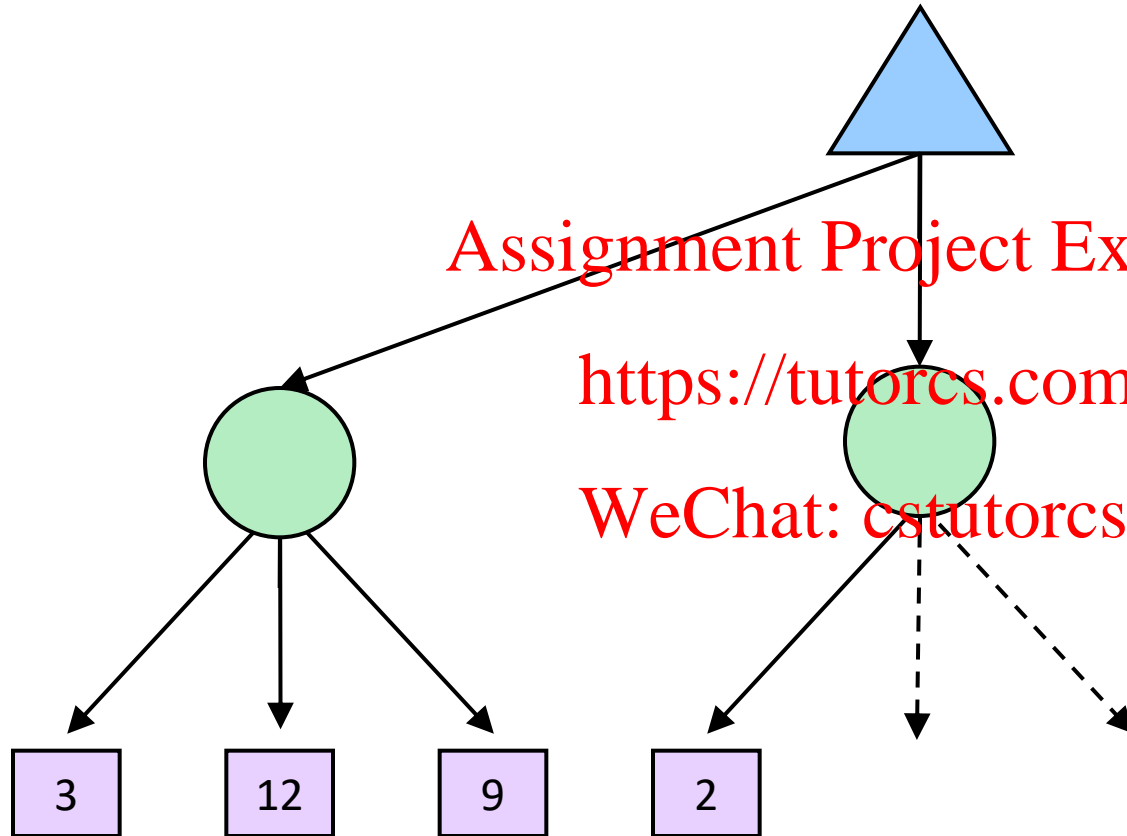
$$v = (1/2) (8) + (1/3) (24) + (1/6) (-12) = 10$$



Expectimax Example



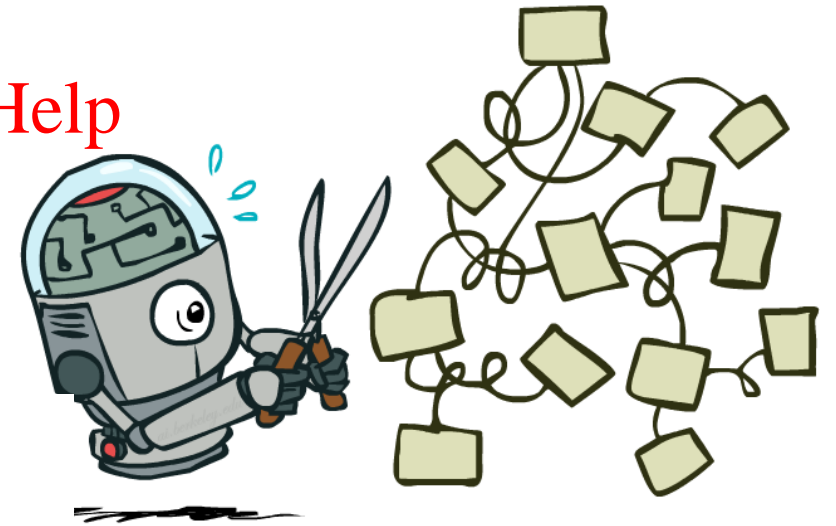
Expectimax Pruning?



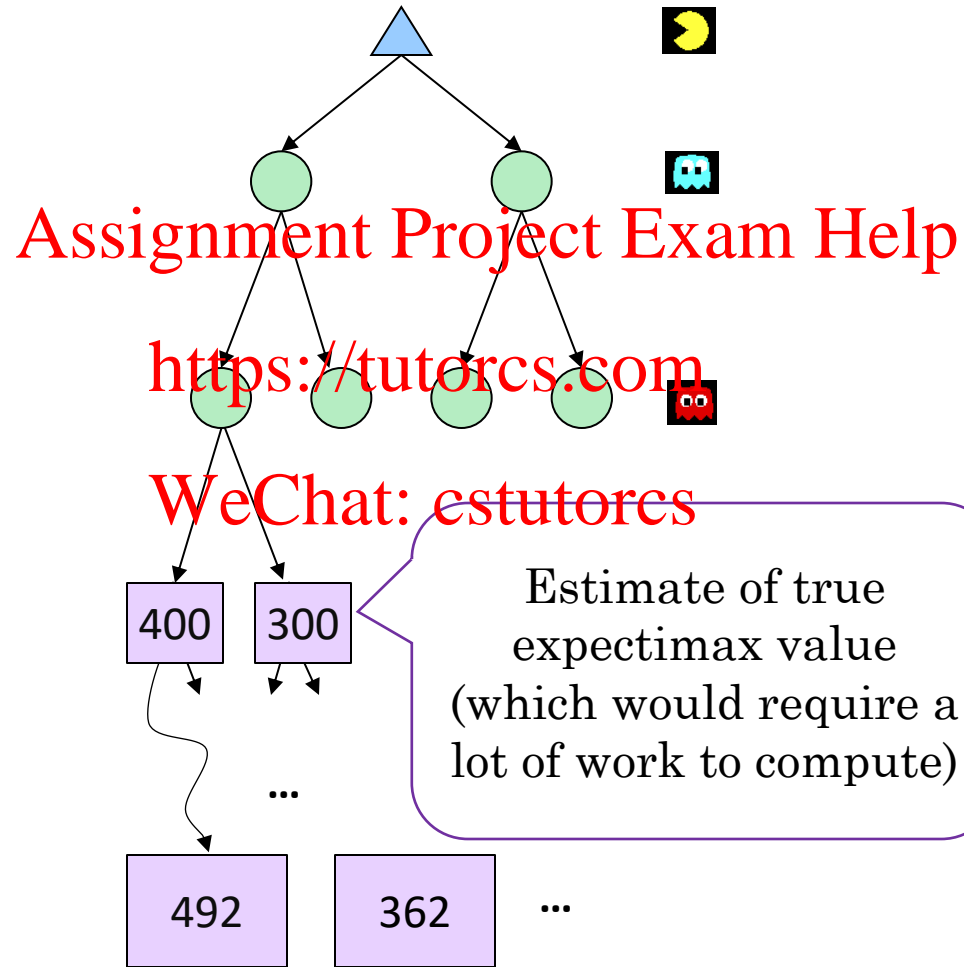
Assignment Project Exam Help

<https://tutorcs.com>

WeChat: cstutorcs



Depth-Limited Expectimax

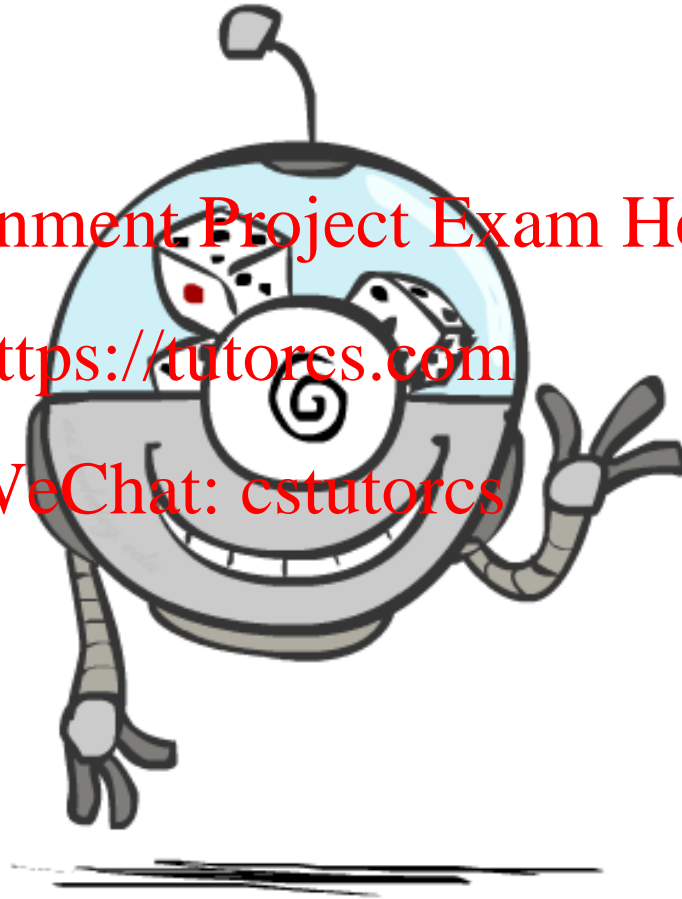


Probabilities

Assignment Project Exam Help

<https://tutorcs.com>

WeChat: cstutorcs



Reminder: Probabilities

- A **random variable** represents an event whose outcome is unknown
- A **probability distribution** is an assignment of weights to outcomes

- Example: Traffic on freeway

- Random variable: T = whether there's traffic
- Outcomes: T in {none, light, heavy}
- Distribution: $P(T=\text{none}) = 0.25$, $P(T=\text{light}) = 0.50$, $P(T=\text{heavy}) = 0.25$

WeChat: cstutorcs

- Some laws of probability (more later):
 - Probabilities are always non-negative
 - Probabilities over all possible outcomes sum to one
- As we get more evidence, probabilities may change:
 - $P(T=\text{heavy}) = 0.25$, $P(T=\text{heavy} \mid \text{Hour}=8\text{am}) = 0.60$
 - We'll talk about methods for reasoning and updating probabilities later



0.25



0.50



0.25



Reminder: Expectations

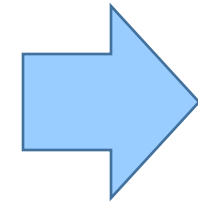
- The expected value of a function of a random variable is the average, weighted by the probability distribution over outcomes

Assignment Project Exam Help

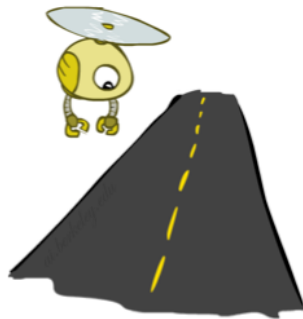
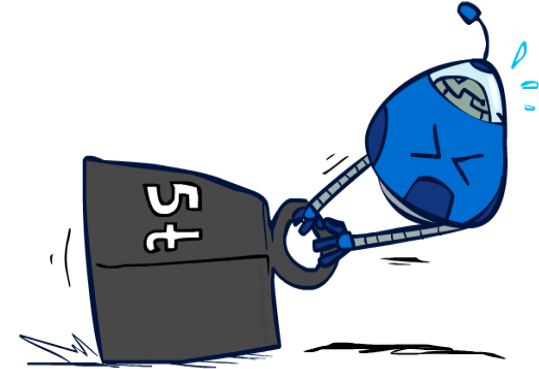
- Example: How long to get to the airport?

Time:	20 min	30 min	60 min
	x	x	x
Probability:	0.25	0.50	0.25

WeChat: cstutorcs

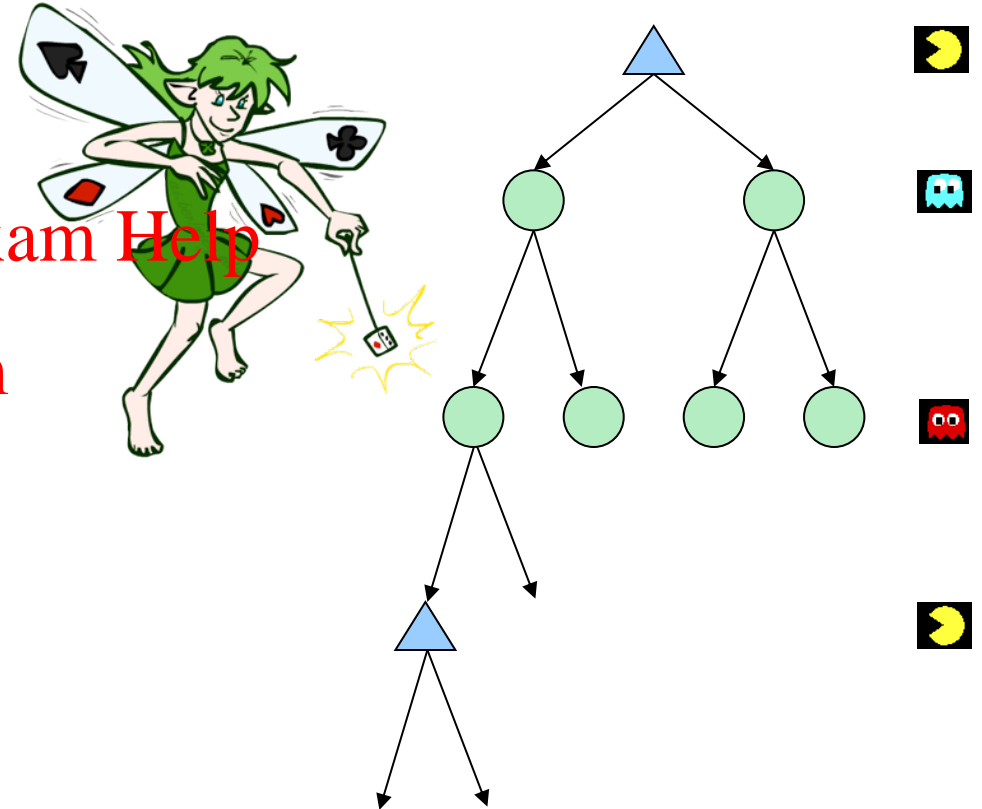


35 min



What Probabilities to Use?

- In expectimax search, we have a probabilistic model of how the opponent (or environment) will behave in any state
 - Model could be a simple uniform distribution (roll a die)
 - Model could be sophisticated and require a great deal of computation
 - We have a chance node for any outcome out of our control: opponent or environment
 - The model might say that adversarial actions are likely!
- For now, assume each chance node magically comes along with probabilities that specify the distribution over its outcomes



Having a probabilistic belief about another agent's action does not mean that the agent is flipping any coins!

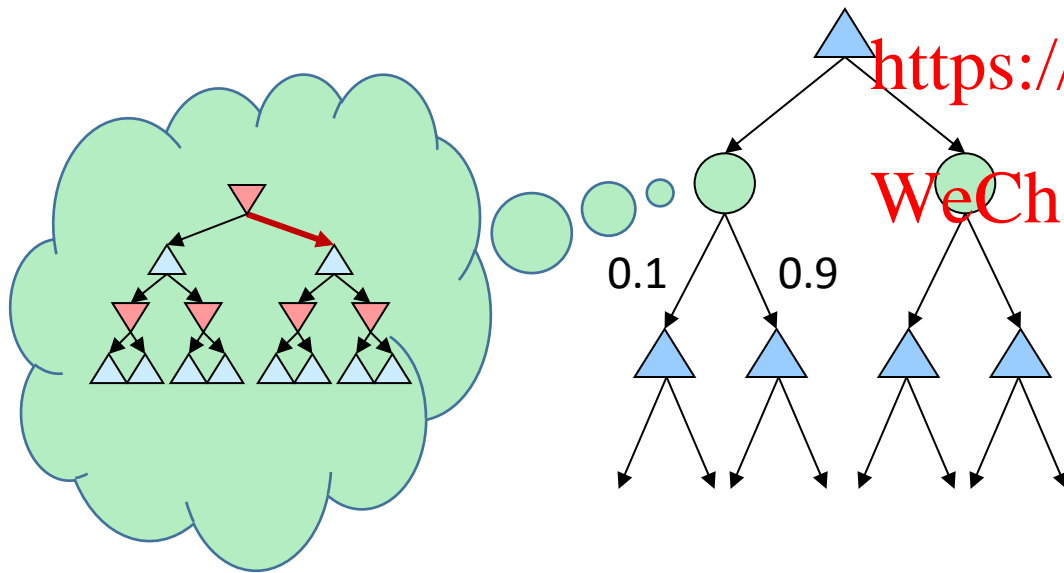
Quiz: Informed Probabilities

- Let's say you know that your opponent is actually running a depth 2 minimax, using the result 80% of the time, and moving randomly otherwise
- Question: What tree search should you use?

Assignment Project Exam Help

<https://tutorcs.com>: Expectimax!

WeChat: cstutorcs



- To figure out EACH chance node's probabilities, you have to run a simulation of your opponent
- This kind of thing gets very slow very quickly
- Even worse if you have to simulate your opponent simulating you...
- ... except for minimax, which has the nice property that it all collapses into one game tree



Modeling Assumptions



Assignment Project Exam Help

<https://tutorcs.com>

WeChat: cstutorcs



The Dangers of Optimism and Pessimism

Dangerous Optimism

Assuming chance when the world is adversarial



Dangerous Pessimism

Assuming the worst case when it's not likely



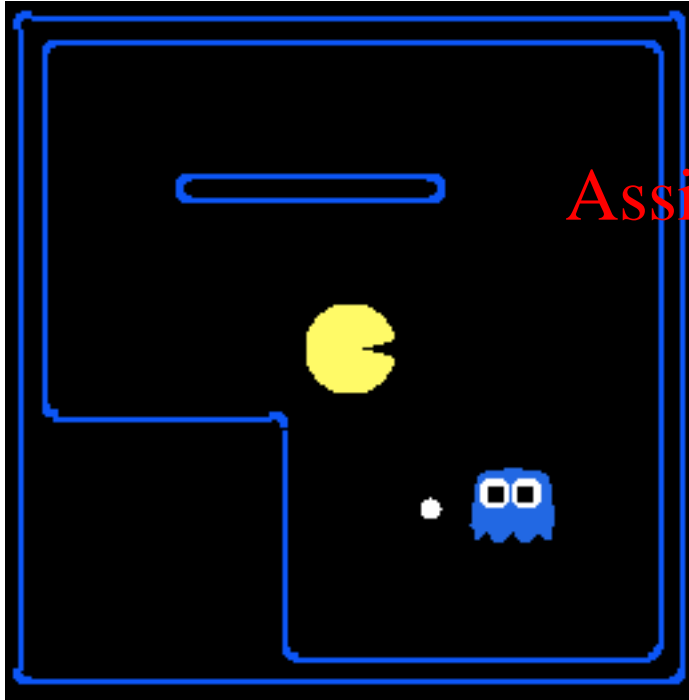
Assignment Project Exam Help

<https://tutorcs.com>

WeChat: cstutorcs



Assumptions vs. Reality



Assignment Project Exam Help

<https://tutorcs.com>

WeChat: cstutorcs

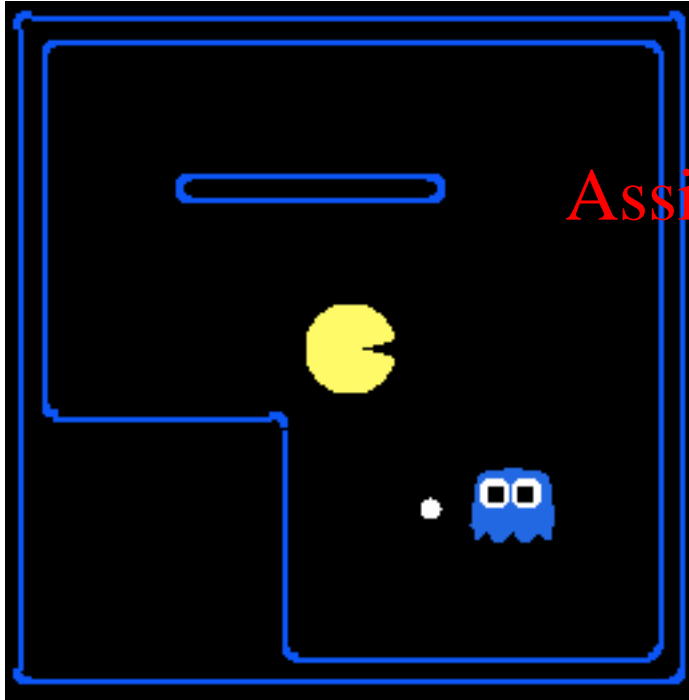
	Adversarial Ghost	Random Ghost
Minimax Pacman	1	3
Expectimax Pacman	1	1

Results from playing 5 games

Pacman used depth 4 search with an eval function that avoids trouble
Ghost used depth 2 search with an eval function that seeks Pacman



Assumptions vs. Reality



Assignment Project Exam Help

<https://tutorcs.com>

WeChat: cstutorcs

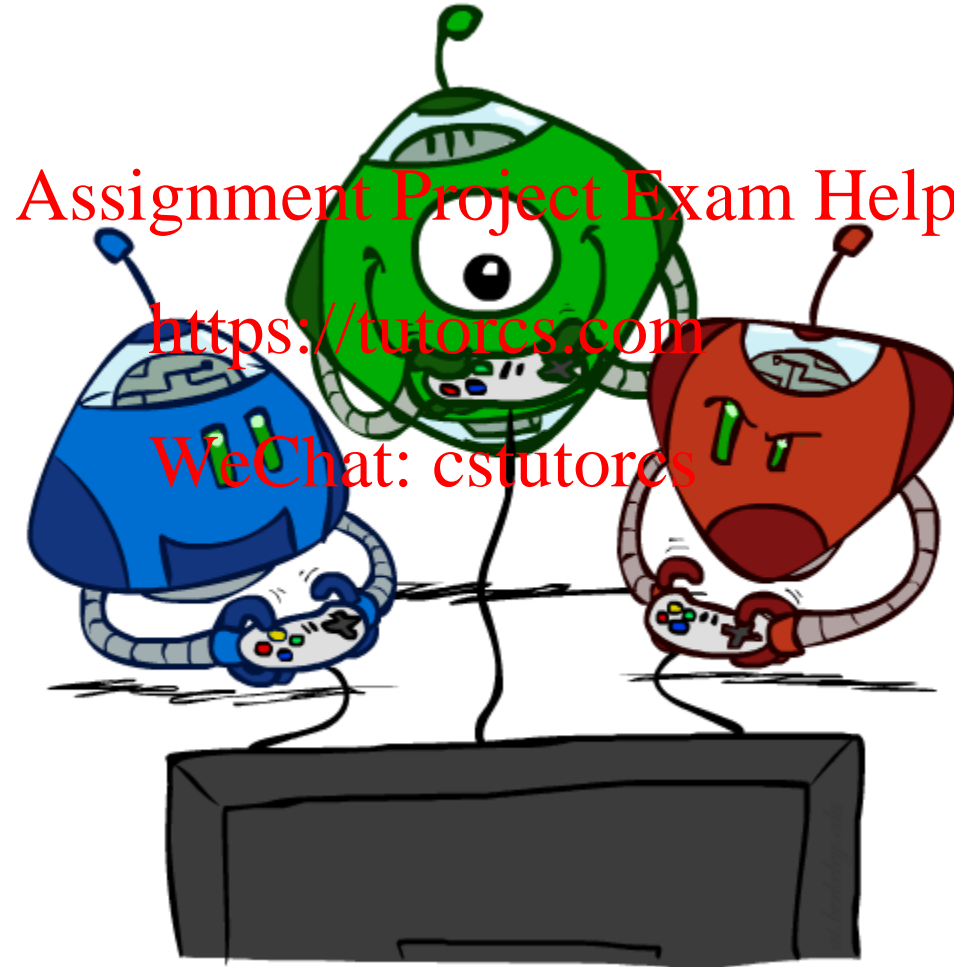
	Adversarial Ghost	Random Ghost
Minimax Pacman	Won 5/5 Avg. Score: 483	Won 5/5 Avg. Score: 493
Expectimax Pacman	Won 1/5 Avg. Score: -303	Won 5/5 Avg. Score: 503

Results from playing 5 games

Pacman used depth 4 search with an eval function that avoids trouble
Ghost used depth 2 search with an eval function that seeks Pacman



Other Game Types



Mixed Layer Types

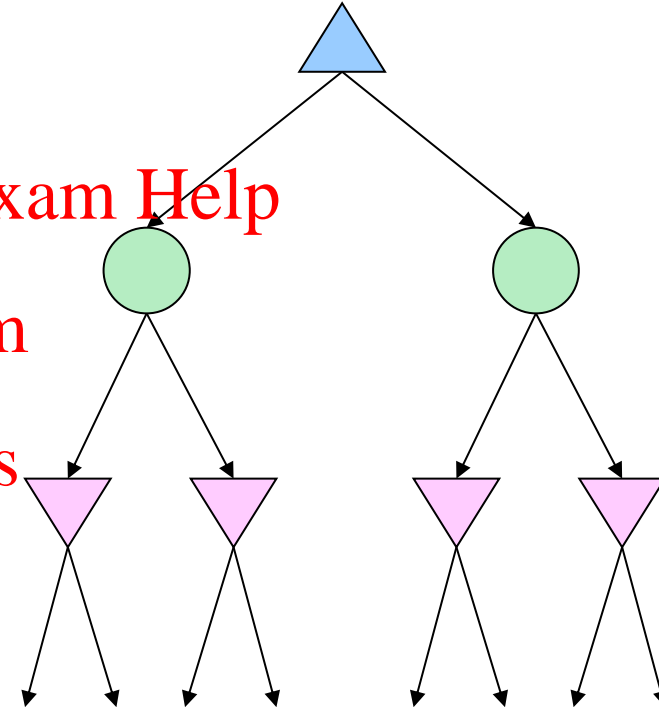
- E.g. Backgammon
- Expectiminimax
 - Environment is an extra “random agent” player that moves after each min/max agent
 - Each node computes the appropriate combination of its children



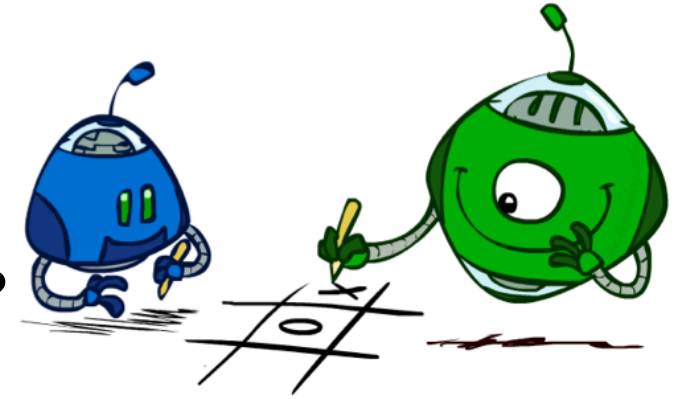
Assignment Project Exam Help

<https://tutorcs.com>

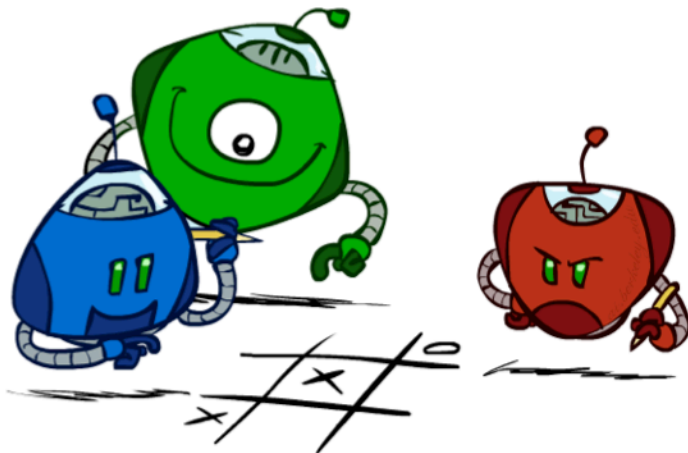
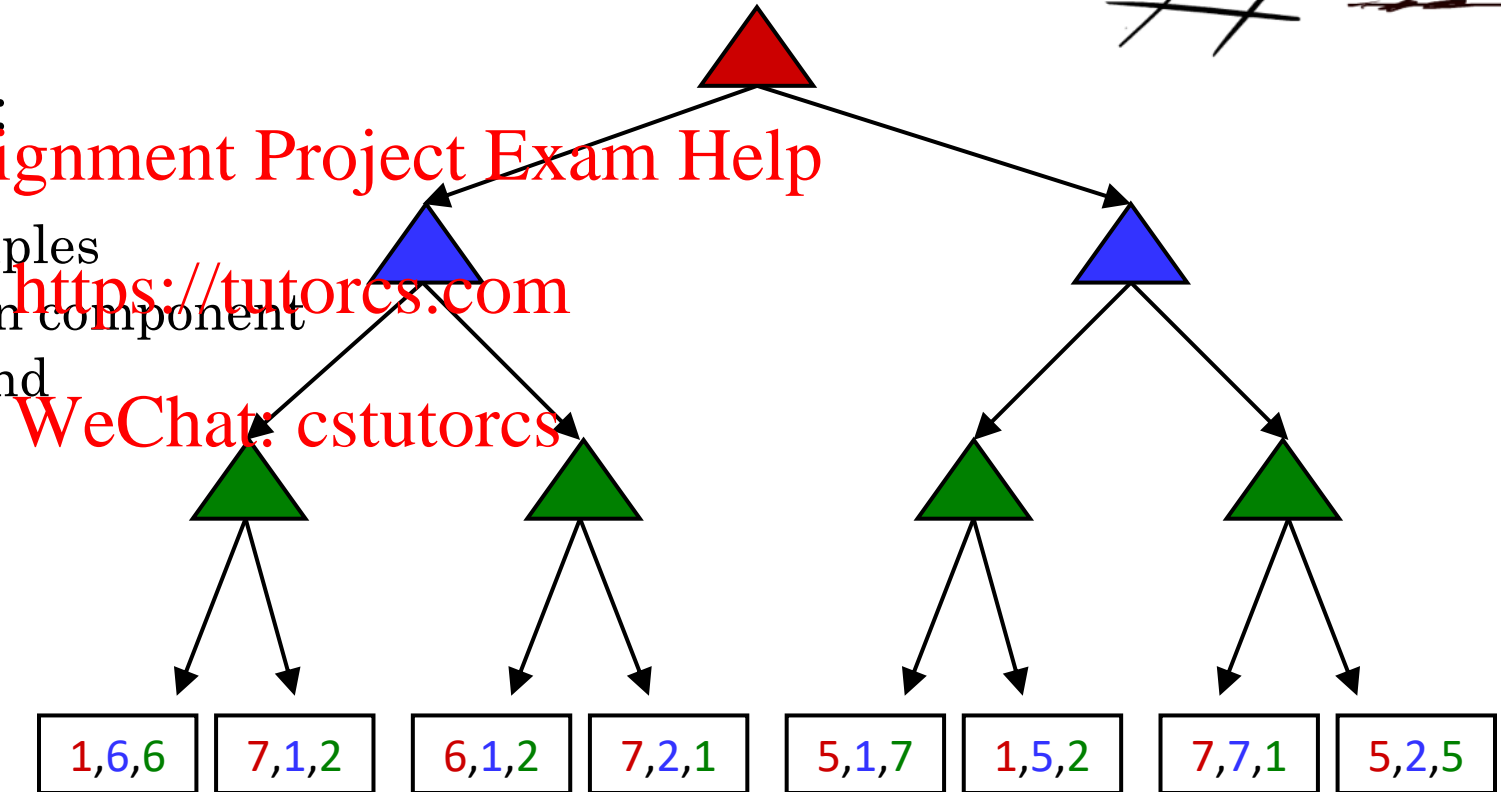
WeChat: cstutorcs



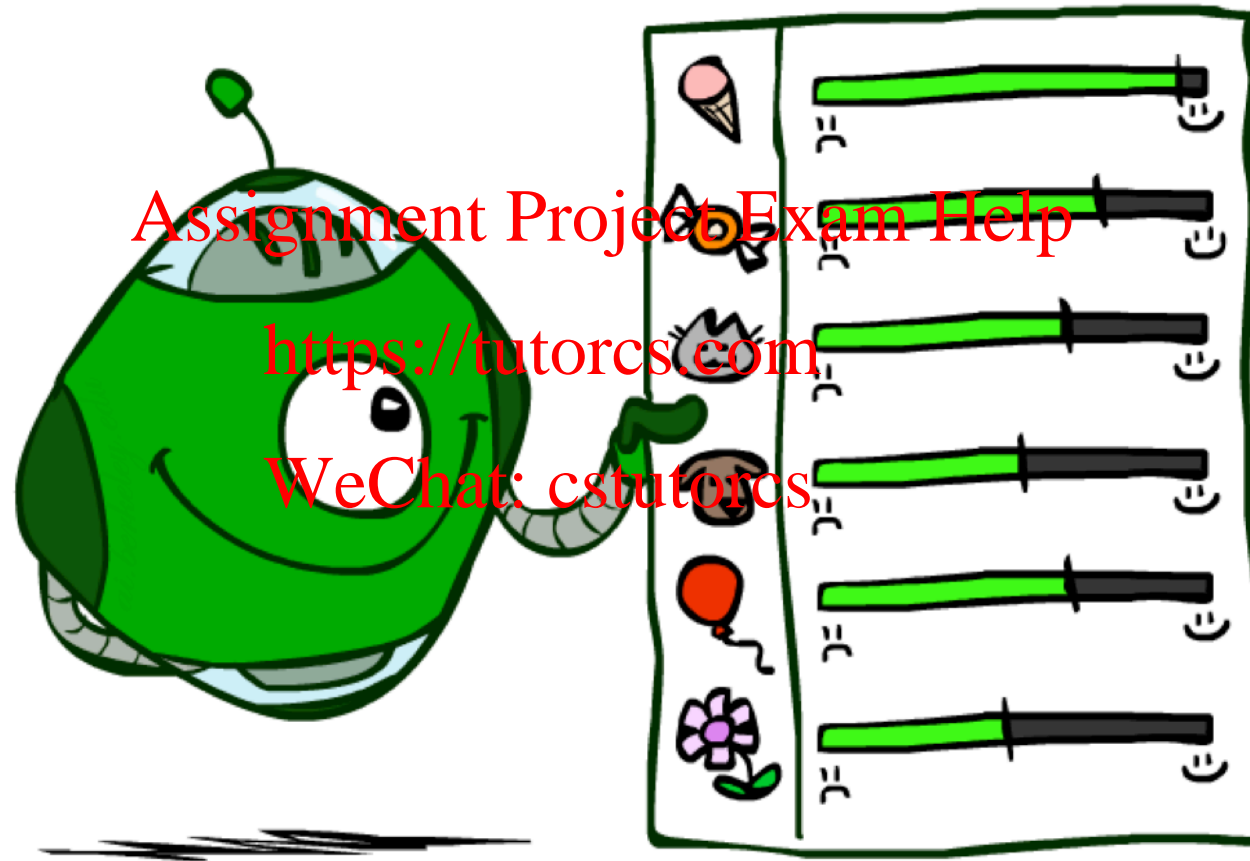
Multi-Agent Utilities



- What if the game is not zero-sum, or has multiple players?
- Generalization of minimax:
 - Terminals have utility tuples
 - Node values are also utility tuples
 - Each player maximizes its own component
 - Can give rise to cooperation and competition dynamically...



Utilities



Assignment Project Exam Help

<https://tutorcs.com>

WeChat: cstutorcs



Maximum Expected Utility

- Why should we average utilities? Why not minimax?

- Principle of maximum expected utility:

- A rational agent should choose the action that maximizes its expected utility, given its knowledge

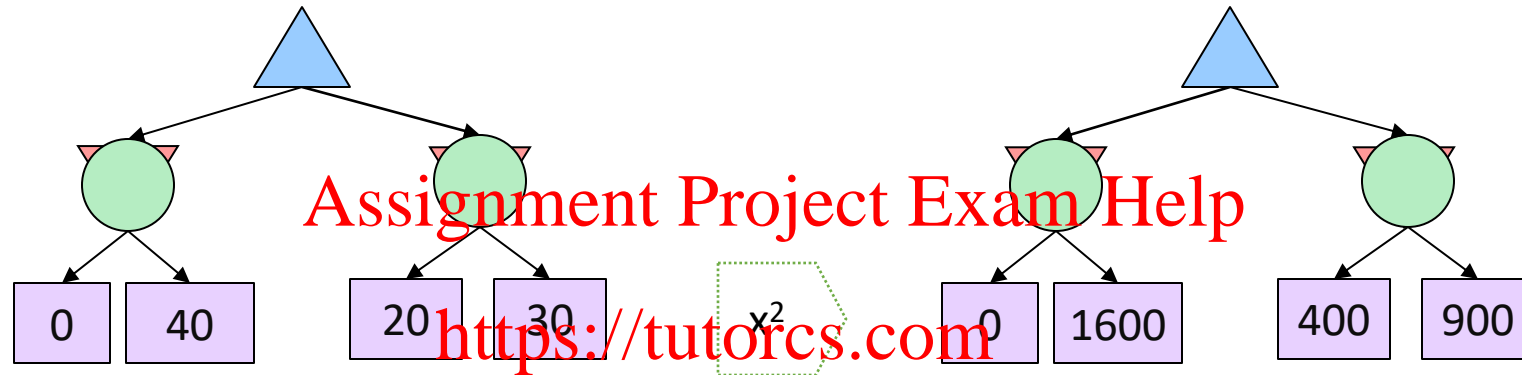
WeChat: cstutorcs

- Questions:

- Where do utilities come from?
- How do we know such utilities even exist?
- How do we know that averaging even makes sense?
- What if our behavior (preferences) can't be described by utilities?



What Utilities to Use?



WeChat: cstutorcs

- For worst-case minimax reasoning, terminal function scale doesn't matter
 - We just want better states to have higher evaluations (get the ordering right)
 - We call this **insensitivity to monotonic transformations**
- For average-case expectimax reasoning, we need *magnitudes* to be meaningful



Utilities

- Utilities are functions from outcomes (states of the world) to real numbers that describe an agent's preferences

Assignment Project Exam Help

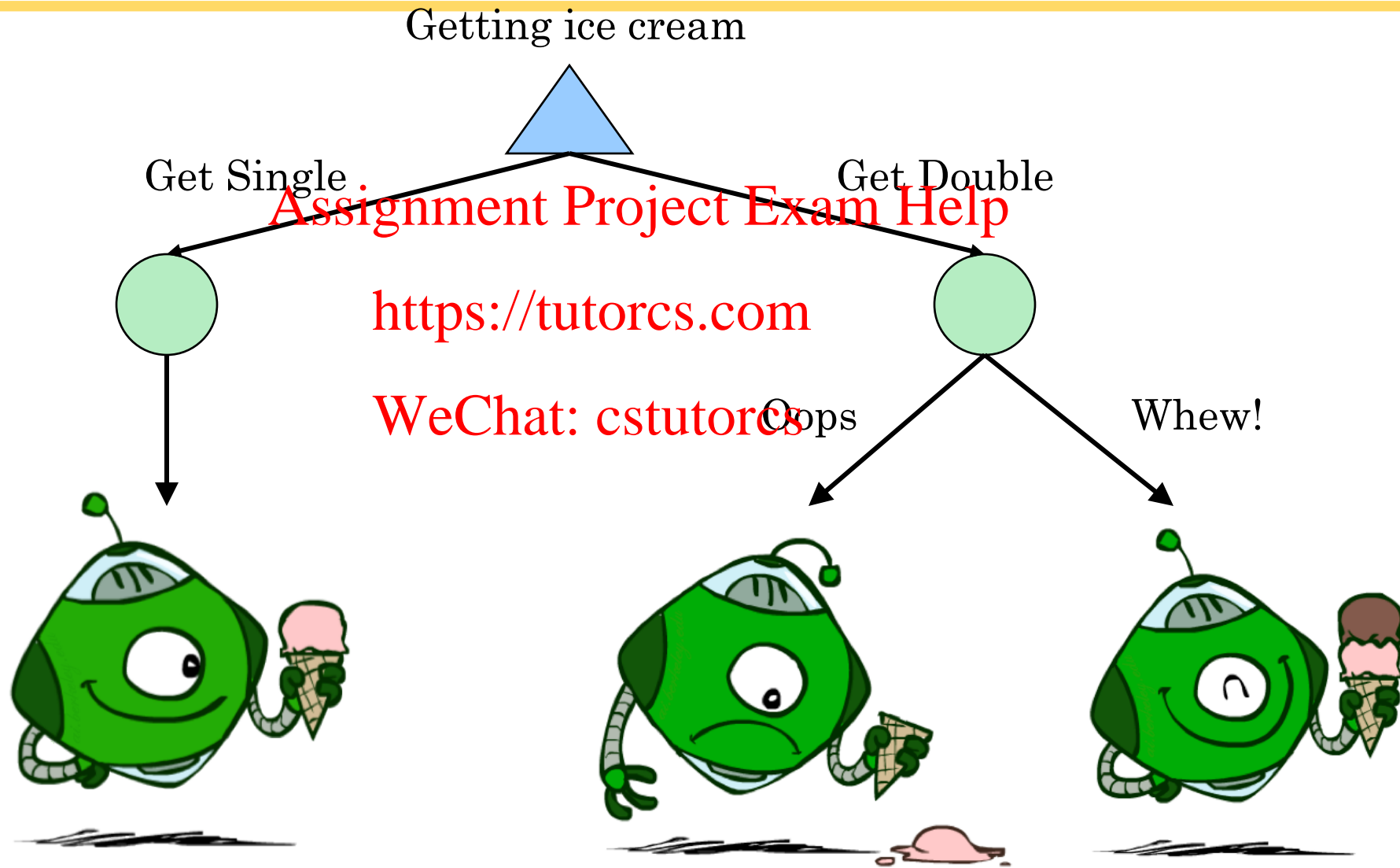
<https://tutorcs.com>

- Where do utilities come from?

- In a game, may be simple (+1/-1)
- Utilities summarize the agent's goals
- Theorem: any “rational” preferences can be summarized as a utility function



Utilities: Uncertain Outcomes



Assignment Project Exam Help

<https://tutorcs.com>

WeChat: cstutorcs



Preferences

- An agent must have preferences among:

- Prizes: A , B , etc.
- Lotteries: situations with uncertain prizes

$$L = [p, A; (1 - p), B]$$

Assignment Project Exam Help

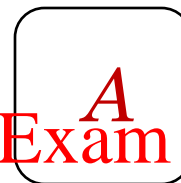
<https://tutorcs.com>

WeChat: cstutorcs

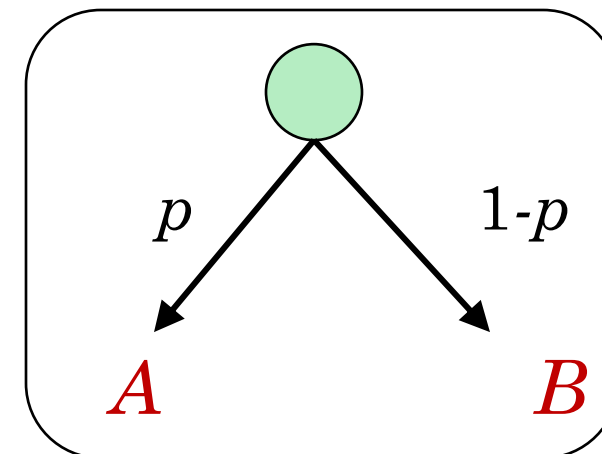
- Notation:

- Preference: $A \succ B$
- Indifference: $A \sim B$

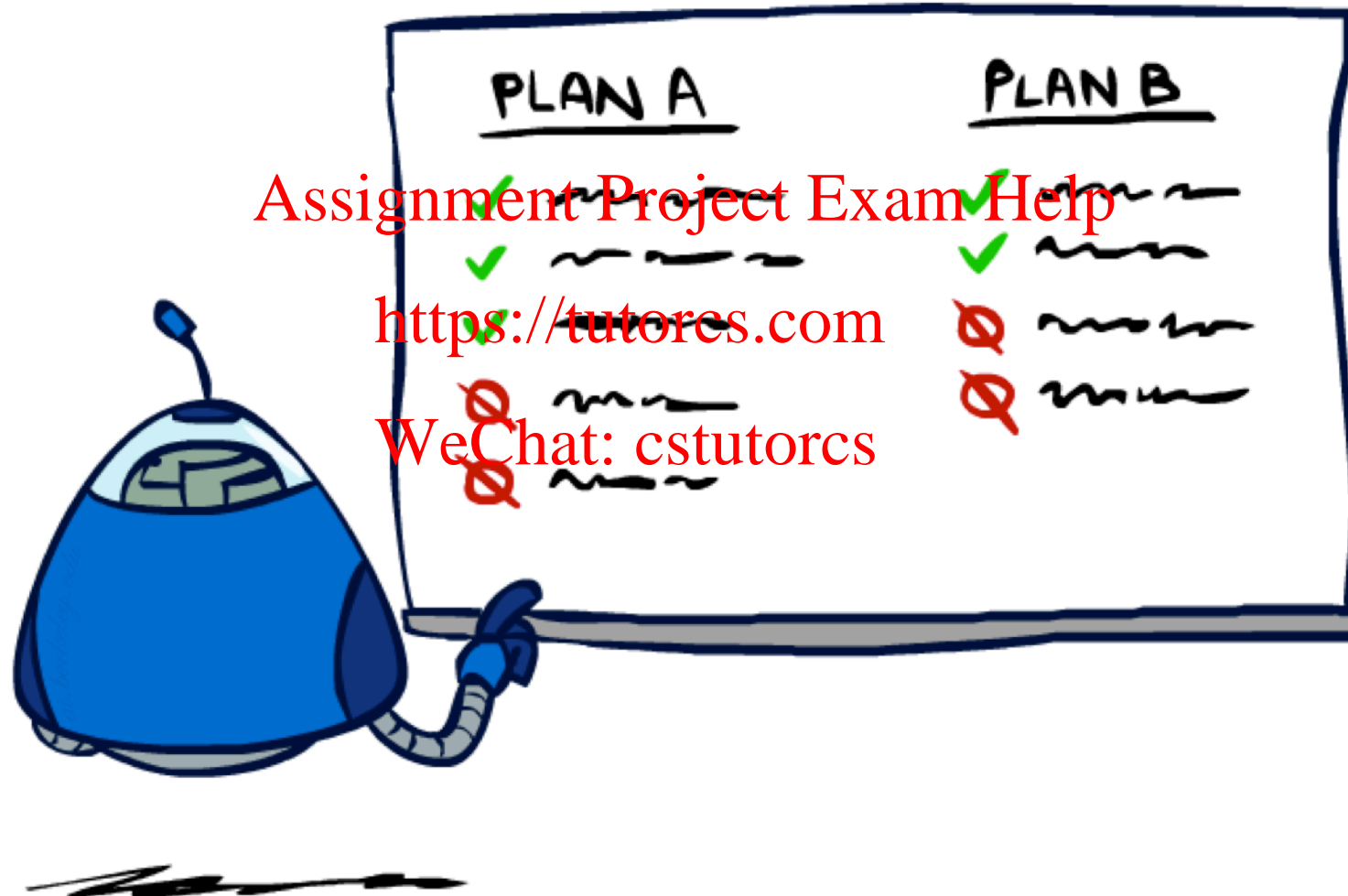
A Prize



A Lottery



Rationality



Rational Preferences

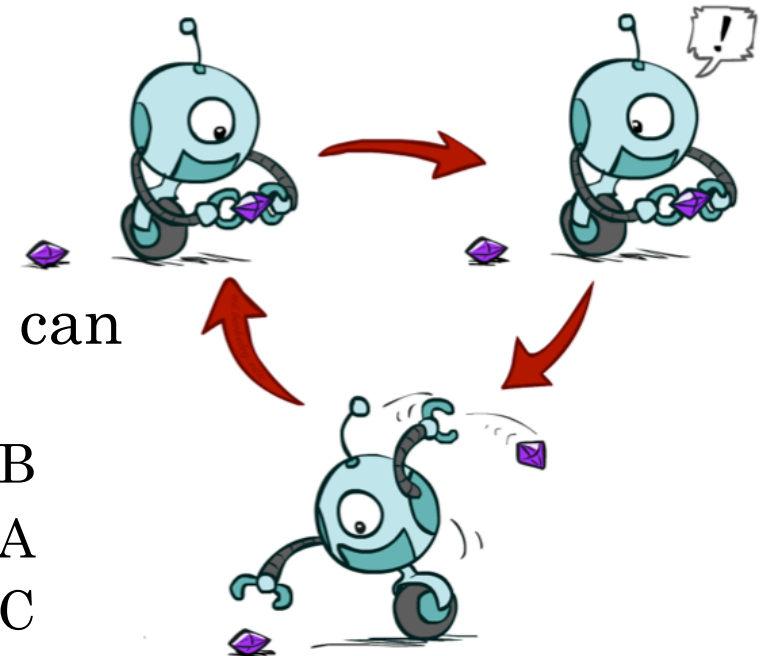
- We want some constraints on preferences before we call them rational, such as:

Axiom of Transitivity: $(A \succ B) \wedge (B \succ C) \Rightarrow (A \succ C)$

<https://tutorcs.com>

WeChat: cstutorcs

- For example: an agent with **intransitive preferences** can be induced to give away all of its money
 - If $B \succ C$, then an agent with C would pay (say) 1 cent to get B
 - If $A \succ B$, then an agent with B would pay (say) 1 cent to get A
 - If $C \succ A$, then an agent with A would pay (say) 1 cent to get C



Rational Preferences

The Axioms of Rationality

Orderability

$$(A \succ B) \vee (B \succ A) \vee (A \sim B)$$

Transitivity

$$(A \succ B) \wedge (B \succ C) \Rightarrow (A \succ C)$$

Continuity

$$A \succ B \succ C \Rightarrow \exists p [p, A; 1 - p, C] \sim B$$

Substitutability

$$A \sim B \Rightarrow [p, A; 1 - p, C] \sim [p, B; 1 - p, C]$$

Monotonicity

$$A \succ B \Rightarrow (p \geq q \Leftrightarrow [p, A; 1 - p, B] \succeq [q, A; 1 - q, B])$$

Assignment Project Exam Help

<https://tutorcs.com>

WeChat: cstutorcs



Theorem: Rational preferences imply behavior describable as maximization of expected utility



MEU Principle

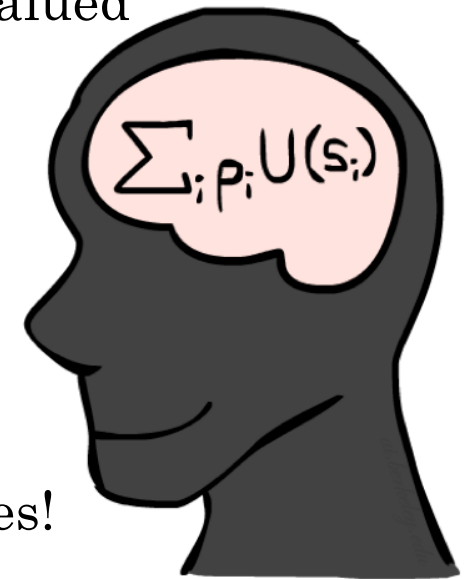
- Theorem [Ramsey, 1931; von Neumann & Morgenstern, 1944]
 - Given any preferences satisfying these constraints, there exists a real-valued function U such that:

$$U(A) \geq U(B) \Leftrightarrow A \succeq B$$

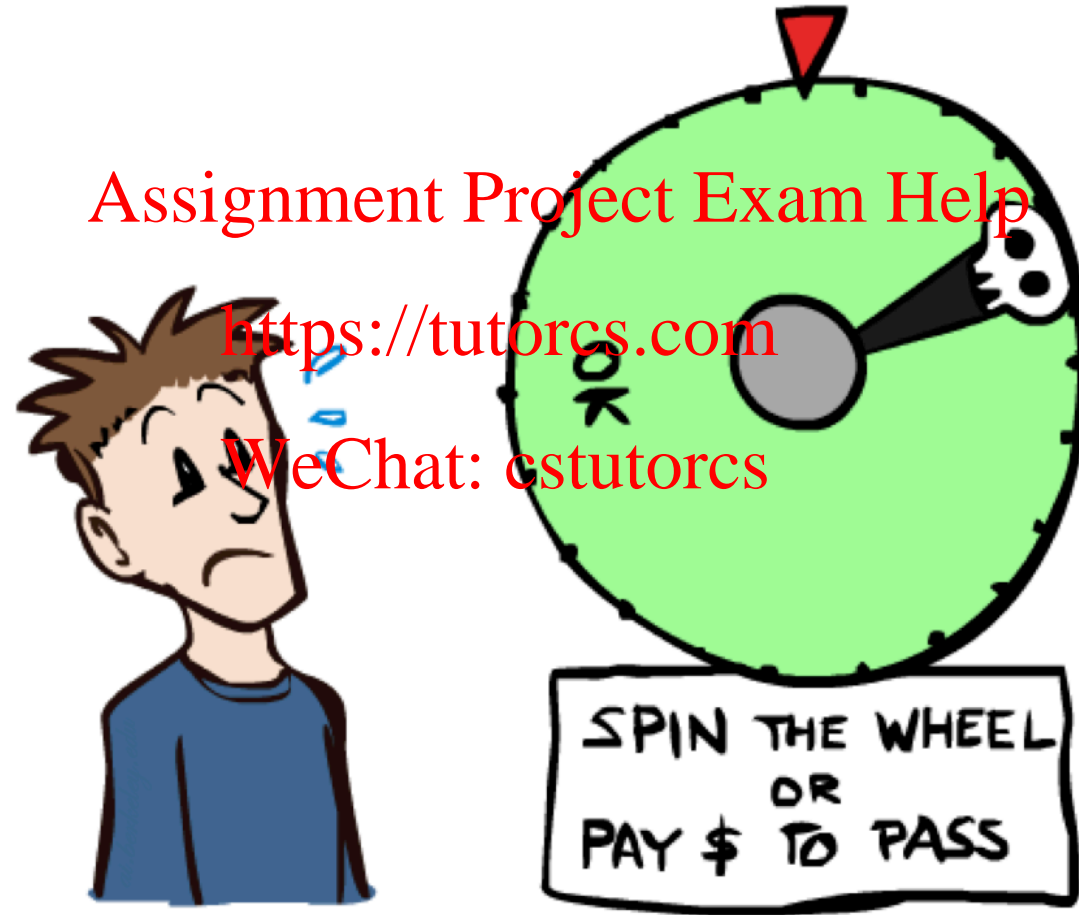
$$U([p_1, S_1; \dots; p_n, S_n]) = \sum_i p_i U(S_i)$$

WeChat: cstutorcs

- I.e. values assigned by U preserve preferences of both prizes and lotteries!
- Maximum expected utility (MEU) principle:
 - Choose the action that maximizes expected utility



Human Utilities



Human Utilities

- Utilities map states to real numbers. Which numbers?
- Standard approach to assessment (elicitation) of human utilities:
 - Compare a prize A to a standard lottery L_p between
 - “best possible prize” u_+ with probability p
 - “worst possible catastrophe” u_- with probability $1-p$
 - Adjust lottery probability p until indifference: $A \sim L_p$
 - Resulting p is a utility in $[0,1]$



Human Utilities: Example

- A person is given the choice between 2 scenarios:
 - Guaranteed scenario: the person receives \$50
 - Uncertain scenario: a coin is flipped to decide the person receive \$100 or not.
- Which choice would that person make?

Assignment Project Exam Help

<https://tutorcs.com>

WeChat: cstutorcs

Risk Aversion

- **Risk averse**: would accept the guaranteed payment of (less than) \$50 rather than take the gamble

Assignment Project Exam Help

- **Risk neutral**: indifferent between the bet and the guaranteed \$50 payment

<https://tutorcs.com>

WeChat: cstutorcs

- **Risk seeking**: would accept the bet even when the guaranteed payment is more than \$50

Prospect Theory: Utility Function

- **Risk aversion:** convexity

- Risk averse regarding gain
- Risk seeking regarding loss

- **Loss aversion**

- Losses are felt more strongly than gains

- **Endowment effect**

- We value things we own more highly
- Reference point: differentiate gains and loss

Assignment Project Exam Help

<https://tutorcs.com>

WeChat: cstutorcs

