Chapter 0: Preliminaries

Mathematics for Game Theory

Sets, functions, counting | Σ, series, argmax, derivative | Exponents, e and ln

Chain rule, partial derivative | Probability, Bayes, expectation | Mixed strategy, softmax

Payoff matrix | Best response | Quadratic equation | Indifference condition

Game tree | Growth rate: O and Θ

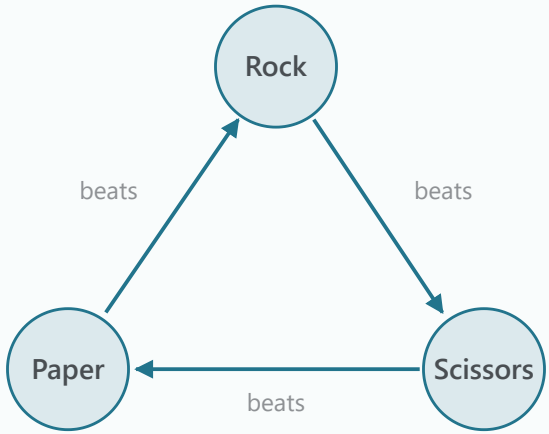
Sercan KÜLCÜ · Giresun University · Computer Engineering

© Sercan Külcü. All rights reserved. · Interactive HTML version of these slides: https://sercankulcu.github.io/files/agt/slides/Chapter_00_Preliminaries.html

Warm-up Game: Rock-Paper-Scissors

Two players simultaneously pick **Rock**, **Paper** or **Scissors**: Rock beats Scissors, Scissors beats Paper, Paper beats Rock; equal picks tie. The winner gains 1 point and the loser pays 1.

Rock-Paper-Scissors is often played against a computer that plays at random or learns the *frequency* or the *pattern* of your moves.



Think about it

Is there a way of playing that a learning computer cannot beat in the long run?

Someone always plays Rock. Once the opponent notices, how many points per round does this player lose on average?

Learning Outcomes

Notation

Read sets, Cartesian products and notation like $u_1(s_1, s_2)$.

Σ, max, argmax

Evaluate sums and geometric series; find a maximum and its argmax, with derivatives (chain rule, partials) for continuous choices.

Probability

Find probabilities by counting, multiplying and conditioning (Bayes); use U[0, 1].

Expected value

Expected payoff of a lottery or a mixed strategy; read exponents, ln and softmax.

Matrices and best responses

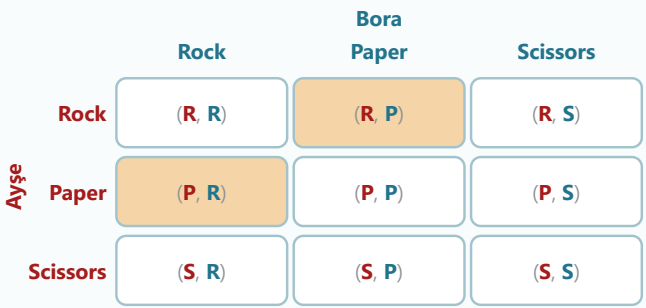
Read a pair (a, b) and mark best responses in a matrix.

Equations and trees

Solve quadratics and indifference conditions; solve a game tree from the leaves; read O and Θ.

Sets and Cartesian Products

A set is a collection of options; the **Cartesian product** of two sets is the set of all *ordered* pairs formed by taking one element from each.



Example

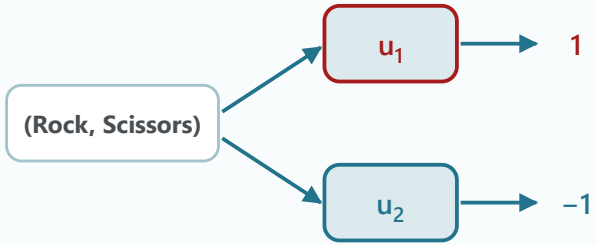
Ayşe's strategy set is S_1 = {Rock, Paper, Scissors}; Rock $\in S_1$ ("Rock is an element of S_1") and $|S_1| = 3$.

Bora's is also S_2 = {Rock, Paper, Scissors}. The possible outcomes form $S_1 \times S_2$: $|S_1 \times S_2| = 3 \cdot 3 =$ **9** pairs. Each pair is a **strategy profile**.

Common mistake: treating (Rock, Paper) and (Paper, Rock) as the same. Order matters: the first entry is Ayşe's choice, the second is Bora's. These are two different outcomes; Ayşe wins one and Bora wins the other (orange cells in the figure).

Function Notation

A function is a machine: it takes an input and returns *exactly one* output. The payoff function u_i takes a strategy profile and returns player i's payoff (a number).



input: profile → output: one number

Example (Rock-Paper-Scissors)

u_1(Rock, Scissors) = 1: if Ayşe plays Rock and Bora Scissors, Ayşe gains 1 point.

u_2(Rock, Scissors) = −1: in the *same* profile Bora pays 1 point.

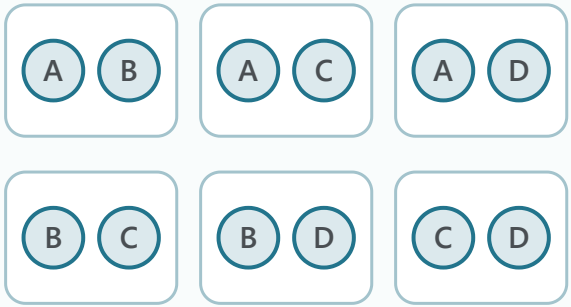
u_1(Paper, Paper) = 0: a tie.

Notation: $u_1 : S_1 \times S_2 \to \mathbb{R}$ ("from profiles to real numbers").

Common mistake: reading u_1(Rock, Paper) as "Paper wins". The value of a function is a number: u_1(Rock, Paper) = −1. The subscript says whose payoff we look at; u_1 and u_2 give different numbers for the same profile.

Counting: Product Rule, n! and C(n, k)

For successive choices, the numbers of options **multiply**. n distinct objects can be ordered in n! = n · (n − 1) · … · 1 ways. The number of ways to choose k of n objects, order ignored, is **C(n, k) = n! / (k! (n − k)!)**.



C(4, 2) = 6 choices; no order: {A, B} = {B, A}

Example

Three players with 2, 3 and 4 strategies: 2 · 3 · 4 = **24** strategy profiles.

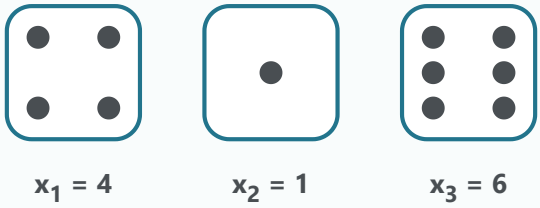
Three people can stand in a line in 3! = 3 · 2 · 1 = **6** ways.

A guard protects 2 of 4 targets: C(4, 2) = 4!/(2! · 2!) = **6** pure strategies. 5 of 20 targets: C(20, 5) = **15,504**.

Common mistake: counting order in a selection. "First one target, then another" gives 4 · 3 = 12 ordered pairs; but {A, B} and {B, A} are the same guard plan, so 12 / 2! = 6.

The Summation Symbol Σ

Σ means "add them all up". The index below says where to start, the number above where to stop: $\sum_{i=1}^{3} x_i = x_1 + x_2 + x_3$.



$x_1 = 4$ $x_2 = 1$ $x_3 = 6$

Example: three dice rolls

$\sum_{i=1}^{3} x_i = 4 + 1 + 6 = \mathbf{11}$

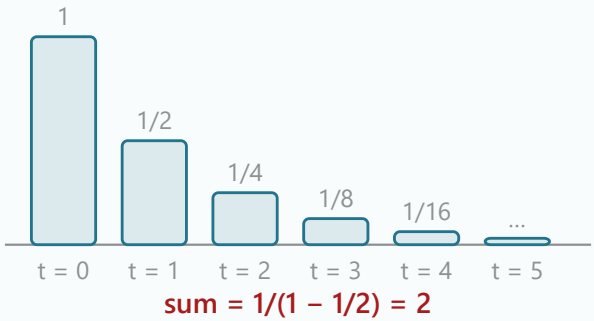
On a fair die each face has probability 1/6: $\sum_{i=1}^{6} p_i = 6 \cdot 1/6 = \mathbf{1}$. Probabilities always sum to 1.

An expected value is a sum as well: $\sum_{i=1}^{n} x_i \cdot p_i$.

Common mistake: pulling the square outside the sum. $(\Sigma\, x_i)^2$ and $\Sigma\, x_i^2$ differ: $(4 + 1 + 6)^2 = 121$, but $4^2 + 1^2 + 6^2 = 53$. Evaluate the term for each i first, then add.

Geometric Series and the Discount Factor δ

A game is repeated every round; a payoff t rounds ahead is multiplied by δ^t ($0 \le \delta < 1$): a payoff of 1 today is worth 1, one round later δ, two rounds later δ^2. Earning 1 every round gives $1 + \delta + \delta^2 + \ldots = \mathbf{1/(1-\delta)}$.



Why?

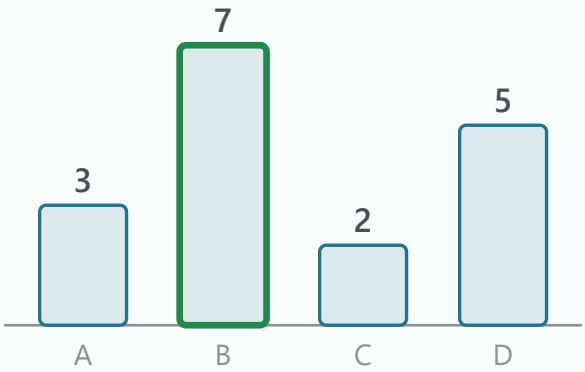
If $S = 1 + \delta + \delta^2 + \ldots$, then $\delta S = \delta + \delta^2 + \ldots$; subtracting gives $S - \delta S = 1 \Rightarrow S = 1/(1-\delta)$.

Example: δ = 0.9 and a payoff of 3 per round: 3/(1 − 0.9) = **30**. δ = 1/2 and 1 per round: 1 + 1/2 + 1/4 + … = **2**.

Common mistake: forgetting the first term: a sum starting "from tomorrow" is $\delta + \delta^2 + \ldots = \delta/(1-\delta)$, not $1/(1-\delta)$. For δ = 1 the formula fails; the sum grows without bound.

max and argmax

max answers "what is the largest value?"; **argmax** answers "*which option* gives the largest value?". In game theory a player chooses an action, not a value, so argmax is used more often.



Example: picking a card

You pick one of four face-down cards; their prizes are A: 3, B: 7, C: 2, D: 5 chips.

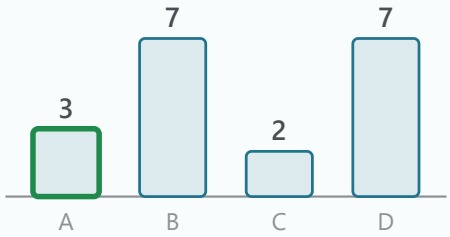
max{3, 7, 2, 5} = **7** (a number)

argmax = **B** (an option): "pick B".

Common mistake: writing the value instead of the argmax ("argmax = 7"), or the reverse. The second trap is ties: with prizes 3, 7, 2, 7 the argmax is not one element but the set {B, D}; both are best choices.

Trace: Finding the argmax with a Loop

```
action = ["A", "B", "C", "D"]
payoff = [3, 7, 2, 7]
best = 0              # index of the best
for i in range(1, 4):
    if payoff[i] > payoff[best]:
        best = i
print(payoff[best], action[best])
```



Start: take the first card (A, 3 chips) as the best.

Step 1 / 6

VARIABLES

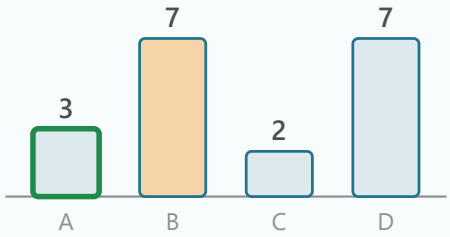
`i = –` `best = 0` `payoff[best] = 3`

OUTPUT

Orange bar: the card being examined. Green frame: the best card so far.

Trace: Finding the argmax with a Loop

```
1 action = ["A", "B", "C", "D"]
2 payoff = [3, 7, 2, 7]
3 best = 0              # index of the best
▶ for i in range(1, 4):
▶     if payoff[i] > payoff[best]:
6         best = i
7 print(payoff[best], action[best])
```



i = 1: payoff[1] = 7 > 3? **Yes**.

Step 2 / 6

VARIABLES

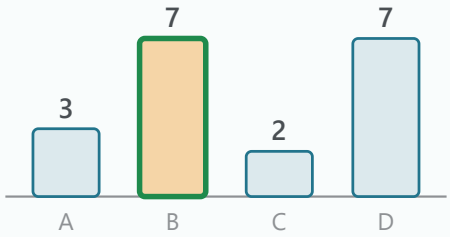
`i = 1` `best = 0` `payoff[best] = 3`

OUTPUT

Orange bar: the card being examined. Green frame: the best card so far.

Trace: Finding the argmax with a Loop

```
1 action = ["A", "B", "C", "D"]
2 payoff = [3, 7, 2, 7]
3 best = 0              # index of the best
4 for i in range(1, 4):
5     if payoff[i] > payoff[best]:
          best = i
7 print(payoff[best], action[best])
```



best = 1: new best is B (7 chips).

Step 3 / 6

VARIABLES

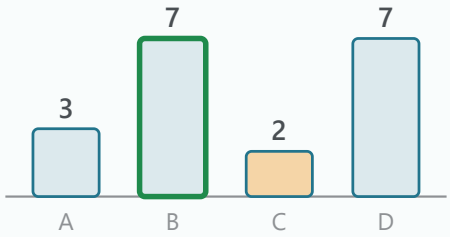
`i = 1` `best = 1` `payoff[best] = 7`

OUTPUT

Orange bar: the card being examined. Green frame: the best card so far.

Trace: Finding the argmax with a Loop

```
1 action = ["A", "B", "C", "D"]
2 payoff = [3, 7, 2, 7]
3 best = 0              # index of the best
▶ for i in range(1, 4):
▶     if payoff[i] > payoff[best]:
6         best = i
7 print(payoff[best], action[best])
```



i = 2: payoff[2] = 2 > 7? No; the best stays.

Step 4 / 6

VARIABLES

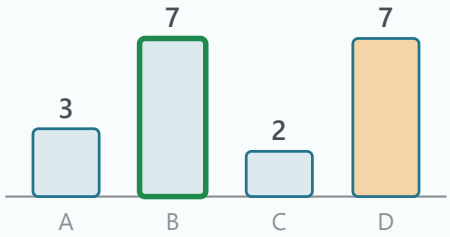
`i = 2` `best = 1` `payoff[best] = 7`

OUTPUT

Orange bar: the card being examined. Green frame: the best card so far.

Trace: Finding the argmax with a Loop

```
1 action = ["A", "B", "C", "D"]
2 payoff = [3, 7, 2, 7]
3 best = 0              # index of the best
▶ for i in range(1, 4):
▶     if payoff[i] > payoff[best]:
6         best = i
7 print(payoff[best], action[best])
```



i = 3: 7 > 7? **No**: with strict ">" a tie keeps the first one found.

Step 5 / 6

VARIABLES

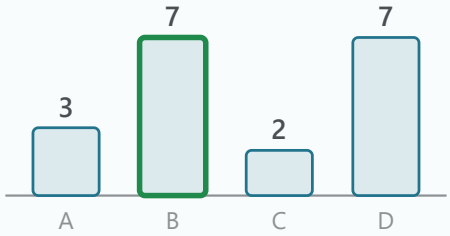
`i = 3` `best = 1` `payoff[best] = 7`

OUTPUT

Orange bar: the card being examined. Green frame: the best card so far.

Trace: Finding the argmax with a Loop

```
1 action = ["A", "B", "C", "D"]
2 payoff = [3, 7, 2, 7]
3 best = 0              # index of the best
4 for i in range(1, 4):
5     if payoff[i] > payoff[best]:
6         best = i
▶ print(payoff[best], action[best])
```



max = **7** and the code returns **B** as the argmax; since D also gives 7, the argmax set is really {B, D}.

Step 6 / 6

VARIABLES

`i = 3` `best = 1` `payoff[best] = 7`

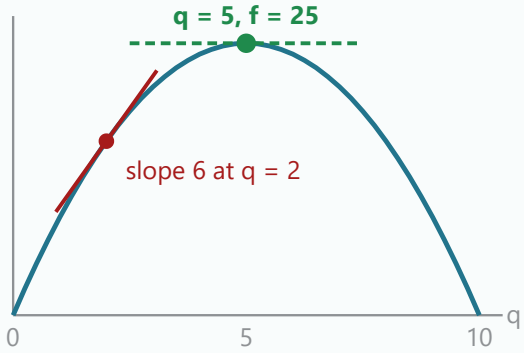
OUTPUT

```
7 B
```

Orange bar: the card being examined. Green frame: the best card so far.

Maximizing a Function: Derivative = 0

When the choice is continuous (a quantity q, say), the argmax cannot be read off a table. At the **top of the payoff curve the slope is zero**: positive to the left of the peak, negative to the right. The slope is the **derivative** $f'(q)$.



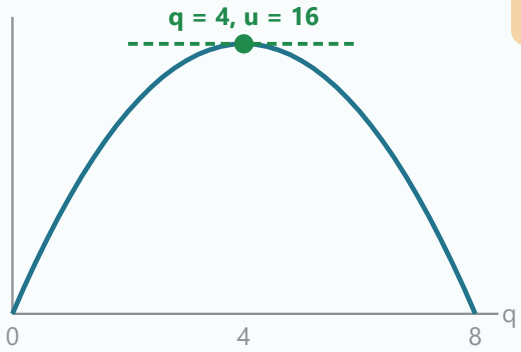
Rules you need

$(q^k)' = k \cdot q^{k-1}$ · derivative of a constant is 0 · $(c \cdot f)' = c \cdot f'$ · derivative of a sum is the sum of derivatives.

Example: $f(q) = q(10 - q) = 10q - q^2$. $f'(q) = 10 - 2q = 0 \Rightarrow$ **q = 5**, $f(5) =$ **25**. The second derivative f'' (derivative of the derivative) tells how the slope changes: $f'' = -2 < 0$, the slope decreases, so the point is a peak.

Common mistake: never checking the point where $f' = 0$. It may be a valley; $f'' < 0$ or the neighbors ($f(4) = f(6) = 24 < 25$) show it is a peak. If the choice is restricted to an interval, check the endpoints too.

Trace: Best Quantity by Derivative

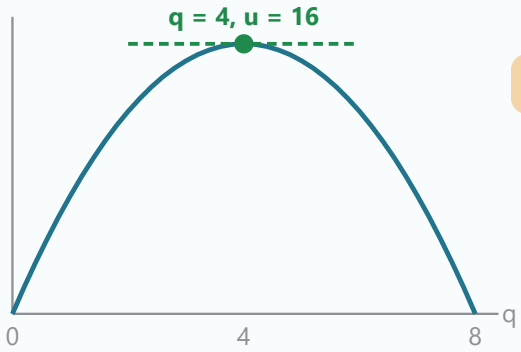


Price = 12 − (q + r); q: the firm's quantity, r: the rival's; no cost. The plot is for r = 4.

Profit = quantity × price: u(q) = q · (12 − q − r).

Step 1 / 6

Trace: Best Quantity by Derivative



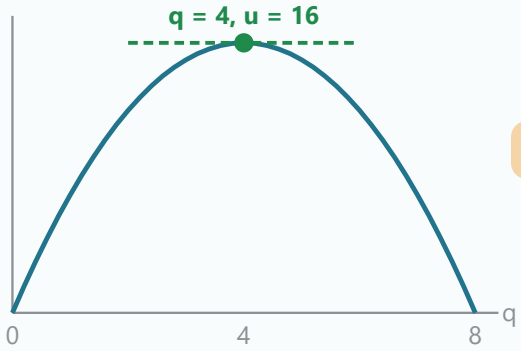
Price = 12 − (q + r); q: the firm's quantity, r: the rival's; no cost. The plot is for r = 4.

Profit = quantity × price: u(q) = q · (12 − q − r).

If the rival produces r = 4: u(q) = q(8 − q) = **$8q - q^2$**.

Step 2 / 6

Trace: Best Quantity by Derivative



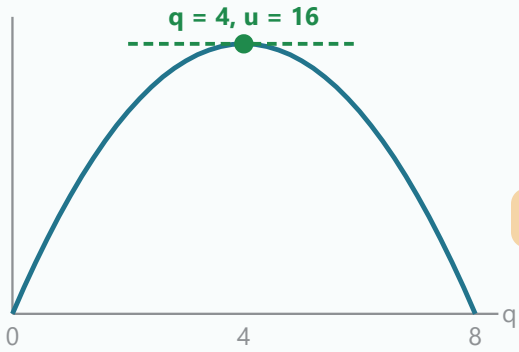
Price = 12 − (q + r); q: the firm's quantity, r: the rival's; no cost. The plot is for r = 4.

Profit = quantity × price: u(q) = q · (12 − q − r).

If the rival produces r = 4: u(q) = q(8 − q) = **$8q - q^2$**.

Derivative: u′(q) = 8 − 2q.

Trace: Best Quantity by Derivative



Price = 12 − (q + r); q: the firm's quantity, r: the rival's; no cost. The plot is for r = 4.

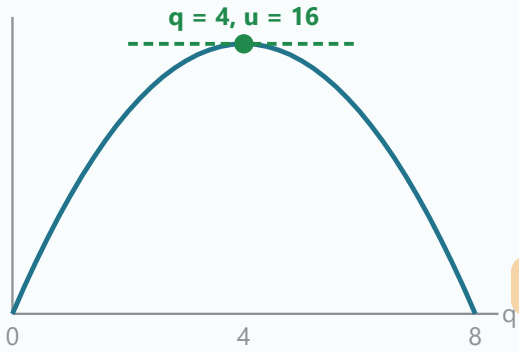
Profit = quantity × price: u(q) = q · (12 − q − r).

If the rival produces r = 4: u(q) = q(8 − q) = **$8q - q^2$**.

Derivative: u′(q) = 8 − 2q.

Set it to zero: 8 − 2q = 0 ⇒ **q = 4**; profit u(4) = 4 · 4 = **16**.

Trace: Best Quantity by Derivative



Price = 12 − (q + r); q: the firm's quantity, r: the rival's; no cost. The plot is for r = 4.

Profit = quantity × price: u(q) = q · (12 − q − r).

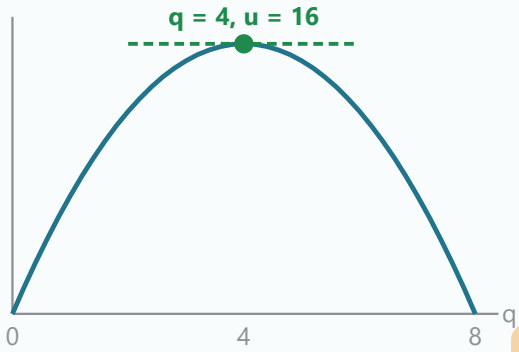
If the rival produces r = 4: u(q) = q(8 − q) = **$8q - q^2$**.

Derivative: u′(q) = 8 − 2q.

Set it to zero: 8 − 2q = 0 ⇒ **q = 4**; profit u(4) = 4 · 4 = **16**.

Check: u″ = −2 < 0; neighbors u(3) = u(5) = 15 < 16 ✓

Trace: Best Quantity by Derivative



Price = 12 − (q + r); q: the firm's quantity, r: the rival's; no cost. The plot is for r = 4.

Profit = quantity × price: u(q) = q · (12 − q − r).

If the rival produces r = 4: u(q) = q(8 − q) = **$8q - q^2$**.

Derivative: u′(q) = 8 − 2q.

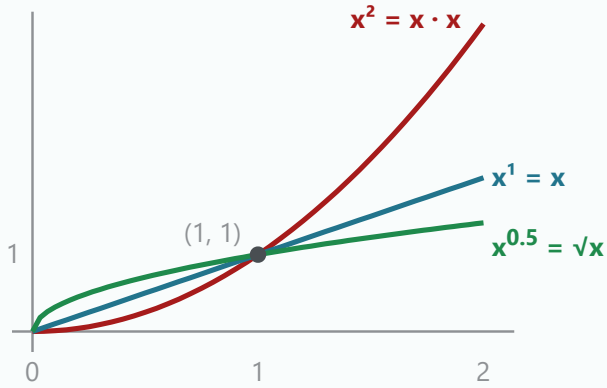
Set it to zero: 8 − 2q = 0 ⇒ **q = 4**; profit u(4) = 4 · 4 = **16**.

Check: u″ = −2 < 0; neighbors u(3) = u(5) = 15 < 16 ✓

General r: u′(q) = 12 − 2q − r = 0 ⇒ **q = (12 − r)/2**. Quantities cannot be negative: if r ≥ 12 it is best not to produce, so the full formula is **q*(r) = max{0, (12 − r)/2}**. Giving the best quantity for every rival quantity, it is a best response in continuous form.

Exponent Rules and Fractional Exponents

For an integer a, x^a is x multiplied by itself a times. Rules: $x^a \cdot x^b = x^{a+b}$, $(x^a)^b = x^{a \cdot b}$, $\mathbf{x^0 = 1}$, $x^{-a} = 1/x^a$, $x^{1/2} = \sqrt{x}$. Fractional exponents (such as $x^{0.88}$) obey the same rules.



Example

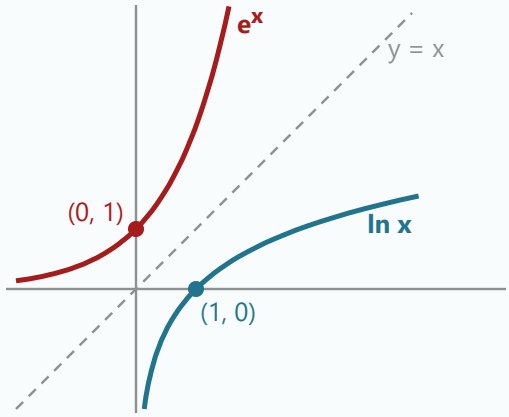
$\delta^2 \cdot \delta^3 = \delta^5$; $2^{-1} = 1/2$; $9^{1/2} = \sqrt{9} = 3$; $5^0 = 1$.

Utility $x^{0.88}$: $100^{0.88} \approx 57.5$ and $200^{0.88} \approx 105.9$. Doubling the money multiplies utility by $2^{0.88} \approx 1.84$: with an exponent below 1, growth slows down.

Common mistake: adding exponents in a sum: $2^2 + 2^3 = 4 + 8 = 12$, but $2^5 = 32$. Exponents add only in a *product* with the same base. Another: thinking $x^0 = 0$; in fact $x^0 = 1$ ($x \neq 0$).

The Number e and the Natural Logarithm ln

$e \approx 2.718$ is a special base; e^x is positive for every x. **ln** (natural logarithm) undoes e^x: $\ln(e^x) = x$ and $e^{\ln y} = y$ for $y > 0$. Rules: $\ln(a \cdot b) = \ln a + \ln b$, $\ln(a^k) = k \cdot \ln a$, $\ln 1 = 0$. Derivatives: $(e^x)' = e^x$, **$(\ln x)' = 1/x$**.



Example

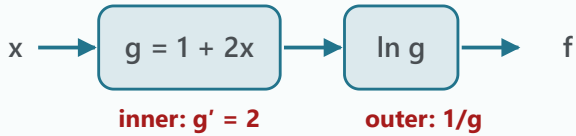
$\ln 8 = \ln 2^3 = 3 \cdot \ln 2 \approx 3 \cdot 0.693 =$ **2.079**; back again: $e^{2.079} \approx 8$.

Utility $u(x) = \ln x$: wealth going from 10 to 20 raises utility by $\ln 20 - \ln 10 = \ln 2 \approx 0.69$; going from 20 to 40 again adds 0.69. $u'(x) = 1/x$: the richer you are, the less an extra lira is worth.

Common mistake: thinking $\ln(a + b) = \ln a + \ln b$: $\ln(2 + 6) = \ln 8 \approx 2.08$, but $\ln 2 + \ln 6 = \ln 12 \approx 2.48$. The log of a product splits into a sum, not the log of a sum. ln 0 and ln of negative numbers are undefined.

The Product Rule and the Chain Rule

Product: $(f \cdot g)' = f' \cdot g + f \cdot g'$. **Chain** (nested functions): $[f(g(x))]' = f'(g(x)) \cdot g'(x)$; differentiate the outer function first (leaving the inner one as is), then multiply by the derivative of the inner one.



derivative = outer derivative × inner derivative

$f' = 1/(1 + 2x) \cdot 2$

Example

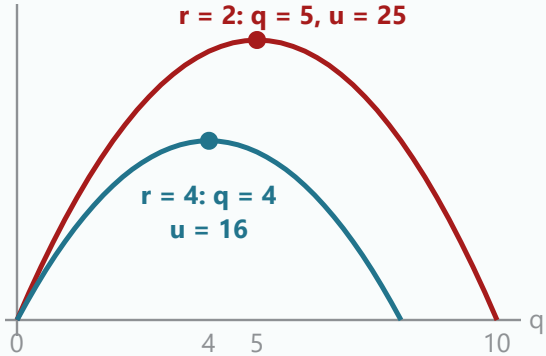
Profit $u(q) = q \cdot (12 - q - r)$: $f = q$, $g = 12 - q - r$. $u' = 1 \cdot (12 - q - r) + q \cdot (-1) =$ **$12 - 2q - r$**; expanding the product first gives the same result.

Chain: $[\ln(1 + 2x)]' = 2/(1 + 2x)$; $(e^{-x})' = -e^{-x}$; $[(1 - p)^2]' = 2(1 - p) \cdot (-1) = -2(1 - p)$.

Common mistake: forgetting the inner derivative: writing $(e^{-x})' = e^{-x}$ (correct: $-e^{-x}$). In a product, $(f \cdot g)' = f' \cdot g'$ is wrong: $(q \cdot q)' = 2q$, but $f' \cdot g' = 1 \cdot 1 = 1$.

Partial Derivatives: Hold the Others Fixed

If a function depends on several variables (such as u(q, r)), **∂u/∂q** (partial derivative) is the derivative with respect to q alone, treating r as a constant. In a game each player chooses only its own variable, so it sets the partial derivative with respect to that variable to zero.



one curve per r; ∂u/∂q is its slope

Example

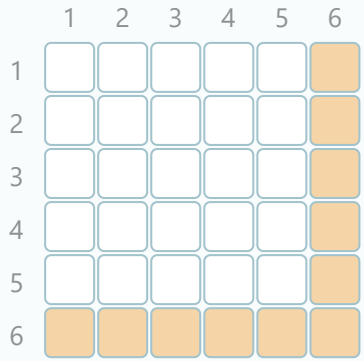
$u_1(q, r) = q(12 - q - r) = 12q - q^2 - qr$: $\partial u_1/\partial q =$ **$12 - 2q - r$**, $\partial u_1/\partial r = -q$.

The rival's profit is $u_2 = r(12 - q - r)$: $\partial u_2/\partial r = 12 - q - 2r$. Both zero: $12 - 2q - r = 0$ and $12 - q - 2r = 0 \Rightarrow$ **$q = r = 4$**.

Common mistake: setting the fixed variable to zero, or differentiating it too: $\partial(q \cdot r)/\partial q = r$, not 0 or $1 + r$. r is fixed, but it is not zero.

Probability by Counting

If outcomes are equally likely, P(event) = favorable outcomes / all outcomes. For independent events (rolls that do not affect each other) probabilities **multiply**.



Examples

Even number on a die: {2, 4, 6} → 3/6 = **1/2**.

Two heads in two coin tosses: 1/2 · 1/2 = **1/4**.

At least one 6 with two dice: use the complement. No 6 at all has probability (5/6)·(5/6) = 25/36, so 1 − 25/36 = **11/36**. The table has 11 orange cells.

Common mistake: saying 1/6 + 1/6 = 12/36 for "at least one 6 with two dice". This sum counts (6, 6) twice; the correct answer is 11/36. Probabilities add only when the events cannot happen together.

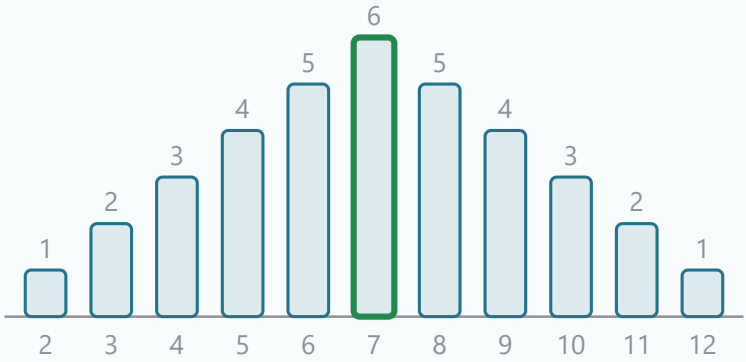
Probability Distributions

A probability distribution is a table assigning a probability to each possible outcome. Every probability lies between 0 and 1, and they sum to 1.

Example: the sum of two dice

There are 36 equally likely outcomes. The pairs summing to 7 are (1,6), (2,5), (3,4), (4,3), (5,2), (6,1): P(7) = **6/36**. A sum of 2 only comes from (1,1): P(2) = 1/36.

Check: 1 + 2 + 3 + 4 + 5 + 6 + 5 + 4 + 3 + 2 + 1 = 36, so the total probability is 36/36 = 1.



Common mistake: treating every sum as equally likely ("11 possible sums, so 1/11 each"). The outcomes (pairs of dice) are equally likely, not the sums. The numbers above the bars count how many of the 36 outcomes give that sum.

Conditional Probability: Information Changes Probabilities

Once we learn that event B occurred, only the outcomes in B remain. A's share of this smaller world is the **conditional probability**:

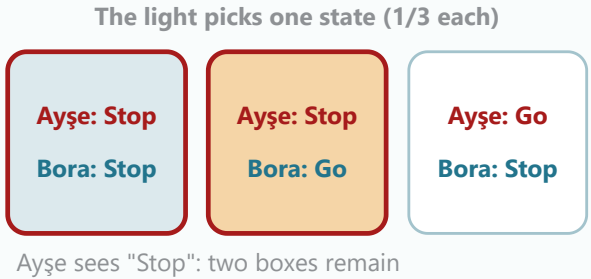
$$P(A \mid B) = P(A \cap B) / P(B)$$

A ∩ B: both A and B occur.

Example: a traffic light

The light picks one of three states with probability 1/3 each and shows every driver only their own signal. Ayşe sees "Stop". P(Ayşe Stop) = 2/3, P(Ayşe Stop ∩ Bora Go) = 1/3.

P(Bora Go | Ayşe Stop) = (1/3) / (2/3) = **1/2**.



P(Bora Go | Ayşe Stop) = (1/3)/(2/3) = 1/2

Common mistake: ignoring the information ("Bora sees Go: 1/3") or reversing the condition. P(A | B) and P(B | A) usually differ: P(Ayşe Stop | Bora Go) = 1, but P(Bora Go | Ayşe Stop) = 1/2.

Bayes' Rule: Updating Beliefs after an Observation

We start with a belief (the **prior**, P(A)); after seeing a signal we update it (the **posterior**, P(A | B)):

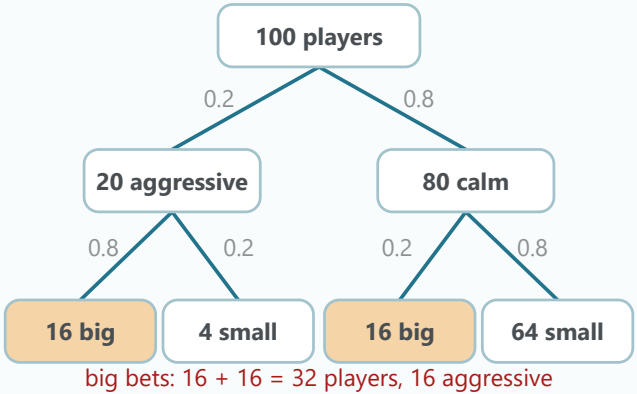
$$P(A \mid B) = P(B \mid A) \cdot P(A) / P(B)$$

Example: a big bet

20% of players are aggressive. An aggressive player bets big with probability 0.8, a calm one with probability 0.2. We see a big bet.

$P(\text{big}) = 0.8 \cdot 0.2 + 0.2 \cdot 0.8 = 0.32 \Rightarrow P(\text{aggressive} \mid \text{big}) = 0.16 / 0.32 =$ **1/2**.

Counting with 100 players gives the same answer: 16 of the 32 big bettors are aggressive.

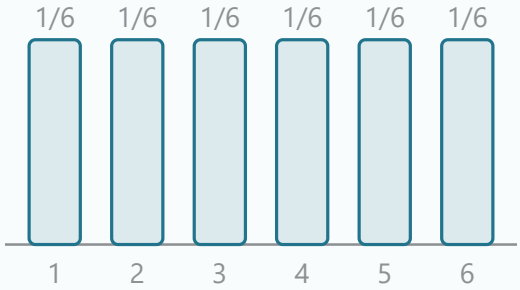


Common mistake: forgetting the prior: "Aggressive players bet big 80% of the time, so this player is aggressive with probability 80%." Calm players are four times as many, so their rare big bets make up half of all big bets.

Expected Value

X is a number that depends on chance (e.g. the face of a die). Its expected value is the **average per round** X would have if the experiment were repeated many times: multiply each value x of X by its probability and add.

$$E[X] = \sum_{x} x \cdot P(X = x)$$



Example: a fair die

E[X] = 1·(1/6) + 2·(1/6) + 3·(1/6) + 4·(1/6) + 5·(1/6) + 6·(1/6)

= 21/6 = **3.5**

Common mistake: taking the expected value to be "the most likely outcome" or "the outcome that will occur". A die never shows 3.5; 3.5 is only the average over many rolls. A lottery paying 0 or 100 may have expected value 30, yet it never pays 30.

Trace: Expected Value of a Lottery

```
prize = [100, 20, 0]          # TL
prob = [0.05, 0.25, 0.70]
E = 0
for i in range(3):
    E = E + prize[i] * prob[i]
print("E =", E)
print("net =", E - 12)  # ticket 12 TL
```

	prize	probability	product
i = 0	100 TL	0.05	...
i = 1	20 TL	0.25	...
i = 2	0 TL	0.70	...

E = 0

Three prizes and their probabilities; total probability 0.05 + 0.25 + 0.70 = 1. Accumulator E = 0.

Step 1 / 6

VARIABLES

`i = –` `E = 0`

OUTPUT

Trace: Expected Value of a Lottery

```
1 prize = [100, 20, 0]          # TL
2 prob = [0.05, 0.25, 0.70]
3 E = 0
for i in range(3):
    E = E + prize[i] * prob[i]
6 print("E =", E)
7 print("net =", E - 12)  # ticket 12 TL
```

	prize	probability	product
i = 0	100 TL	0.05	5
i = 1	20 TL	0.25	...
i = 2	0 TL	0.70	...

E = 5

Add 100 · 0.05 = 5: E = 5.

Step 2 / 6

VARIABLES

`i = 0` `prize[i] = 100` `prob[i] = 0.05` `E = 5.0`

OUTPUT

Trace: Expected Value of a Lottery

```
1 prize = [100, 20, 0]          # TL
2 prob = [0.05, 0.25, 0.70]
3 E = 0
for i in range(3):
    E = E + prize[i] * prob[i]
6 print("E =", E)
7 print("net =", E - 12)  # ticket 12 TL
```

	prize	probability	product
i = 0	100 TL	0.05	5
i = 1	20 TL	0.25	5
i = 2	0 TL	0.70	...

E = 10

Add 20 · 0.25 = 5: E = 10.

Step 3 / 6

VARIABLES

`i = 1` `prize[i] = 20` `prob[i] = 0.25` `E = 10.0`

OUTPUT

Trace: Expected Value of a Lottery

```
1 prize = [100, 20, 0]          # TL
2 prob = [0.05, 0.25, 0.70]
3 E = 0
for i in range(3):
    E = E + prize[i] * prob[i]
6 print("E =", E)
7 print("net =", E - 12)  # ticket 12 TL
```

	prize	probability	product
i = 0	100 TL	0.05	5
i = 1	20 TL	0.25	5
i = 2	0 TL	0.70	0

E = 10

0 · 0.70 = 0: a losing ticket does not change E, but its probability still counts.

Step 4 / 6

VARIABLES

`i = 2` `prize[i] = 0` `prob[i] = 0.7` `E = 10.0`

OUTPUT



Trace: Expected Value of a Lottery

```
1 prize = [100, 20, 0]          # TL
2 prob = [0.05, 0.25, 0.70]
3 E = 0
4 for i in range(3):
5     E = E + prize[i] * prob[i]
▶ print("E =", E)
7 print("net =", E - 12)  # ticket 12 TL
```

	prize	probability	product
i = 0	100 TL	0.05	5
i = 1	20 TL	0.25	5
i = 2	0 TL	0.70	0

E = 10

Expected prize per ticket: **10 TL**.

Step 5 / 6

VARIABLES

`i = 2` `E = 10.0`

OUTPUT

```
E = 10.0
```

Trace: Expected Value of a Lottery

```
1 prize = [100, 20, 0]          # TL
2 prob = [0.05, 0.25, 0.70]
3 E = 0
4 for i in range(3):
5     E = E + prize[i] * prob[i]
6 print("E =", E)
▶ print("net =", E - 12)  # ticket 12 TL
```

	prize	probability	product
i = 0	100 TL	0.05	5
i = 1	20 TL	0.25	5
i = 2	0 TL	0.70	0

E = 10

The ticket costs 12 TL: net expected payoff 10 − 12 = **−2 TL**. Someone who plays many times loses 2 TL per ticket on average.

Step 6 / 6

VARIABLES

`i = 2` `E = 10.0`

OUTPUT

```
E = 10.0
net = -2.0
```

Example: Is This Dice Game Worth Playing?

A fairground game: you pay **10 TL** and roll a die. A **6** pays 40 TL, a **5** pays 10 TL; any other number pays nothing.

Die	Prize x	P(x)	x · P(x)
6	40	1/6	40/6
5	10	1/6	10/6
1, 2, 3, 4	0	4/6	0
Total		**6/6 = 1**	**50/6 ≈ 8.33**

Verdict

Expected prize ≈ 8.33 TL against a 10 TL fee: net ≈ **−1.67 TL** per game. Ignoring risk, playing does not pay.

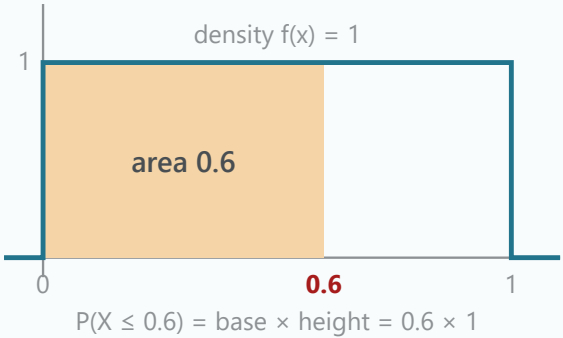
What would make it fair?

Let the prize for a 6 be y: (y + 10)/6 = 10 ⇒ y = **50 TL**. This is an equation in one unknown.

Common mistake: deciding "I win with probability 2/6, not bad". The winning probability alone is not enough; the size of the prizes and the fee count too. Decide by the net expected value.

The Continuous Uniform Distribution U[0, 1]

$X \sim U[0, 1]$ reads "X is uniformly distributed between 0 and 1": every value in the interval is equally likely. The density $f(x) = 1$ is the height of the graph; probability is the area under it. With infinitely many values, a single point has probability 0; an **interval** has probability equal to its length: $P(a \le X \le b) = b - a$, in particular $\mathbf{P(X \le x) = x}$.



Example: a rival's value

A rival's value is U[0, 1]. A bid of 0.6 beats it with probability $P(X \le 0.6) = \mathbf{0.6}$.

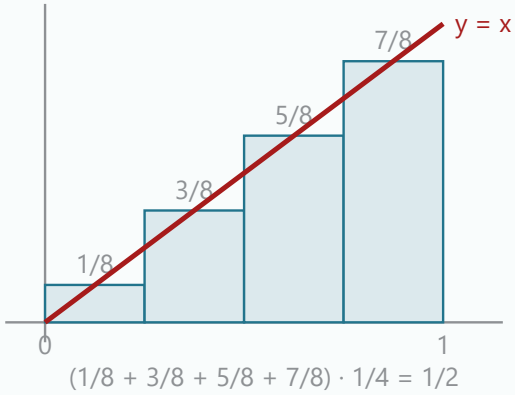
Two independent rivals: beating both has probability $0.6 \cdot 0.6 = \mathbf{0.36}$.

$P(0.2 \le X \le 0.5) = 0.3$. The mean is $E[X] = \mathbf{1/2}$.

Common mistake: taking the height of the density (1) for a probability, or giving $P(X = 0.5)$ a positive value. In a continuous distribution a probability is an *area*: base (interval) × height.

Integrals: Area under a Curve

$\int_a^b f(x)\,dx$ is the **area** under the graph of f between a and b: the continuous version of Σ. Cut the area into thin rectangles and add them up; as they get thinner, the sum approaches the integral.



Example

$\int_0^1 x\,dx$ = area of the triangle with base 1 and height 1 = **1/2**. In general $\int_0^b x\,dx = b^2/2$.

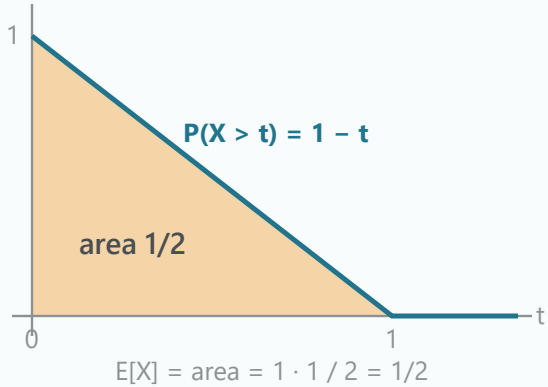
For U[0, 1], probability and expectation are areas: $P(X \le b) = \int_0^b 1\,dx = b$, $E[X] = \int_0^1 x \cdot 1\,dx =$ **1/2**.

Common mistake: taking the integral for the height of the curve at a point. $\int_0^1 x\,dx$ is not the height at x = 1 (which is 1) but the accumulated area 1/2; $E[X] = \int x \cdot f(x)\,dx$ is the continuous counterpart of $E[X] = \Sigma\, x \cdot P(x)$.

Expected Value and Tail Probabilities

$P(X > t)$, the probability that X exceeds t, is a **tail probability**. If $X \geq 0$, the expected value is the area under the tail probability:

$$E[X] = \int_0^\infty P(X > t)\, dt$$



Example

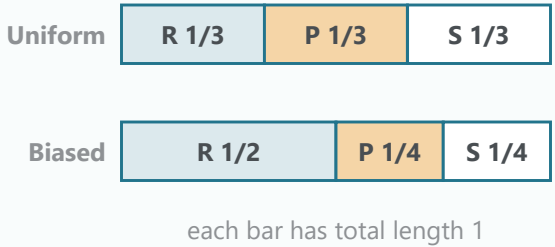
$X \sim U[0, 1]$: $P(X > t) = 1 - t$ $(0 \leq t \leq 1)$, and 0 for $t > 1$. $E[X] = \int_0^1 (1 - t)\, dt$ = area of the triangle with base 1 and height 1 = **1/2** ✓

For integer X the sum form is $E[X] = P(X \geq 1) + P(X \geq 2) + \ldots$ For a die, $6/6 + 5/6 + \ldots + 1/6 = 21/6 =$ **3.5** ✓

Common mistake: taking the area under the density for the expected value. For U[0, 1] the area under the density is 1 (total probability); $E[X] = 1/2$ is the area under the curve $P(X > t)$. The formula holds only for $X \geq 0$.

The Idea of a Mixed Strategy

A **mixed strategy** (randomized strategy) is a probability distribution a player places over its own actions: "let a die or a coin decide my move."



Example

Roll a die: 1–2 → Rock, 3–4 → Paper, 5–6 → Scissors. This is the mixed strategy (1/3, 1/3, 1/3).

Toss a coin: heads, Rock; tails, toss again for Paper or Scissors: (1/2, 1/4, 1/4).

Playing one action for sure is a **pure strategy**: (1, 0, 0) = always Rock.

Common mistake: taking a mixed strategy to mean "cycle through the moves" (Rock, Paper, Scissors, Rock, ...). Such a pattern is predictable and easy to beat. Under a mixed strategy a *single* pure move is played each round; which one is chosen at random.

Expected Payoff against a Mixed Strategy

Bora plays (1/2 Rock, 1/4 Paper, 1/4 Scissors). The expected payoff of each of Ayşe's pure moves: multiply the row by Bora's probabilities and add.

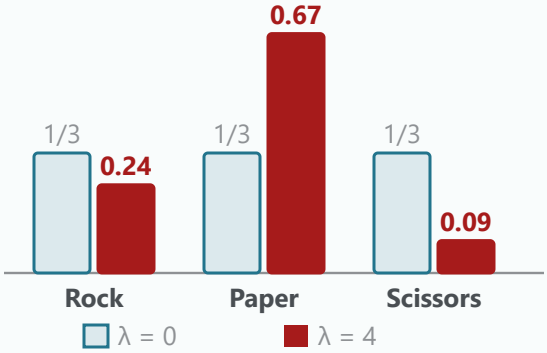
Ayşe ↓ / Bora →	Rock (1/2)	Paper (1/4)	Scissors (1/4)	Expected payoff
Rock	0	−1	+1	0 − 1/4 + 1/4 = **0**
Paper	+1	0	−1	1/2 + 0 − 1/4 = **1/4**
Scissors	−1	+1	0	−1/2 + 1/4 + 0 = **−1/4**

argmax = **Paper**: against a Rock-heavy opponent, Paper earns 1/4 point per round on average. Expected value + argmax together give the "best response".

Common mistake: looking only at the opponent's most frequent move. Had Bora played (0.4 Rock, 0.25 Paper, 0.35 Scissors), Paper, which beats Rock, would earn 0.4 − 0.35 = 0.05, while Rock earns −0.25 + 0.35 = 0.10. Compute every row's expected value before deciding.

Exponential Weights: Logit and Softmax

A smooth way to assign probabilities to options with values $v_1, \ldots, v_k$ is **softmax**: $P(i) = e^{\lambda v_i} / \Sigma_j e^{\lambda v_j}$. With two options it is called **logit**: $P(1) = 1 / (1 + e^{-\lambda(v_1 - v_2)})$. $\lambda \geq 0$ is the sensitivity: at $\lambda = 0$ all are equal; as λ grows, probability piles up on the argmax.



Example

Expected payoffs Rock 0, Paper 1/4, Scissors −1/4 and $\lambda = 4$: exponents 0, 1, −1.

$e^0 = 1$, $e^1 \approx 2.718$, $e^{-1} \approx 0.368$; sum ≈ 4.086. $P \approx$ (**0.24**, **0.67**, **0.09**).

Common mistake: dividing the values by their sum ($v_i / \Sigma v_j$). Values can be zero or negative: 0 + 1/4 − 1/4 = 0 cannot be a divisor. e^x is used because it is always positive; since $e^a/e^b = e^{a-b}$, adding the same number to all values leaves the probabilities unchanged.

PAYOFF MATRICES

Reading a Payoff Matrix

The **row player** picks a row, the **column player** a column; the cell where they meet is the outcome. A cell (a, b): **a** is the row player's payoff, **b** the column player's.

Ayşe (card) \ Bora (guess)	Red	Black
Red	(−3 , 3)	(1 , −1)
Black	(1 , −1)	(−1 , 1)

Example: Guess the Card

Ayşe holds a red or a black card; Bora guesses the color. If Bora guesses red correctly he takes 3 chips from Ayşe, if black, 1 chip; a wrong guess costs him 1 chip. (Red, Red) = (−3, 3): Ayşe held red and Bora guessed it; Ayşe pays 3, Bora gets 3. In every cell a + b = 0: the game is **zero-sum**.

Common mistake: taking b for the row player's payoff, or thinking Ayşe chooses the outcome alone. Ayşe picks only the *row*; Bora's column decides the cell.

Practice: Reading the Matrix

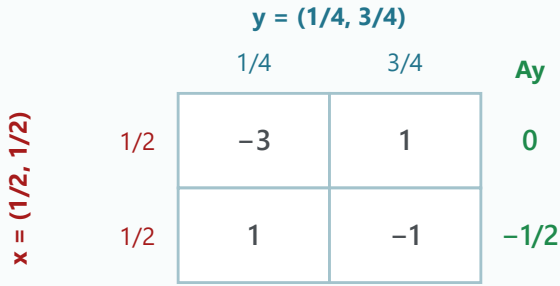
Ayşe (card) \ Bora (guess)	Red	Black
Red	(−3, 3)	(1, −1)
Black	(1, −1)	(−1, 1)

1. Ayşe holds Black and Bora says "Red". Who gets how many chips?
2. When Ayşe holds Black, which payoffs can Bora get?
3. If Bora says "Red" every round, which card should Ayşe hold?
4. What are Ayşe's largest and smallest possible payoffs?

▶ **Show answers**

Expected Payoff Is a Matrix–Vector Product: $x^T Ay$

A mixed strategy is a **probability vector**: $x = (x_1, \ldots, x_m)$, each $x_i \geq 0$, summing to 1. Let A be the row player's payoff matrix. **Ay**: multiply each row of A by y and add; each resulting number is that row's expected payoff against y. **$x^T Ay$**: weight these row values by the probabilities in x and add. (^{T} means x is written as a row.)



$x^T Ay = ½ \cdot 0 + ½ \cdot (−½) = −1/4$

Example: Guess the Card (Ayşe's payoffs)

A = [[−3, 1], [1, −1]], Ayşe x = (1/2, 1/2), Bora y = (1/4, 3/4).

Ay = (−3/4 + 3/4, 1/4 − 3/4) = (0, −1/2); $x^T Ay$ = ½ · 0 + ½ · (−½) = **−1/4**.

Common mistake: stopping at Ay, or swapping x and y. y weights the columns (Ay: each row's expectation), and x weights those row expectations. The position (row) of the largest entry of Ay is Ayşe's best pure response to y; its value is the payoff of that response.

Best Response

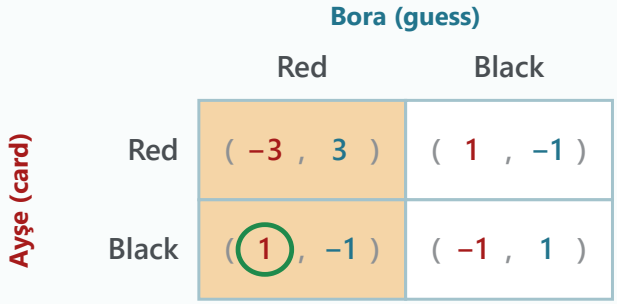
The answer to "what would I do if I knew what my opponent will do?" is a **best response** (BR). Fix the opponent's choice, look at *your own* payoffs, take the argmax.

Example: Guess the Card

Bora always says "Red": Ayşe's first entries in that column are Red −3, Black 1 ⇒ best response **Black**.

Bora always says "Black": Red 1, Black −1 ⇒ best response **Red**.

If Bora mixes: compute each row's expected payoff and pick the larger.



Common mistake: looking at the opponent's payoff (the second entry) when choosing a best response. Ayşe compares only her own numbers. Also, a best response is not fixed: it can change when the opponent changes.

Trace: Best Responses in a 2×2 Matrix

Ayşe \ Bora	Left	Right
Up	(1 , 2)	(3 , 1)
Down	(2 , 3)	(0 , 0)

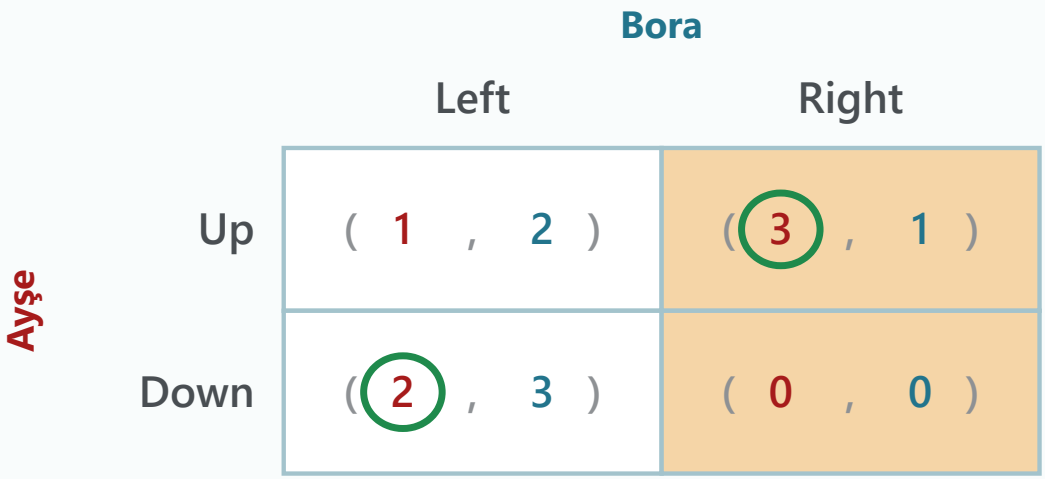
Ayşe picks a row (Up/Down), Bora a column (Left/Right). Nothing marked yet.

Trace: Best Responses in a 2×2 Matrix

Ayşe \ Bora	Left	Right
Up	(1, 2)	(3, 1)
Down	(2, 3)	(0, 0)

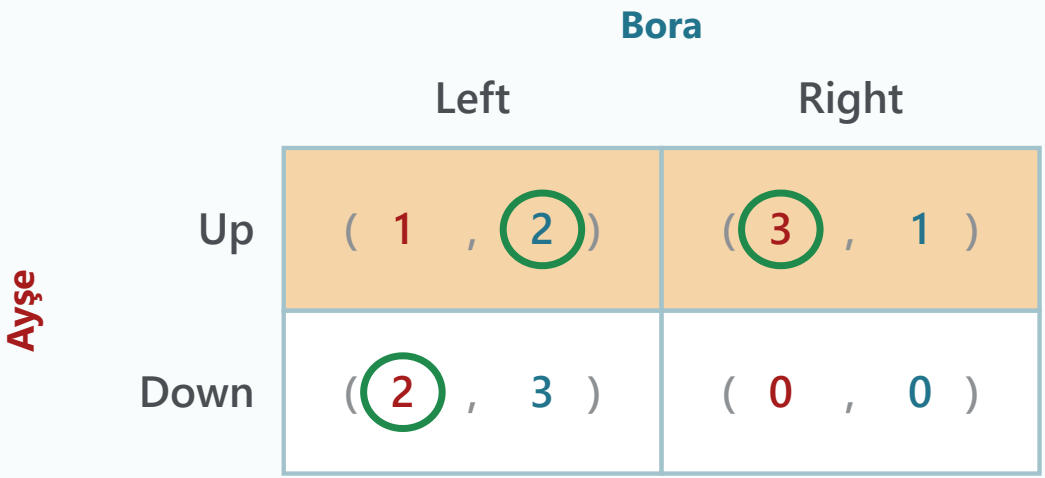
If Bora plays **Left**, Ayşe compares the first entries: Up 1, Down 2 ⇒ **Down**; mark 2.

Trace: Best Responses in a 2×2 Matrix



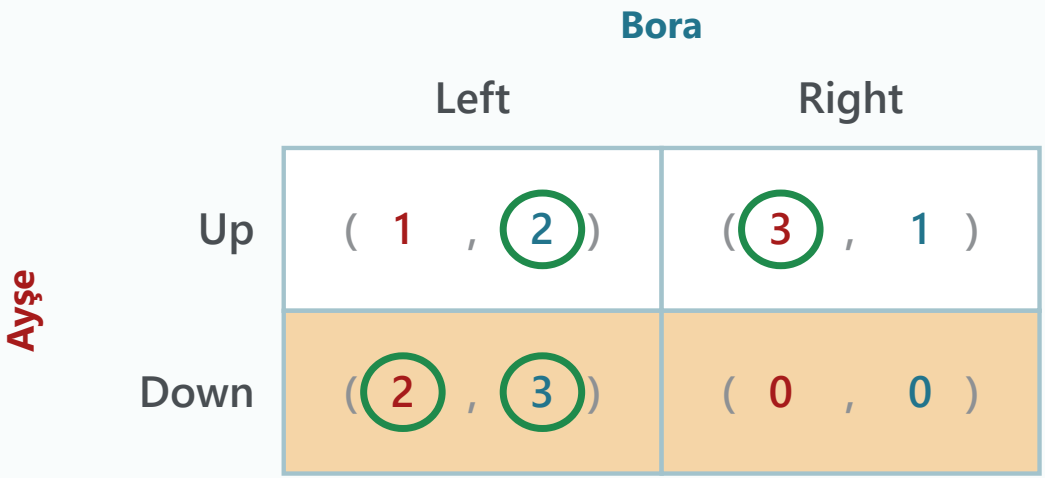
If Bora plays **Right**: Up 3, Down 0 ⇒ **Up**; mark 3.

Trace: Best Responses in a 2×2 Matrix



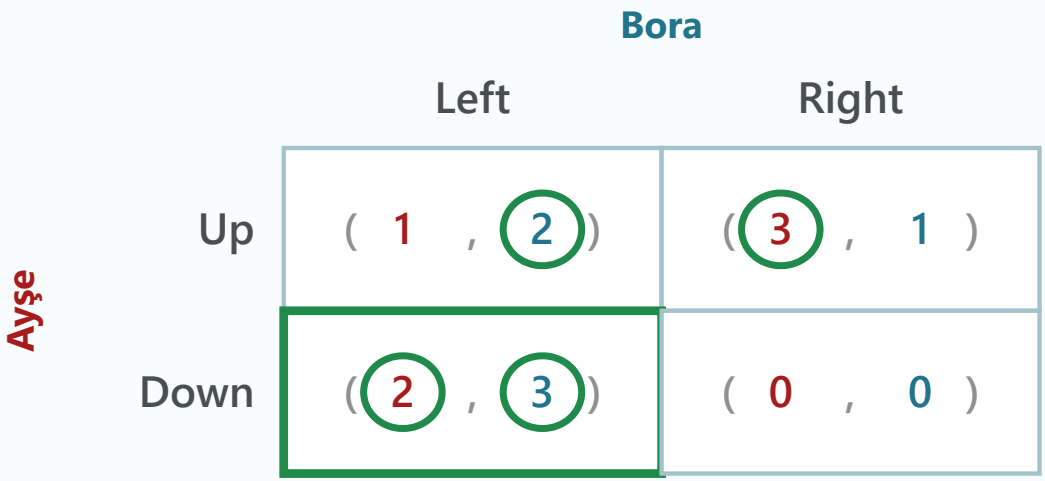
If Ayşe plays **Up**, Bora compares the second entries: Left 2, Right 1 ⇒ **Left**.

Trace: Best Responses in a 2×2 Matrix



If Ayşe plays **Down**: Left 3, Right 0 ⇒ **Left**. Left is better for Bora either way.

Trace: Best Responses in a 2×2 Matrix

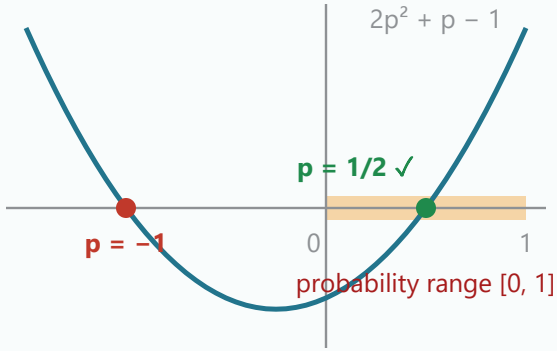


The only cell with both entries marked is **(Down, Left) = (2, 3)**: both play best responses, so neither wants to deviate alone. This is a **Nash equilibrium**.

Quadratic Equations and the Discriminant

$ax^2 + bx + c = 0$ ($a \neq 0$). The **discriminant** $\Delta = b^2 - 4ac$ gives the number of roots: two if $\Delta > 0$, one if $\Delta = 0$, no real root if $\Delta < 0$. The roots are:

$$x = (-b \pm \sqrt{\Delta}) / (2a)$$



Example

For a probability p we got $2p^2 + p - 1 = 0$. $\Delta = 1^2 - 4 \cdot 2 \cdot (-1) = 9$, $\sqrt{\Delta} = 3$.

$p = (-1 \pm 3)/4 \Rightarrow p = 1/2$ or $p = -1$. A probability lies between 0 and 1: **$p = 1/2$**. Check: $2 \cdot 1/4 + 1/2 - 1 = 0$ ✓

Common mistake: writing b instead of −b, or dividing only $\sqrt{\Delta}$ by 2a: the fraction bar runs under all of $(-b \pm \sqrt{\Delta})$. Missing the sign of −4ac when c is negative is also common: here $-4 \cdot 2 \cdot (-1) = +8$.

Two Linear Equations in Two Unknowns

Two unknowns need two equations: eliminate one, solve for the other, substitute back.

Example: a game shop

2 dice and 1 deck cost 50 TL; 1 die and 1 deck cost 35 TL. Prices?

(1) $2z + d = 50$ (2) $z + d = 35$ z: die, d: deck

Step 1 / 4

Typical form in game theory: probabilities x and y with $x + y = 1$ and $x = 2y \Rightarrow 2y + y = 1 \Rightarrow y = 1/3$, $x = 2/3$.

Common mistake: losing a sign when subtracting: $(2z + d) - (z + d)$ leaves z, and the right side is $50 - 35$. Check both equations.

Two Linear Equations in Two Unknowns

Two unknowns need two equations: eliminate one, solve for the other, substitute back.

Example: a game shop

2 dice and 1 deck cost 50 TL; 1 die and 1 deck cost 35 TL. Prices?

(1) $2z + d = 50$ (2) $z + d = 35$ z: die, d: deck

(1) − (2): $(2z + d) - (z + d) = 50 - 35 \Rightarrow \mathbf{z = 15}$

Step 2 / 4

Typical form in game theory: probabilities x and y with $x + y = 1$ and $x = 2y \Rightarrow 2y + y = 1 \Rightarrow y = 1/3$, $x = 2/3$.

Common mistake: losing a sign when subtracting: $(2z + d) - (z + d)$ leaves z, and the right side is $50 - 35$. Check both equations.

Two Linear Equations in Two Unknowns

Two unknowns need two equations: eliminate one, solve for the other, substitute back.

Example: a game shop

2 dice and 1 deck cost 50 TL; 1 die and 1 deck cost 35 TL. Prices?

(1) $2z + d = 50$ (2) $z + d = 35$ z: die, d: deck

(1) − (2): $(2z + d) - (z + d) = 50 - 35 \Rightarrow \mathbf{z = 15}$

Substitute into (2): $15 + d = 35 \Rightarrow \mathbf{d = 20}$

Step 3 / 4

Typical form in game theory: probabilities x and y with $x + y = 1$ and $x = 2y \Rightarrow 2y + y = 1 \Rightarrow y = 1/3$, $x = 2/3$.

Common mistake: losing a sign when subtracting: $(2z + d) - (z + d)$ leaves z, and the right side is $50 - 35$. Check both equations.

Two Linear Equations in Two Unknowns

Two unknowns need two equations: eliminate one, solve for the other, substitute back.

Example: a game shop

2 dice and 1 deck cost 50 TL; 1 die and 1 deck cost 35 TL. Prices?

(1) $2z + d = 50$ (2) $z + d = 35$ z: die, d: deck

(1) − (2): $(2z + d) - (z + d) = 50 - 35 \Rightarrow \mathbf{z = 15}$

Substitute into (2): $15 + d = 35 \Rightarrow \mathbf{d = 20}$

Check (1): $2 \cdot 15 + 20 = 50$ ✓

Step 4 / 4

Typical form in game theory: probabilities x and y with $x + y = 1$ and $x = 2y \Rightarrow 2y + y = 1 \Rightarrow y = 1/3$, $x = 2/3$.

Common mistake: losing a sign when subtracting: $(2z + d) - (z + d)$ leaves z, and the right side is $50 - 35$. Check both equations.

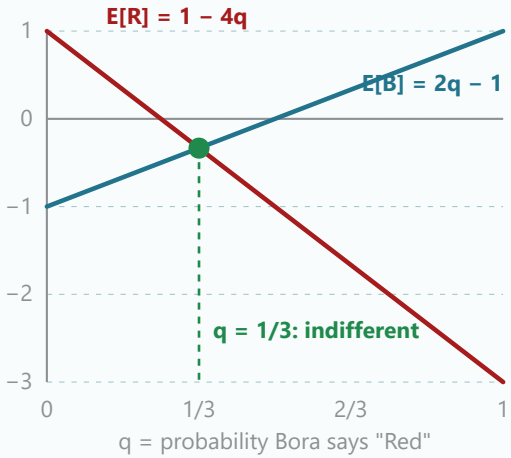
The Idea of Indifference

If a player genuinely mixes between two actions, both must give her the **same** expected payoff; if one were better, she would always play it. This is **indifference**.

In Guess the Card

If Bora says "Red" with probability q, Ayşe gets $E[R] = -3q + 1 \cdot (1 - q) = 1 - 4q$ and $E[B] = 1 \cdot q - 1 \cdot (1 - q) = 2q - 1$.

$q < 1/3$: Red is better. $q > 1/3$: Black is better. Only at $q = 1/3$ is Ayşe indifferent.



Common mistake: trying to get Ayşe's own probability from her own indifference condition. Ayşe's indifference pins down *Bora's* q. Ayşe's probability p of holding Red comes from the condition that makes Bora indifferent.

Trace: The Indifference Condition

Ayşe \ Bora	"R"	"B"
R	(−3 , 3)	(1 , −1)
B	(1 , −1)	(−1 , 1)

p: probability Ayşe holds Red. q: probability Bora says "Red".

Let Bora say "Red" with probability **q** and "Black" with **1 − q** (total 1).

Step 1 / 5

Trace: The Indifference Condition

Ayşe \ Bora	"R"	"B"
R	(−3 , 3)	(1 , −1)
B	(1 , −1)	(−1 , 1)

p: probability Ayşe holds Red. q: probability Bora says "Red".

Let Bora say "Red" with probability **q** and "Black" with **1 − q** (total 1).

Ayşe holds Red: E[R] = −3 · q + 1 · (1 − q) = **1 − 4q**

Step 2 / 5

Trace: The Indifference Condition

Ayşe \ Bora	"R"	"B"
R	(−3 , 3)	(1 , −1)
B	(1 , −1)	(−1 , 1)

p: probability Ayşe holds Red. q: probability Bora says "Red".

Let Bora say "Red" with probability **q** and "Black" with **1 − q** (total 1).

Ayşe holds Red: E[R] = −3 · q + 1 · (1 − q) = **1 − 4q**

Ayşe holds Black: E[B] = 1 · q + (−1) · (1 − q) = **2q − 1**

Step 3 / 5

Trace: The Indifference Condition

Ayşe \ Bora	"R"	"B"
R	(−3 , 3)	(1 , −1)
B	(1 , −1)	(−1 , 1)

p: probability Ayşe holds Red. q: probability Bora says "Red".

Let Bora say "Red" with probability **q** and "Black" with **1 − q** (total 1).

Ayşe holds Red: E[R] = −3 · q + 1 · (1 − q) = **1 − 4q**

Ayşe holds Black: E[B] = 1 · q + (−1) · (1 − q) = **2q − 1**

Indifference: 1 − 4q = 2q − 1 ⇒ 2 = 6q ⇒ **q = 1/3**

Trace: The Indifference Condition

Ayşe \ Bora	"R"	"B"
R	(−3 , 3)	(1 , −1)
B	(1 , −1)	(−1 , 1)

p: probability Ayşe holds Red. q: probability Bora says "Red".

Let Bora say "Red" with probability **q** and "Black" with **1 − q** (total 1).

Ayşe holds Red: E[R] = −3 · q + 1 · (1 − q) = **1 − 4q**

Ayşe holds Black: E[B] = 1 · q + (−1) · (1 − q) = **2q − 1**

Indifference: 1 − 4q = 2q − 1 ⇒ 2 = 6q ⇒ **q = 1/3**

Check: E[R] = 1 − 4/3 = −1/3 and E[B] = 2/3 − 1 = −1/3 ✓ With q = 1/3, Ayşe gets the same expected payoff whichever card she holds.

Back to the Game: Rock-Paper-Scissors

Is there an unbeatable way to play?

Yes: (1/3, 1/3, 1/3). Whatever the opponent plays, the expected payoff is 0; e.g. against Paper: $1/3 \cdot (-1) + 1/3 \cdot 0 + 1/3 \cdot (+1) = 0$. Every reply earns the same, so there is nothing to exploit.

What does always-Rock lose?

The opponent's best response is always Paper: **−1** per round. Even a milder bias loses: playing (1/2, 1/4, 1/4) earns $1/2 \cdot (-1) + 1/4 \cdot 0 + 1/4 \cdot (+1) = -1/4$ against Paper.

Your mix (R, P, S)	Opponent's best response	Your expected payoff
(1, 0, 0): always Rock	Paper	−1
(1/2, 1/4, 1/4)	Paper	−1/4
(1/3, 1/3, 1/3)	all equal	0

How do people play? In lab experiments each move's frequency is close to 1/3, yet consecutive moves are not independent: winners tend to repeat their move and losers to switch (Wang, Xu and Zhou, 2014). A pattern-learning opponent exploits exactly this.

Reading a Game Tree

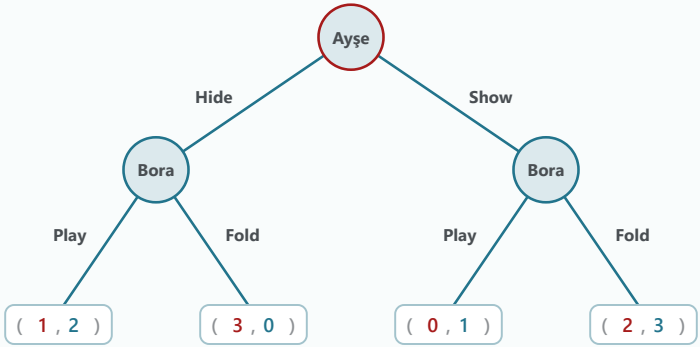
Sequential games use a **game tree**: a circled node shows whose turn it is, branches are actions, and the **leaves** at the bottom give the (Ayşe, Bora) payoffs when the game ends.

Example

Ayşe hides or shows her card. Bora, *seeing* what Ayşe did, plays on or folds.

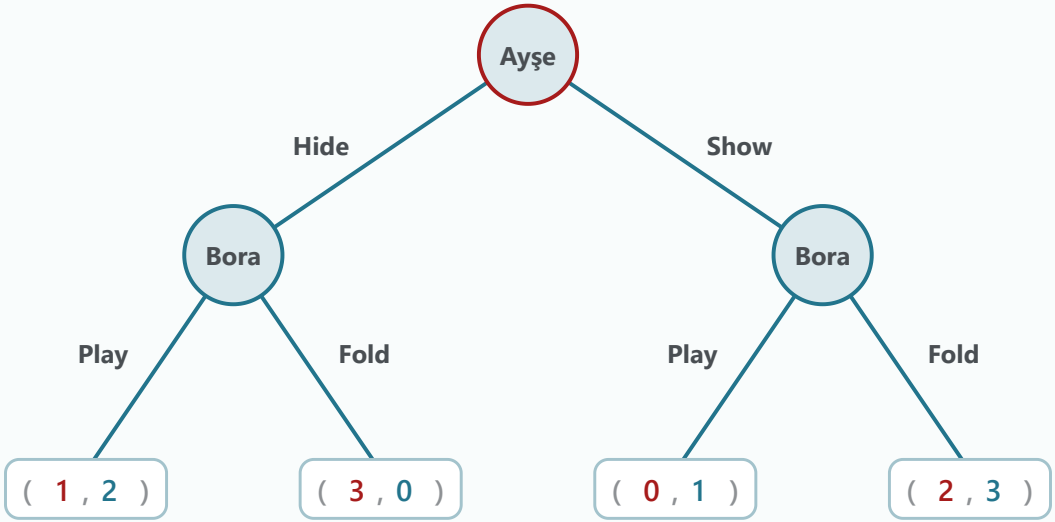
Path Hide → Fold: leaf (3, 0); Ayşe gets 3, Bora 0.

Bora has two nodes and decides separately at each.



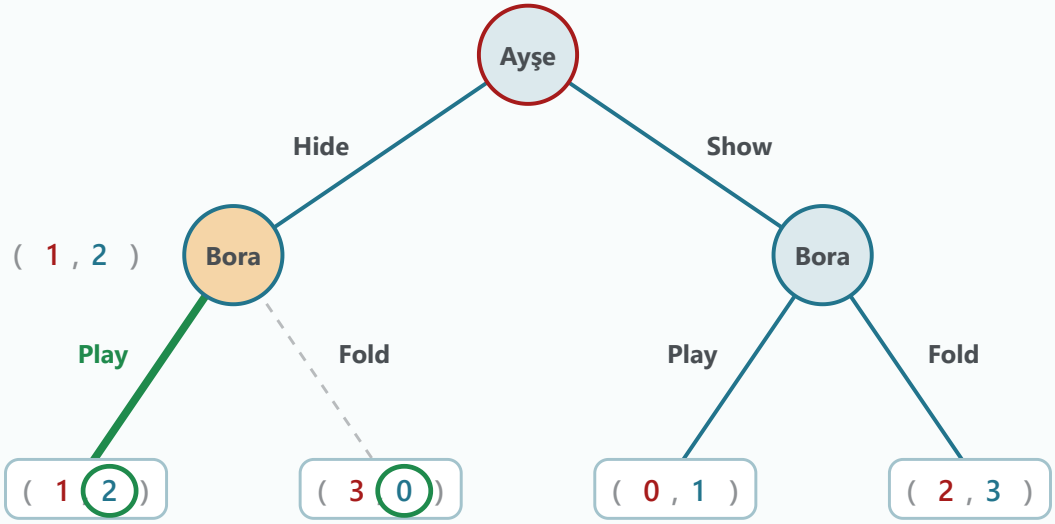
Common mistake: mixing up the order at a leaf: the first number is the payoff of the player who moves first (Ayşe), the second is Bora's. Also forgetting the difference from a matrix: here Bora *sees* Ayşe's move.

Trace: Backward Induction



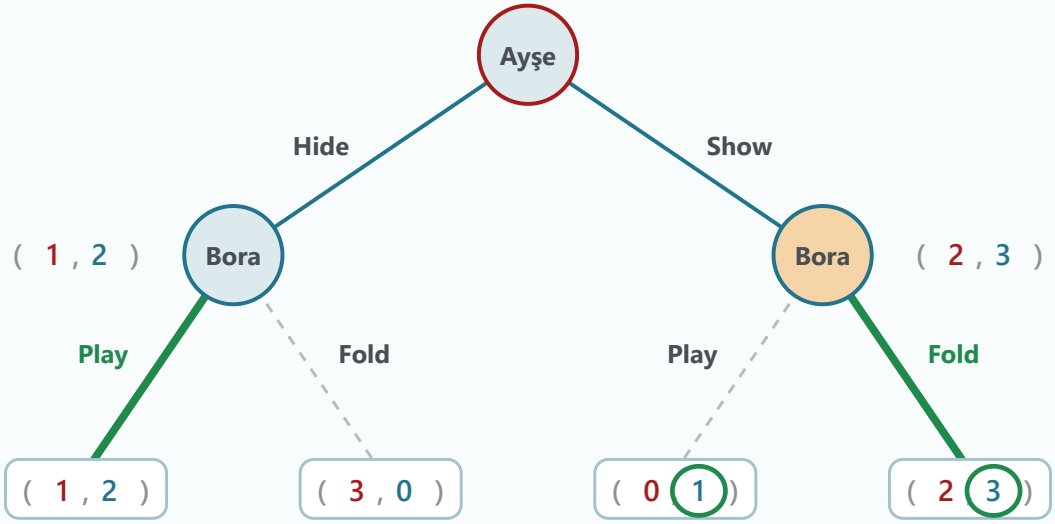
Idea: start with the player who moves **last**. Once Bora's choice at each node is known, Ayşe's decision is easy.

Trace: Backward Induction



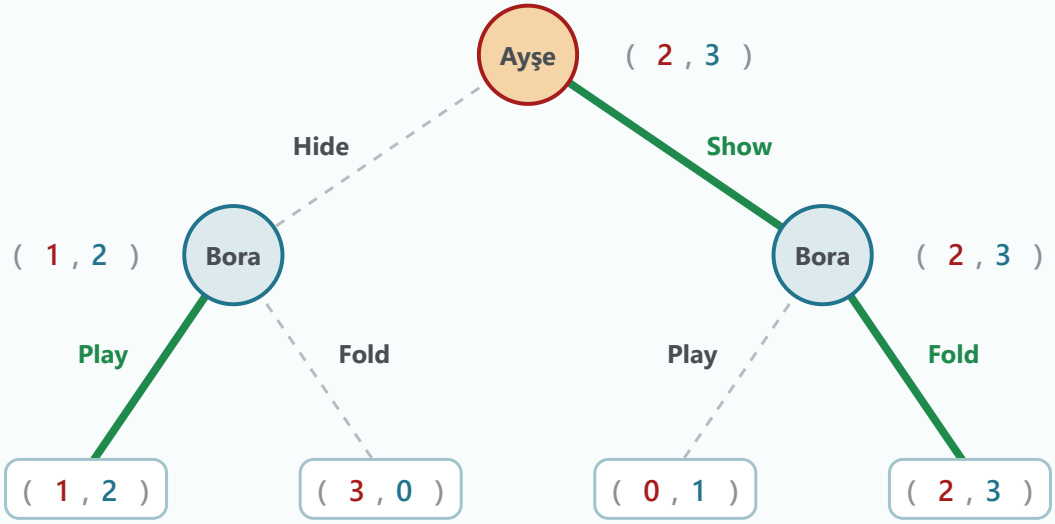
After Hide, Bora compares the second entries: Play 2, Fold 0 ⇒ **Play**. This node is worth (1, 2).

Trace: Backward Induction



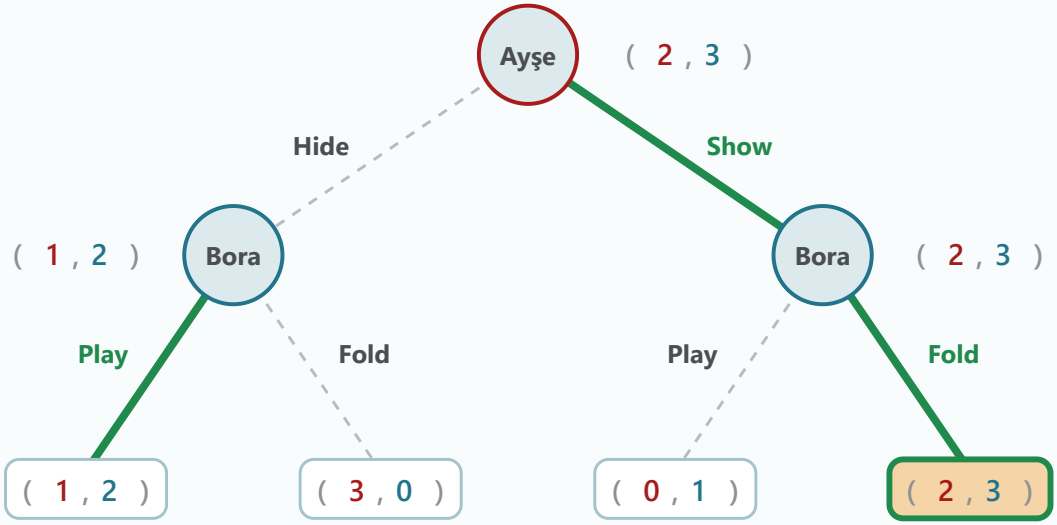
After Show, Bora: Play 1, Fold 3 ⇒ **Fold**. This node is worth (2, 3).

Trace: Backward Induction



At the root, Ayşe anticipates Bora's replies: Hide → 1, Show → 2 ⇒ **Show**.

Trace: Backward Induction



The resulting path is Show → Fold with payoff **(2, 3)**. Leaf (3, 0) is Ayşe's largest number, but Bora never goes there.

Growth Rates: O and Θ Notation

How does an algorithm's step count grow with the input size n? **f(n) = O(g(n))**: for large n, f stays below a constant multiple of g (an upper bound). **f(n) = Θ(g(n))**: f grows at the same rate as g (squeezed between constant multiples of g). Constant factors and lower-order terms are dropped: $3n^2 + 5n = \Theta(n^2)$.

Example

Order: $1 < \log\log n < \log n < n < n \log n < n^2 < 2^n$.

If every node of a game tree has 2 branches and the depth is d, there are 2^d leaves: backward induction takes $\Theta(2^d)$ steps, about 10^9 for d = 30.

n	$\log_2 \log_2 n$	$\log_2 n$	n^2	2^n
16	2	4	256	65,536
256	3	8	65,536	$\approx 10^{77}$
65,536	4	16	$\approx 4.3 \cdot 10^9$	huge

Common mistake: reading O as "exactly": $n = O(n^2)$ is true (just an upper bound), but $n = \Theta(n^2)$ is false. O and Θ describe the trend for large n; for small n a $\Theta(n^2)$ method can beat a $\Theta(n)$ one.

SUMMARY

Common Mistakes

Topic	Mistake	Correct
Profile	(Rock, Paper) = (Paper, Rock)	Order matters: Ayşe first, then Bora
Σ	$(\Sigma x_i)^2 = \Sigma x_i^2$	Evaluate the terms first, then add
argmax	"argmax = 7"	argmax is an option (B); max is a value (7)
Probability	At least one 6: 1/6 + 1/6	$1 - (5/6)^2 = 11/36$
Expected value	"E = 3.5 will come up"	Long-run average; a single roll may never show it
Mixed strategy	Cycling through moves	One random pure move each round
Matrix	Giving b to the row player	(a, b): a is the row player's, b the column player's
Best response	Looking at the opponent's numbers	Fix the opponent's choice, compare your own numbers
Indifference	Getting p from Ayşe's condition	p comes from the condition that makes Bora indifferent
Game tree	Starting at the root	From leaves to root: last move first

Summary

Notation

$S_1 \times S_2$: all profiles; u_i: i's payoff.

Σ, max, argmax

Σ: evaluate, add. max: a value; argmax: the option giving it.

Exponents, e and ln

$x^a \cdot x^b = x^{a+b}$, $x^0 = 1$; ln inverts e^x; $(\ln x)' = 1/x$.

Derivative rules

$(fg)' = f'g + fg'$; chain: outer × inner; $\partial u/\partial q$: others fixed.

Probability

Count and divide; multiply if independent; $P(A \mid B) = P(A \cap B)/P(B)$.

Expected value

$E[X] = \Sigma\, x \cdot P(x)$; if $X \geq 0$, $E[X] = \int P(X > t)\, dt$.

Mixed strategy

Distribution over actions; softmax: $P(i) \propto e^{\lambda v_i}$.

Payoff matrix

(a, b): row payoff a, column payoff b.

Best response

Fix the opponent; argmax of your payoff.

Equations

$x = (-b \pm \sqrt{\Delta})/(2a)$; indifference: actions played in equilibrium pay the same.

Backward induction

Leaves to root; the mover picks the best.

O and Θ

$\log \log n < \log n < n < n^2 < 2^n$; drop constants.

SUMMARY

Test Yourself

1. Two coins are tossed. What is the probability of at least one head?
2. A lottery pays 50 TL with probability 0.2 and 0 TL with probability 0.8; a ticket costs 8 TL. What is the net expected payoff?
3. Payoffs: Rock 2, Paper −1, Scissors 2. What are max and argmax?
4. Bora plays (1/4 Rock, 1/2 Paper, 1/4 Scissors). What is Ayşe's best response?
5. If $x + y = 1$ and $3x = y$, what are x and y?
6. A die is known to show an even number. What is the probability it is a 6?
7. At which q does $f(q) = q(6 - q)$ reach its maximum?

▶ **Show answers**

The End

Chapter 0: Preliminaries — Mathematics for Game Theory

Covered: sets and counting, Σ and series, exponents, e and ln, derivative rules, probability and expected value, softmax, matrices, equations, O and Θ

Next: Chapter 1: Decision Theory