FACULTY OF
COMPUTER SCIENCE

IA@FII

Artificial Intelligence

3rd year, 1st semester

Week 5: Game theory

"The true source of uncertainty lies in the intentions of others." Peter L. Bernstein

What is a Game?

An adversarial search problem

A interactive decision problem where

- The solution is not built exclusively by a single strategy
- Not all final states are desirable

A Game always has:

- A starting state (the game instance)
- At least one state which determines the end of the game if reached
- At least one goal for each player (not necessarily an end state)
- Rules for each player (not always the same)
- At least two sources determining changes to the current state

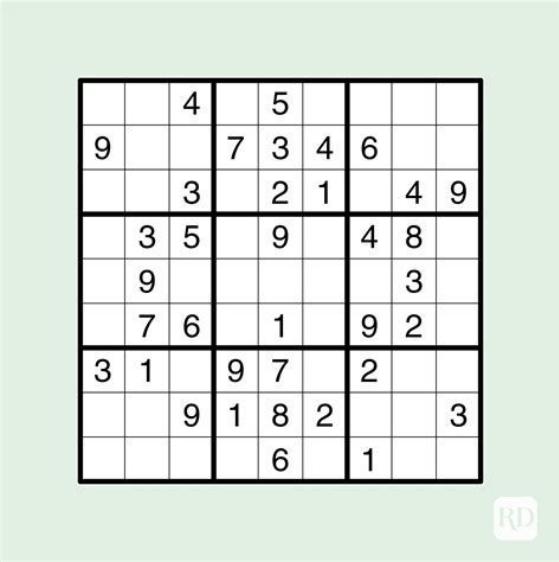
Additional references

Stanford Game Theory Online

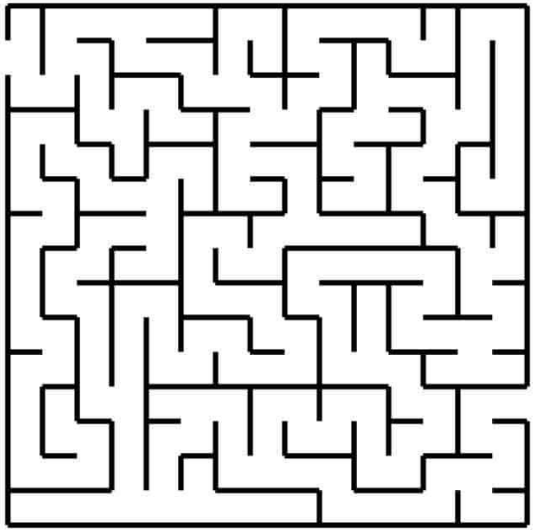
Yale Game Theory

Giacomo Bonanno Game Theory

These are puzzles (decision problems) and not games (interactive decision problems)!



		4		5				
9			7	3	4	6		
		3		2	1		4	9
	3	5		9		4	8	
	9						3	
	7	6		1		9	2	
3	1		9	7		2		
		9	1	8	2			3
				6		1		



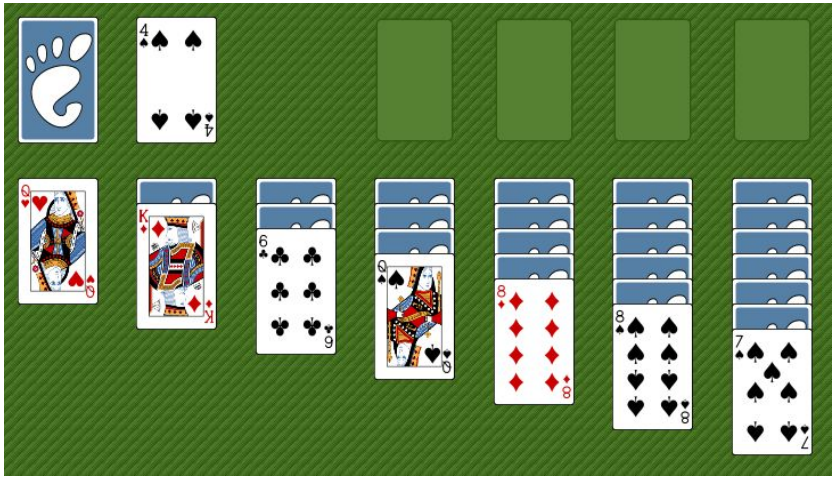
Zero sum games

- Something won by a player is lost by the other(s).
- Also called strictly competitive games.
- Many games are not zero-sum!

Non zero-sum game:
I have *n* money and I want to give it to a group of *m* players. Each of the *m* players has to write a sum on a piece of paper visible only to them. I collect all pieces of paper and total the sums. If the total is at most as much as *n*, each player gets the written amount, otherwise nobody gets anything.

Single player games

- One active player - one controlling strategy
- At least one additional source of information
- Cannot be zero-sum games



Multiple player games

- Can have different rules and goals for all players.
- Symmetrical games: all rules and goals are the same for all players.

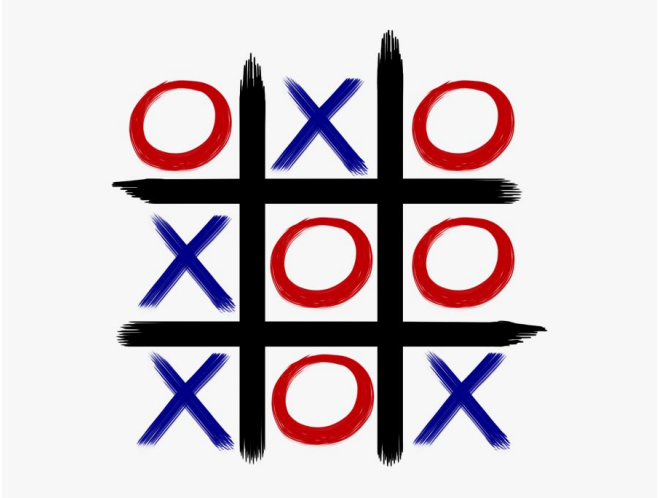
Other types of games

- If the state change is determined by multiple player decisions at the same time, the game is **simultaneous**.
- If the state change is determined in alternative turns by each player, the game has **alternate** moves.
- If each player knows everything about the current state of the game, all previous decisions made by all players and all rules and goals of all players, the game has **perfect information**.
- **Imperfect information** games: card games, kriegspiel, others.

OK, now let's implement a game as an interactive decision problem

The game: Tic-Tac-Toe

On a 3x3 board two players place alternatively pieces (first one X pieces, second one O pieces) until either one of them has three pieces in the same row, column or diagonal (in which case that player has won the game), or no more pieces can be placed (in which case the game ended in a draw).

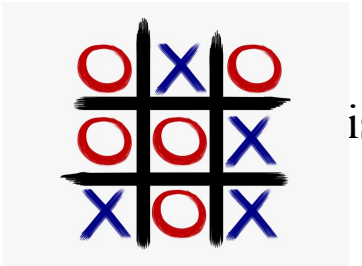


Two player game, symmetrical, zero-sum, alternate moves.

Describing a state

$(p, b_{11}, b_{12}, b_{13}, b_{21}, b_{22}, b_{23}, b_{31}, b_{32}, b_{33})$

- $p \in \{1,2\}$, 1 if we are next to move, 2 if the opponent;
- $b_{ij} \in \{0,1,2\}$, 0 if the b_{ij} position is empty, 1 if it's occupied by one of our pieces and 2 if it's one of the opponents.



is (1, 2, 1, 2, 2, 2, 1, 1, 2, 1)

Game initialisation

State initialisation (int first_player)
{
    return (first_player, 0, 0, 0, 0, 0, 0, 0, 0, 0) ;
}

Select which player is moving first and send selection as parameter for initialisation.

Game ending

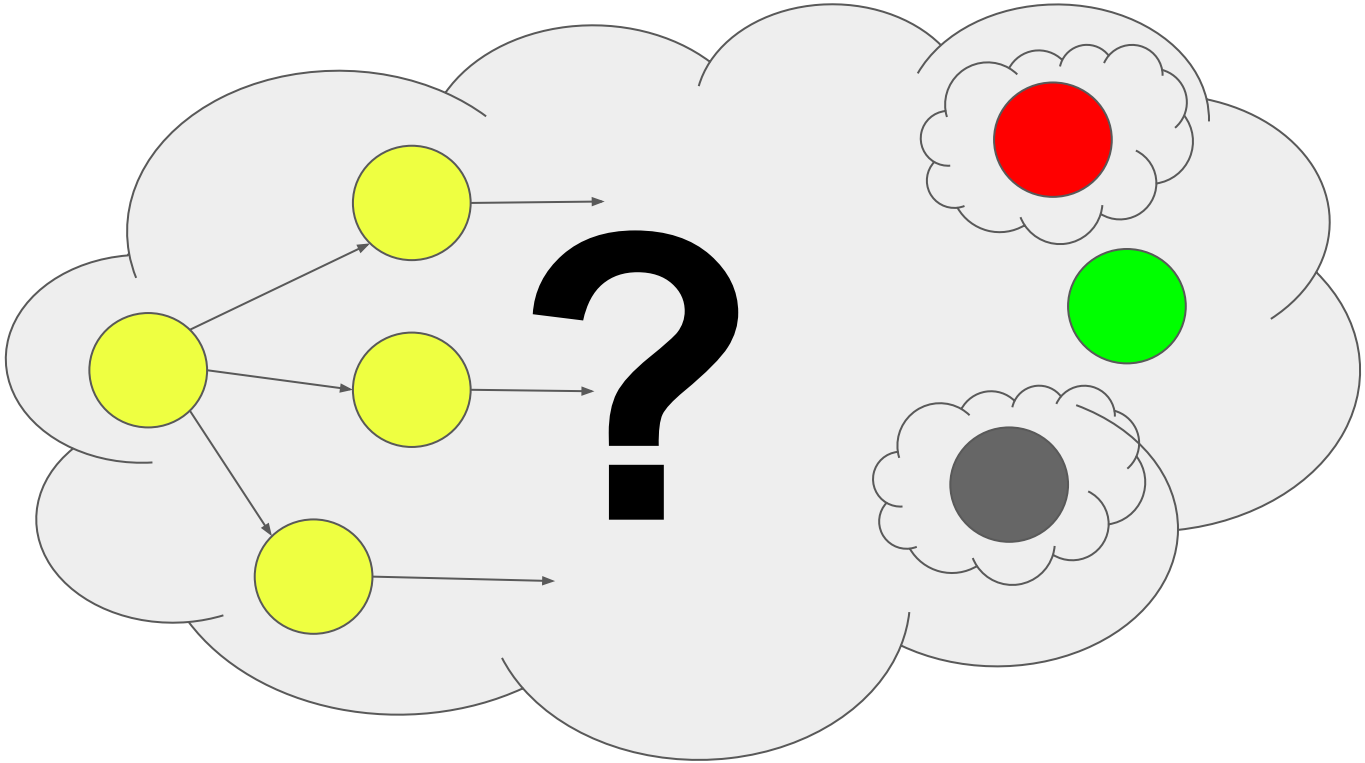
```
int IsFinal (State S)
{
    if((S[1]==S[2]==S[3])||(S[4]==S[5]==S[6])||(S[7]==S[8]==S[9])
    ||((S[1]==S[4]==S[7])||(S[2]==S[5]==S[8])||(S[3]==S[6]==S[9])
    ||(S[1]==S[5]==S[9])||(S[3]==S[5]==S[7]))
    return 3-S[0];
    else
    return -1;
}
```

Transition and validation

```
State move (State S, int position)
{
    S[position]=S[0]; S[0]=3-S[0]; return S;
}

Boolean Validate (State S, int position)
{
    return (S[position]==0);
}
```

How should a strategy look like?



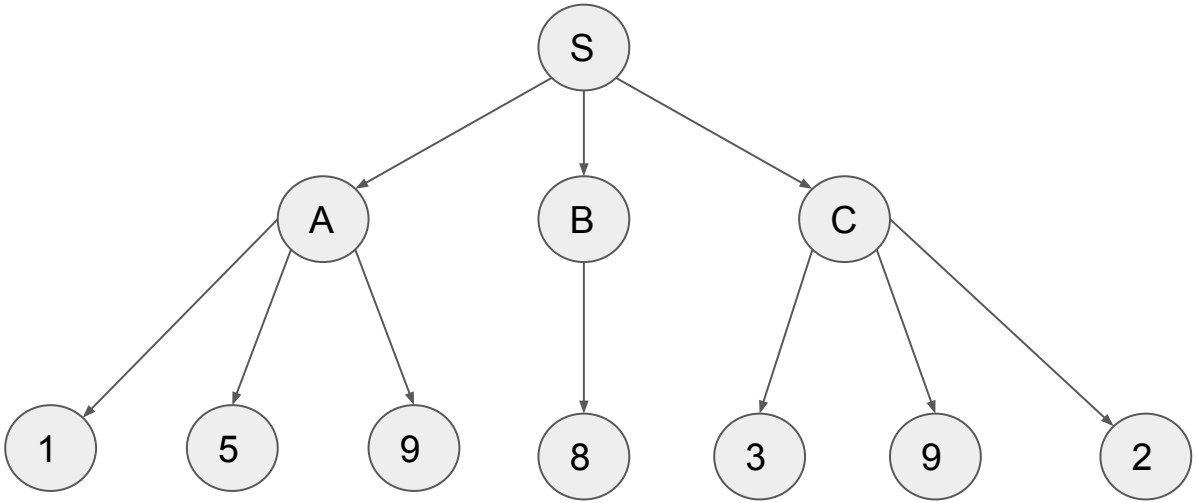
```
Void strategy(State s)
{
    While (!isFinal(s))
        {
            If it's my turn
                Choose position;
                If (Validate (s, position))
                    s = Transition(s, piece, tower);
            Else get other player(s) decisions and change s
        }
}
```

The MINIMAX strategy

- Select a **depth** for exploration, at the minimum it has to include a full step for all other players.
- Select a **score** function (heuristic) which evaluates a state as to its value for the other player(s). The better the score, the less desirable that state is from your perspective.

```
score minimax(depth, state, player)
{
     if(final(state)||(depth==0))
          return score(state);
     else
          if (player is the AI)
               return min(minimax(depth-1,each neighbour state,next_player);
          else
               return max(minimax(depth-1,each neighbour state,next_player);
}
```

The MINIMAX strategy example



minimax(2, S, AI) = min (score (minimax (1, A, OP)), score (minimax (1, B, OP)), score (minimax (1, C, OP))) = min (max (1, 5, 9), max (8), max (3, 9, 2)) = 8 (choice B)

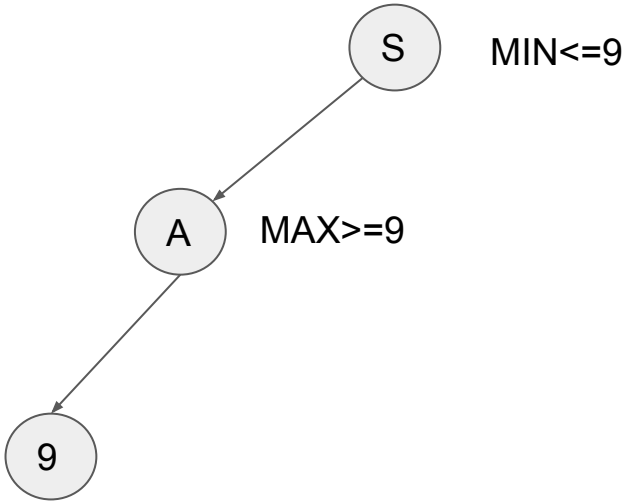
MINIMAX considerations

- The objective of MINIMAX is to allow the opponent to make the worst possible moves, assuming it will always choose the best move available.
- The depth of exploration is crucial to the MINIMAX accuracy.
- The depth of exploration is limited by the necessity to store every explored state. For chess (average branching factor of 35), a depth of 4 would require 1,500,625 states. An exhaustive exploration would require 10^{30} states.
- MINIMAX is predictable and assumes predictable opponents.

FACULTY OF COMPUTER SCIENCE

IA@FII

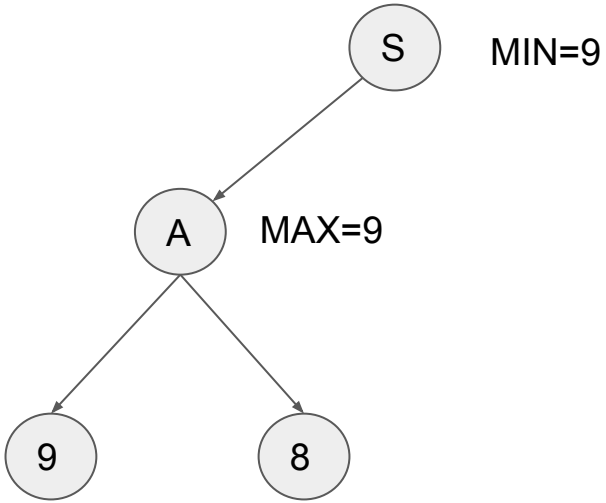
Optimising MINIMAX



FACULTY OF COMPUTER SCIENCE

IA@FII

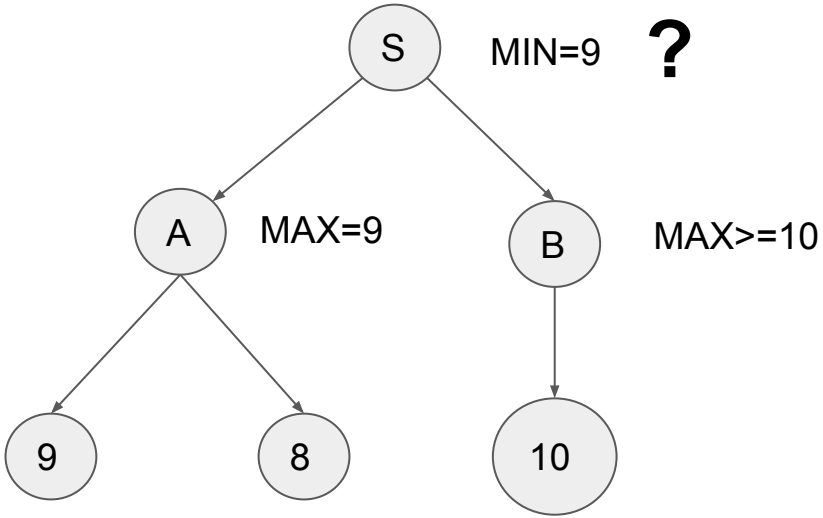
Optimising MINIMAX



FACULTY OF COMPUTER SCIENCE

IA@FII

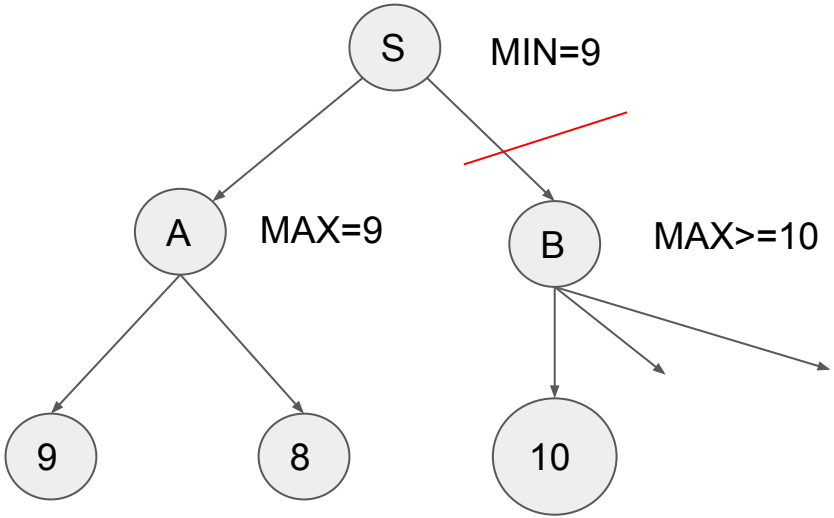
Optimising MINIMAX



FACULTY OF COMPUTER SCIENCE

IA@FII

Optimising MINIMAX

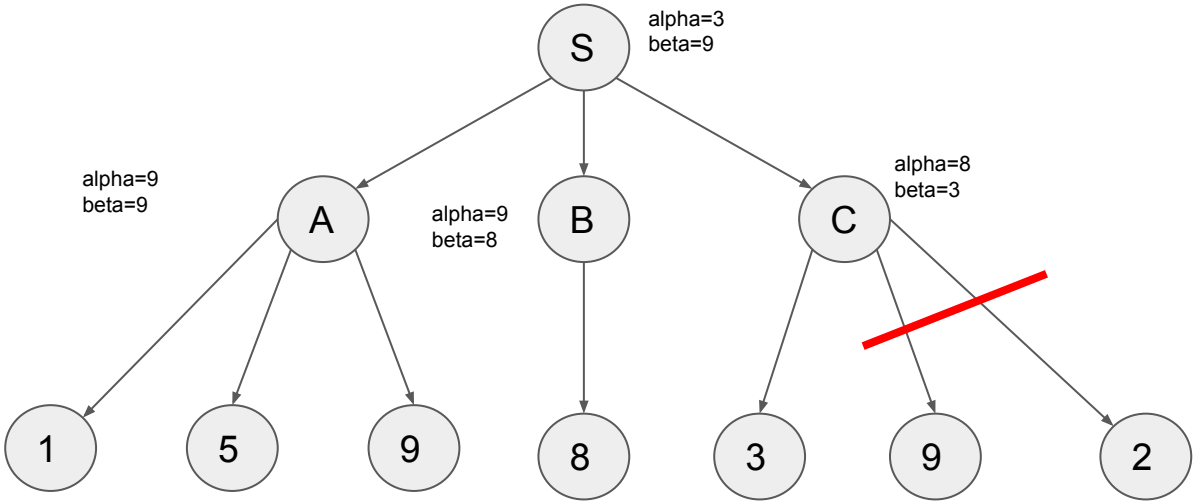


The ALPHA-BETA optimisation

- For each state up to (but not including) the selected depth, keep two values (*alpha* and *beta*), *alpha* being the minimum maximum up to that depth and *beta* being the maximum score up to that depth.
- If for a state *alpha* is greater or equal than *beta*, stop exploring further accessible states.
- Update *alpha* and *beta* scores each time a score is computed.

```
score alphabeta(depth, state, player, alpha, beta)
{
     if(final(state)||(depth==0))
          return score(state);
     else for all neighbour states
               score = alphabeta (depth-1, neighbour, next_player, alpha, beta);
               if (player is the AI)    ;minimizing step
                    if (score <= alpha) return alpha;
                    if (score<beta) beta=score;
               else                     ;maximizing step
                    if (score >= beta) return beta;
                    if (score>alpha) alpha=score;
}
```

ALPHA-BETA example

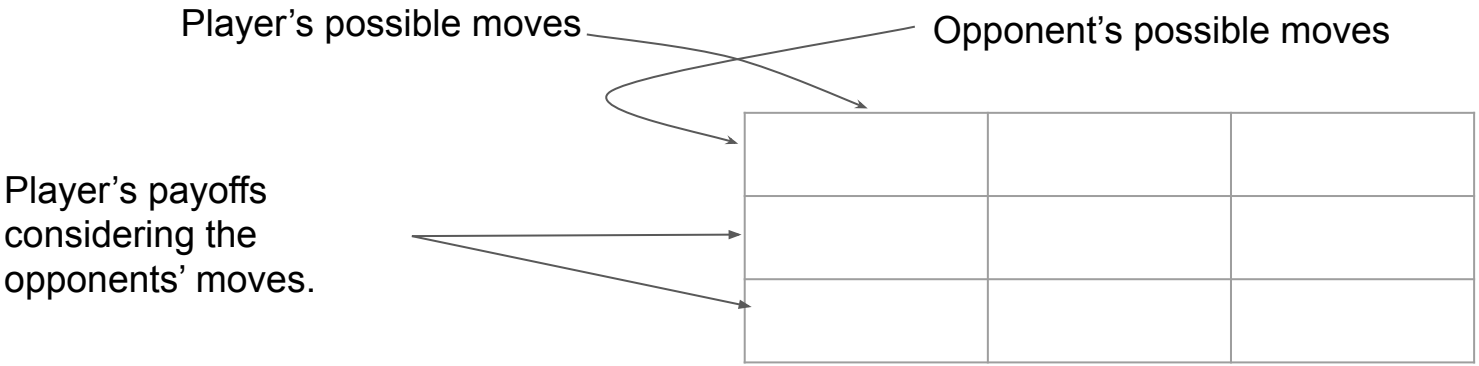


ALPHA-BETA is sensitive to the order of evaluation of neighbours and can be further optimised by an order heuristic.

Minimax game search algorithm with alpha-beta pruning

Normal (strategic) form representation

Uses a matrix of payoffs to represent all possible strategies for each player.
Works for complete games (all information is known to all players).
One matrix for each pair of players. One column for each combination of moves.



Prisoner's dilemma

A \ B	Deny	Confess
Deny	-1, -1	-3, 0
Confess	0, -3	-2, -2

What is the best strategy for A?

Prisoner's dilemma

A \ B	Deny	Confess
Deny	-1, -1	-3, 0
Confess	0, -3	-2, -2

Average payoff for A, with non-cooperation.

If A chooses Deny: (-1+-3)/2 = -2

If A chooses Confess: (0+-2)/2=-1

Prisoner's dilemma

A \ B	Deny	Confess
Deny	-1, -1	-3, 0
Confess	0, -3	-2, -2

Strategy: The decisions made by a player for all possible states in which he has to make a decision.
A strategy making the same choice everytime is called a **pure strategy**. For example, choosing "Deny" all the time is a pure strategy for player A.

Dominant strategy

A strategy is **dominant** if it always provides payoffs at least as good as any other strategy.

A strategy is strictly dominant is the payoffs are always better than any other strategy.

Is there a dominant strategy for Prisoner's Dilemma?

Dominant strategy

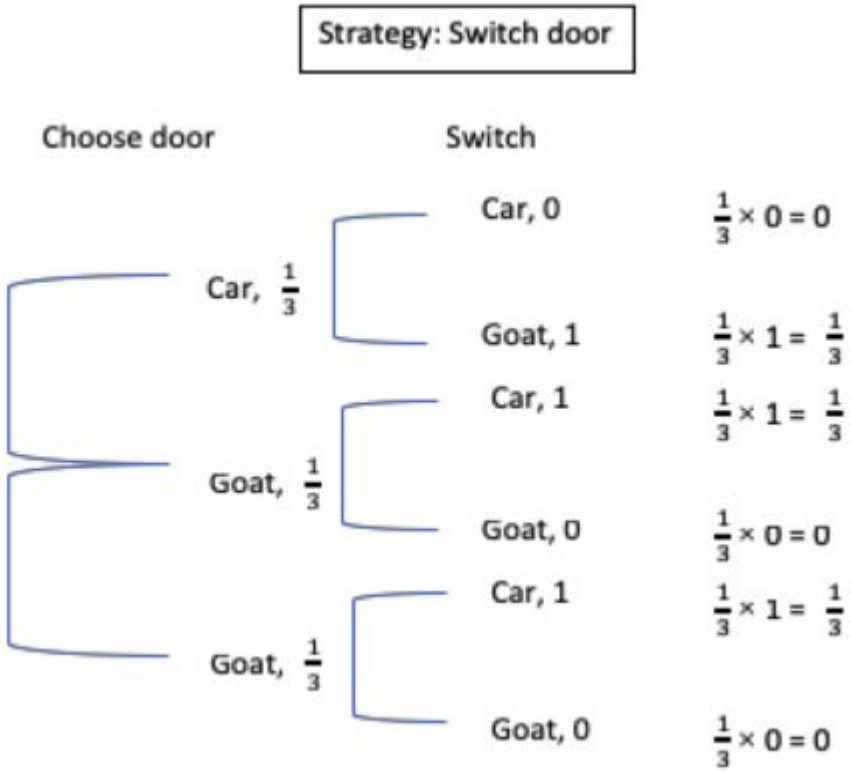
A strategy is **dominant** if it always provides payoffs at least as good as any other strategy.

A strategy is strictly dominant is the payoffs are always better than any other strategy.

Is there a dominant strategy for Prisoner's Dilemma?

Yes, "Confess" is a strictly dominant strategy.

The Monty Hall problem



Paretto optimality

A strategy is Paretto optimal if no change can be made to it in order to improve the players' payoff without diminishing the other players payoffs.

Both pure strategies (Deny/Deny) and (Confess/Confess) for Prisoner's dilemma are Paretto optimal.

What about the game below?

	X	Y
X	1 1	2 2
Y	2 2	2 3

Paretto optimality

A strategy is Paretto optimal if no change can be made to it in order to improve the players' payoff without diminishing the other players payoffs.

Both pure strategies (Deny/Deny) and (Confess/Confess) for Prisoner's dilemma are Paretto optimal.

What about the game below?
(X/X) or (X/Y) or (Y/X) are not Paretto optimal, (Y/Y) is.

	X	Y
X	1 / 1	2 / 2
Y	2 / 2	2 / 3

Nash Equilibrium

If in a game all players obey the game rules and aim to maximize their payoffs, there is at least one strategy for each player which guarantees maximum payoffs for each player.
That strategy is called Nash optimum or equilibrium.
No player has a reason to change an equilibrium as long as anybody follows one.

A dominant and Paretto optimal strategy is always an equilibrium.
However, for most games no dominant strategies exist.
More about equilibria.

Hunting game

	Stag	Rabbit
Stag	5 \ 5	0 \ 3
Rabbit	3 \ 0	4 \ 4

Are there any dominant strategies?
Are there any equilibria?

Hunting game

	Stag	Rabbit
Stag	5 / 5	0 / 3
Rabbit	3 / 0	4 / 4

Are there any dominant strategies? NO
Are there any equilibria? YES: (Stag, Stag) and (Rabbit, Rabbit)
Equilibria are dependent on the other players behaving as expected!

Chicken game

	Straight	Turn
Straight	-3 / -3	3 / -2
Turn	-2 / 3	-3 / -3

The two equilibria (Straight, Turn) and (Turn, Straight) are very dependent on the opponent's strategy, otherwise both players get worse results.

FACULTY OF
COMPUTER SCIENCE

IA@FII

The money game

I have *n* money and I want to give it to a group of *m* players. Each of the *m* players has to write a sum on a piece of paper visible only to them. I collect all pieces of paper and total the sums. If the total is at most as much as *n*, each player gets the written amount, otherwise nobody gets anything.

Is there any dominant strategy?
What is the equilibrium?

FACULTY OF
COMPUTER SCIENCE

IA@FII

The money game

I have *n* money and I want to give it to a group of *m* players. Each of the *m* players has to write a sum on a piece of paper visible only to them. I collect all pieces of paper and total the sums. If the total is at most as much as *n*, each player gets the written amount, otherwise nobody gets anything.

Is there any dominant strategy? NO
What is the equilibrium?

Each player writes n/m.

The Goalkeeper game

Shooter \ Keeper	Left	Right
Left	-1, 1	1, -1
Right	1, -1	-1, 1

There is no pure strategy equilibria.
What is the actual equilibria?

Mixed strategies

Strategies which are not pure are called mixed.

For most games, pure strategies are not equilibria.

Mixed strategies are usually non-deterministic, making some random decisions (stochastic or probabilistic).

How can we compute the equilibria for the goalkeeper game?

Computing equilibria as a mixed strategy

Shooter \ Keeper	Left	Right
Left	-1 / 1	1 / -1
Right	1 / -1	-1 / 1

The equilibria (EQS and EQK) must provide equal payoffs for all possible opponent decisions.
EQS(Left) payoff is -1*EQK(Left)+1*EQK(Right)
EQS(Right) payoff is 1*EQK(Left)-1*EQK(Right)

Equal payoff means that -1*EQK(Left)+1*EQK(Right)=1*EQK(Left)-1*EQK(Right)
So EQK(Left)=EQK(Right)=0.5 since EQK(Left)+EQK(Left)=1. Equilibria is reached when both choices are as probable.

Equilibria work only if all players obey the game rules and aim to maximize their payoffs.

Can you trust the other players? How much can you cooperate with them?

Minimax is an equilibria for all two player zero-sum game. A chess program implementing Minimax is easy to defeat. Why?

Trust in Game Theory

Parrondo's paradox

There is at least one winning strategy which can be built using two losing strategies alternatively.

Game: add or substract money. Game ends if sum at or below zero.

S1: If you have an even sum, add 3. Otherwise substract 5.
S2: If you have an odd sum, substract 1.
S1,S2,S1,S2,... is dominant for both S1 and S2.

More

FACULTY OF
COMPUTER SCIENCE

IA@FII

Overall considerations about strategies

- Strategies should have to have a way of estimating the missing information (eg. other player's decisions)
- Strategies have to be unpredictable: many use either random steps or select randomly between two or more strategies
- You can prove that some strategies are better than others
- If a there is a strategy better than all others then it's not a game, it's a puzzle
- An useful benchmark strategy is the equilibria which exists and can be determined for all games

Computer "outsmarts" Kasparov and Fisher outthinks computers

Monte-Carlo strategies

- Generate all neighbours of the current state and add them to the graph until a set depth has been reached OR until enough final states have been reached
- Run a simulation for all unexplored states in the graph
  - Until n final states are reached OR
  - Up to depth m
- Compute a score for each state after each simulation (for example (positive-negative)/n))
- Update the scores of all nodes above the current one using the newly computed score
- Return to the first step
- Select the neighbour of the starting state with the best score

More