

Introduction

- The normal form game representation does not incorporate any notion of sequence, or time, of the actions of the players
- The **extensive form** is an alternative representation that makes the temporal structure explicit.
- Two variants:
 - **perfect information** extensive-form games
 - **imperfect-information** extensive-form games

Definition

A (finite) **perfect-information game** (in extensive form) is defined by the tuple $(N, A, H, Z, \chi, \rho, \sigma, u)$, where:

- **Players:** N is a set of n players

Definition

A (finite) **perfect-information game** (in extensive form) is defined by the tuple $(N, A, H, Z, \chi, \rho, \sigma, u)$, where:

- **Players:** N
- **Actions:** A is a (single) set of actions

Definition

A (finite) **perfect-information game** (in extensive form) is defined by the tuple $(N, A, H, Z, \chi, \rho, \sigma, u)$, where:

- **Players:** N
- **Actions:** A
- Choice nodes and labels for these nodes:
 - **Choice nodes:** H is a set of non-terminal choice nodes

Definition

A (finite) **perfect-information game** (in extensive form) is defined by the tuple $(N, A, H, Z, \chi, \rho, \sigma, u)$, where:

- **Players:** N
- **Actions:** A
- Choice nodes and labels for these nodes:
 - **Choice nodes:** H
 - **Action function:** $\chi : H \rightarrow 2^A$ assigns to each choice node a set of possible actions

Definition

A (finite) **perfect-information game** (in extensive form) is defined by the tuple $(N, A, H, Z, \chi, \rho, \sigma, u)$, where:

- **Players:** N
- **Actions:** A
- Choice nodes and labels for these nodes:
 - **Choice nodes:** H
 - **Action function:** $\chi : H \rightarrow 2^A$
 - **Player function:** $\rho : H \rightarrow N$ assigns to each non-terminal node h a player $i \in N$ who chooses an action at h

Definition

A (finite) **perfect-information game** (in extensive form) is defined by the tuple $(N, A, H, Z, \chi, \rho, \sigma, u)$, where:

- **Players:** N
- **Actions:** A
- Choice nodes and labels for these nodes:
 - **Choice nodes:** H
 - **Action function:** $\chi : H \rightarrow 2^A$
 - **Player function:** $\rho : H \rightarrow N$
- **Terminal nodes:** Z is a set of terminal nodes, disjoint from H

Definition

A (finite) **perfect-information game** (in extensive form) is defined by the tuple $(N, A, H, Z, \chi, \rho, \sigma, u)$, where:

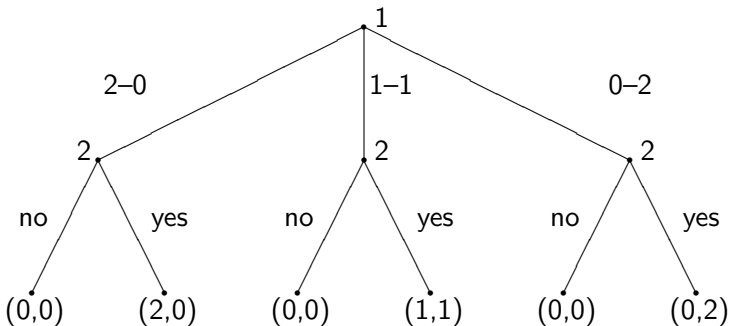
- **Players:** N
- **Actions:** A
- Choice nodes and labels for these nodes:
 - **Choice nodes:** H
 - **Action function:** $\chi : H \rightarrow 2^A$
 - **Player function:** $\rho : H \rightarrow N$
- **Terminal nodes:** Z
- **Successor function:** $\sigma : H \times A \rightarrow H \cup Z$ maps a choice node and an action to a new choice node or terminal node such that for all $h_1, h_2 \in H$ and $a_1, a_2 \in A$, if $\sigma(h_1, a_1) = \sigma(h_2, a_2)$ then $h_1 = h_2$ and $a_1 = a_2$
 - The choice nodes form a tree, so we can identify a node with its history.

Definition

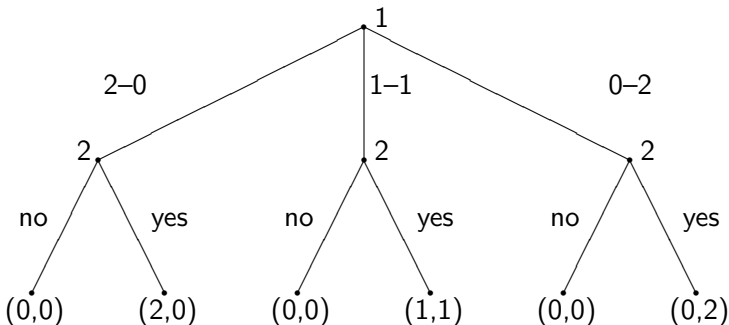
A (finite) **perfect-information game** (in extensive form) is defined by the tuple $(N, A, H, Z, \chi, \rho, \sigma, u)$, where:

- **Players:** N
- **Actions:** A
- Choice nodes and labels for these nodes:
 - **Choice nodes:** H
 - **Action function:** $\chi : H \rightarrow 2^A$
 - **Player function:** $\rho : H \rightarrow N$
- **Terminal nodes:** Z
- **Successor function:** $\sigma : H \times A \rightarrow H \cup Z$
- **Utility function:** $u = (u_1, \dots, u_n)$; $u_i : Z \rightarrow \mathbb{R}$ is a utility function for player i on the terminal nodes Z

Example: the sharing game



Example: the sharing game



Play as a fun game, dividing 100 dollar coins. (Play each partner only once.)

Pure Strategies

- In the sharing game (splitting 2 coins) how many pure strategies does each player have?

Pure Strategies

- In the sharing game (splitting 2 coins) how many pure strategies does each player have?
 - player 1: 3; player 2: 8

Pure Strategies

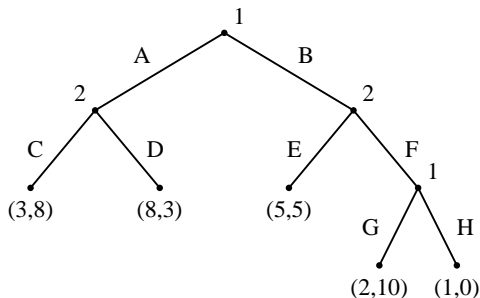
- In the sharing game (splitting 2 coins) how many pure strategies does each player have?
 - player 1: 3; player 2: 8
- Overall, a pure strategy for a player in a perfect-information game is a complete specification of which deterministic action to take at every node belonging to that player.

Definition (pure strategies)

Let $G = (N, A, H, Z, \chi, \rho, \sigma, u)$ be a perfect-information extensive-form game. Then the pure strategies of player i consist of the cross product

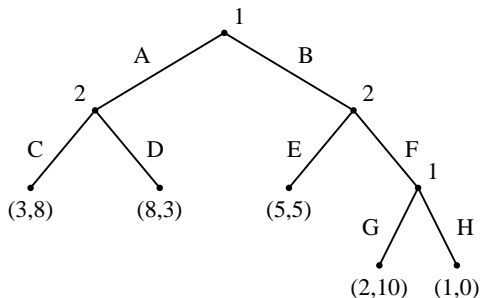
$$\prod_{h \in H, \rho(h)=i} \chi(h)$$

Pure Strategies Example



What are the pure strategies for player 2?

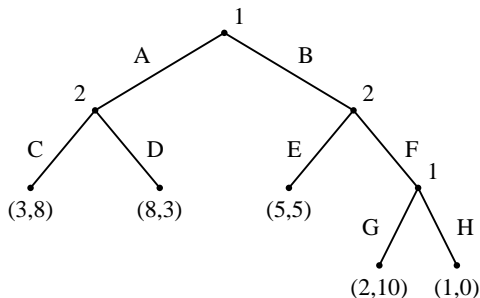
Pure Strategies Example



What are the pure strategies for player 2?

- $S_2 = \{(C, E); (C, F); (D, E); (D, F)\}$

Pure Strategies Example

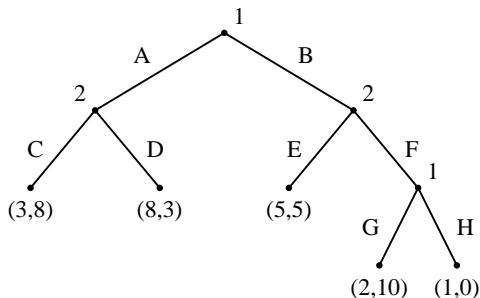


What are the pure strategies for player 2?

- $S_2 = \{(C, E); (C, F); (D, E); (D, F)\}$

What are the pure strategies for player 1?

Pure Strategies Example



What are the pure strategies for player 2?

- $S_2 = \{(C, E); (C, F); (D, E); (D, F)\}$

What are the pure strategies for player 1?

- $S_1 = \{(B, G); (B, H), (A, G), (A, H)\}$
- This is true even though, conditional on taking A , the choice between G and H will never have to be made

Nash Equilibria

Given our new definition of pure strategy, we are able to reuse our old definitions of:

- mixed strategies
- best response
- Nash equilibrium

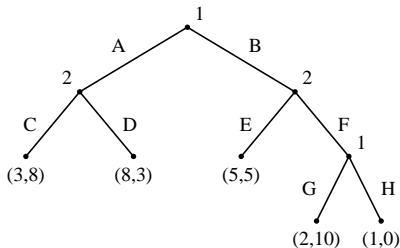
Theorem

Every perfect information game in extensive form has a PSNE

This is easy to see, since the players move sequentially.

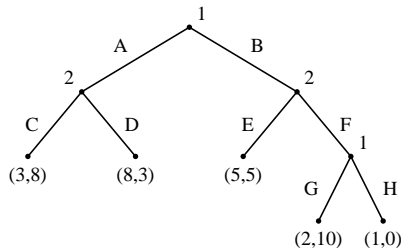
Induced Normal Form

- In fact, the connection to the normal form is even tighter
 - we can “convert” an extensive-form game into normal form



Induced Normal Form

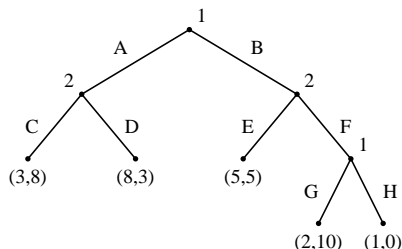
- In fact, the connection to the normal form is even tighter
 - we can “convert” an extensive-form game into normal form



	<i>CE</i>	<i>CF</i>	<i>DE</i>	<i>DF</i>
<i>AG</i>	3, 8	3, 8	8, 3	8, 3
<i>AH</i>	3, 8	3, 8	8, 3	8, 3
<i>BG</i>	5, 5	2, 10	5, 5	2, 10
<i>BH</i>	5, 5	1, 0	5, 5	1, 0

Induced Normal Form

- In fact, the connection to the normal form is even tighter
 - we can “convert” an extensive-form game into normal form

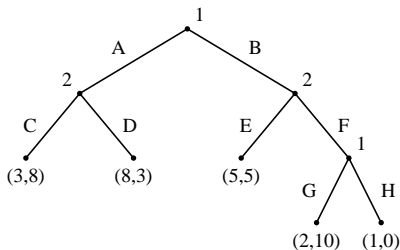


	<i>CE</i>	<i>CF</i>	<i>DE</i>	<i>DF</i>
<i>AG</i>	3, 8	3, 8	8, 3	8, 3
<i>AH</i>	3, 8	3, 8	8, 3	8, 3
<i>BG</i>	5, 5	2, 10	5, 5	2, 10
<i>BH</i>	5, 5	1, 0	5, 5	1, 0

- this illustrates the lack of compactness of the normal form
 - games aren't always this small
 - even here we write down 16 payoff pairs instead of 5

Induced Normal Form

- In fact, the connection to the normal form is even tighter
 - we can “convert” an extensive-form game into normal form

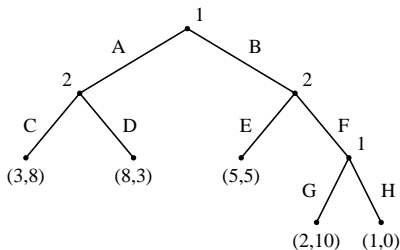


	<i>CE</i>	<i>CF</i>	<i>DE</i>	<i>DF</i>
<i>AG</i>	3, 8	3, 8	8, 3	8, 3
<i>AH</i>	3, 8	3, 8	8, 3	8, 3
<i>BG</i>	5, 5	2, 10	5, 5	2, 10
<i>BH</i>	5, 5	1, 0	5, 5	1, 0

- while we can write any extensive-form game as a NF, we can't do the reverse.
 - e.g., matching pennies cannot be written as a perfect-information extensive form game

Induced Normal Form

- In fact, the connection to the normal form is even tighter
 - we can “convert” an extensive-form game into normal form

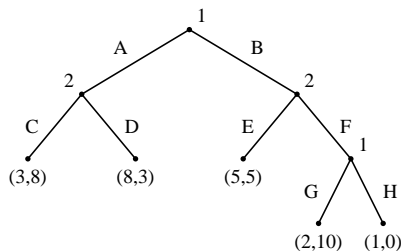


	<i>CE</i>	<i>CF</i>	<i>DE</i>	<i>DF</i>
<i>AG</i>	3, 8	3, 8	8, 3	8, 3
<i>AH</i>	3, 8	3, 8	8, 3	8, 3
<i>BG</i>	5, 5	2, 10	5, 5	2, 10
<i>BH</i>	5, 5	1, 0	5, 5	1, 0

- What are the (three) pure-strategy equilibria?

Induced Normal Form

- In fact, the connection to the normal form is even tighter
 - we can “convert” an extensive-form game into normal form

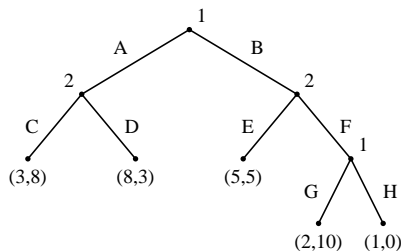


	<i>CE</i>	<i>CF</i>	<i>DE</i>	<i>DF</i>
<i>AG</i>	3, 8	3, 8	8, 3	8, 3
<i>AH</i>	3, 8	3, 8	8, 3	8, 3
<i>BG</i>	5, 5	2, 10	5, 5	2, 10
<i>BH</i>	5, 5	1, 0	5, 5	1, 0

- What are the (three) pure-strategy equilibria?
 - $(A, G), (C, F)$
 - $(A, H), (C, F)$
 - $(B, H), (C, E)$

Induced Normal Form

- In fact, the connection to the normal form is even tighter
 - we can “convert” an extensive-form game into normal form



	<i>CE</i>	<i>CF</i>	<i>DE</i>	<i>DF</i>
<i>AG</i>	3, 8	3, 8	8, 3	8, 3
<i>AH</i>	3, 8	3, 8	8, 3	8, 3
<i>BG</i>	5, 5	2, 10	5, 5	2, 10
<i>BH</i>	5, 5	1, 0	5, 5	1, 0

- What are the (three) pure-strategy equilibria?
 - $(A, G), (C, F)$
 - $(A, H), (C, F)$
 - $(B, H), (C, E)$