OPPA European Social Fund

Prague & EU: We invest in your future.

A4M33MAS - Multiagent Systems
Game Theory: Extensive Form Games

- Michal Pechoucek & Michal Jakob
- Department of Computer Science
  Czech Technical University in Prague

OTEVŘENÁ INFORMATIKA

In parts based on Kevin Leyton-Brown: Foundations of Multiagent Systems an introduction to algorithmic game theory, mechanism design and auctions

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: $\chi : H \to 2^A$ 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: $\chi : H \to 2^A$
  - Player function: $\rho : H \to N$
- 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:

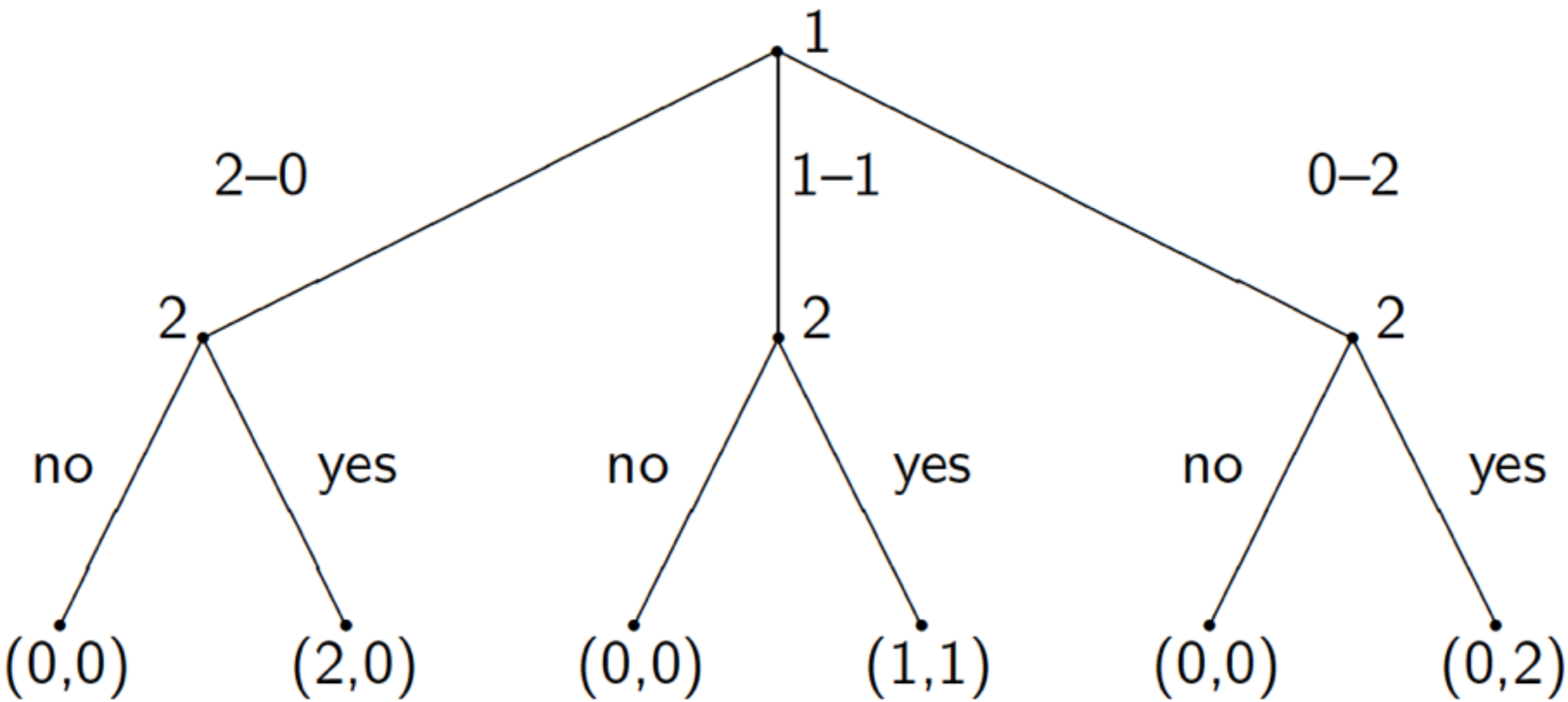
- Players: N
- Actions: A
- Choice nodes and labels for these nodes:
  - Choice nodes: H
  - Action function: $\chi : H \to 2^A$
  - Player function: $\rho : H \to N$
- Terminal nodes: Z
- Successor function: $\sigma : H \times A \to H \cup Z$ maps a choice node and an action to a new choice node or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

Example: Sharing game



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

Pure Strategy

Overall, a pure strategy for a player in a perfect-information game is a complete speciffication of which deterministic action to take at every node belonging to that player.

Pure Strategy

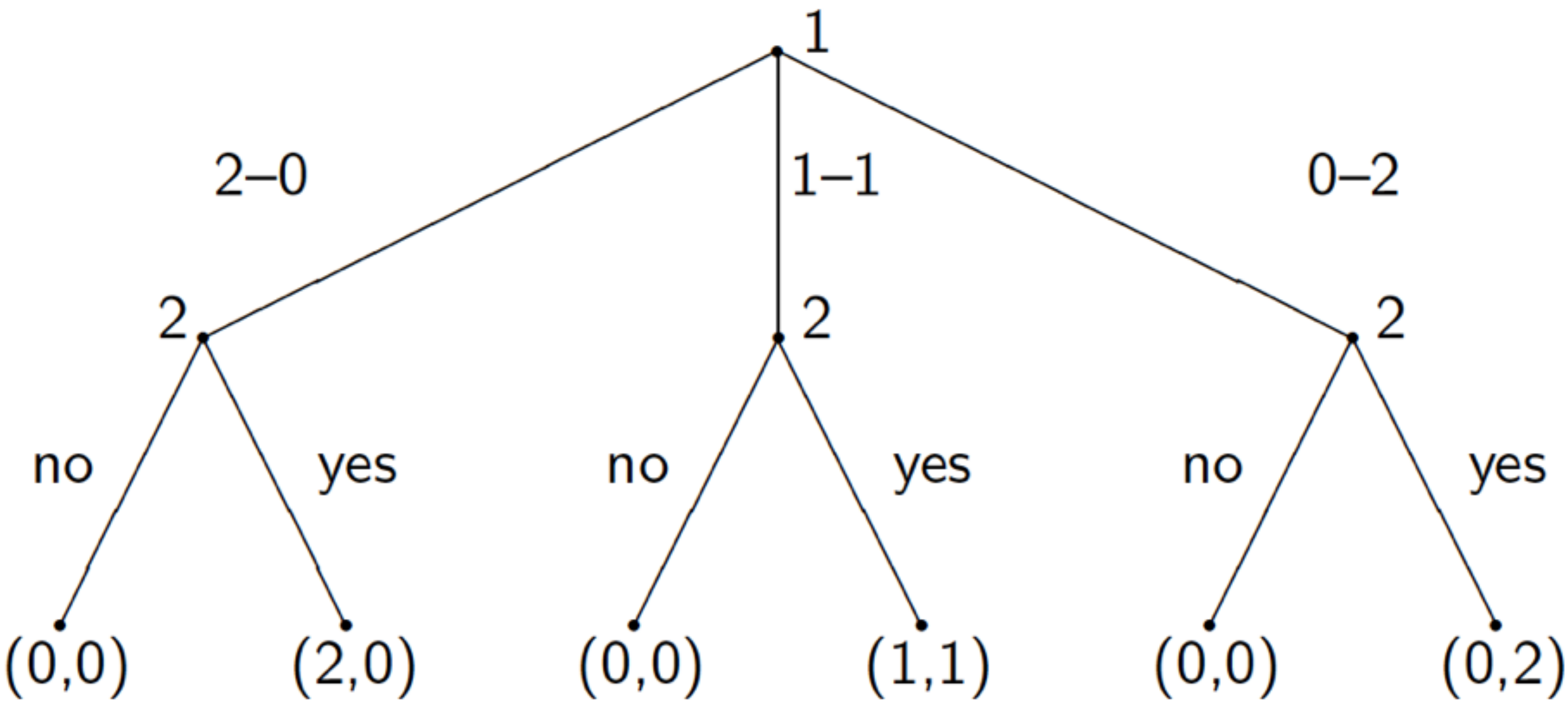
Overall, a pure strategy for a player in a perfect-information game is a complete speciffication of which deterministic action to take at every node belonging to that player.

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

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

Example: Sharing game



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

Pure Strategy

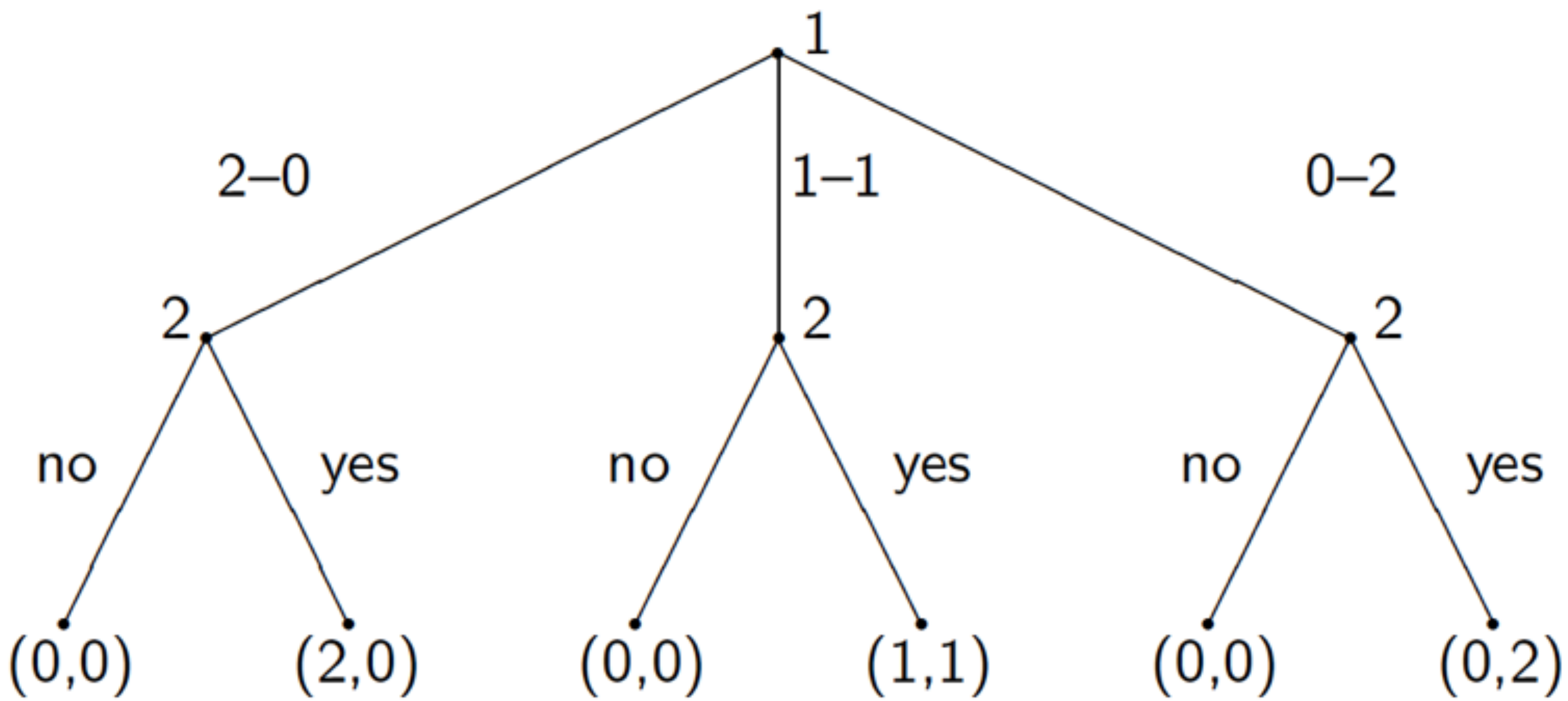
Overall, a pure strategy for a player in a perfect-information game is a complete speciffication of which deterministic action to take at every node belonging to that player.

Definition (pure strategies)

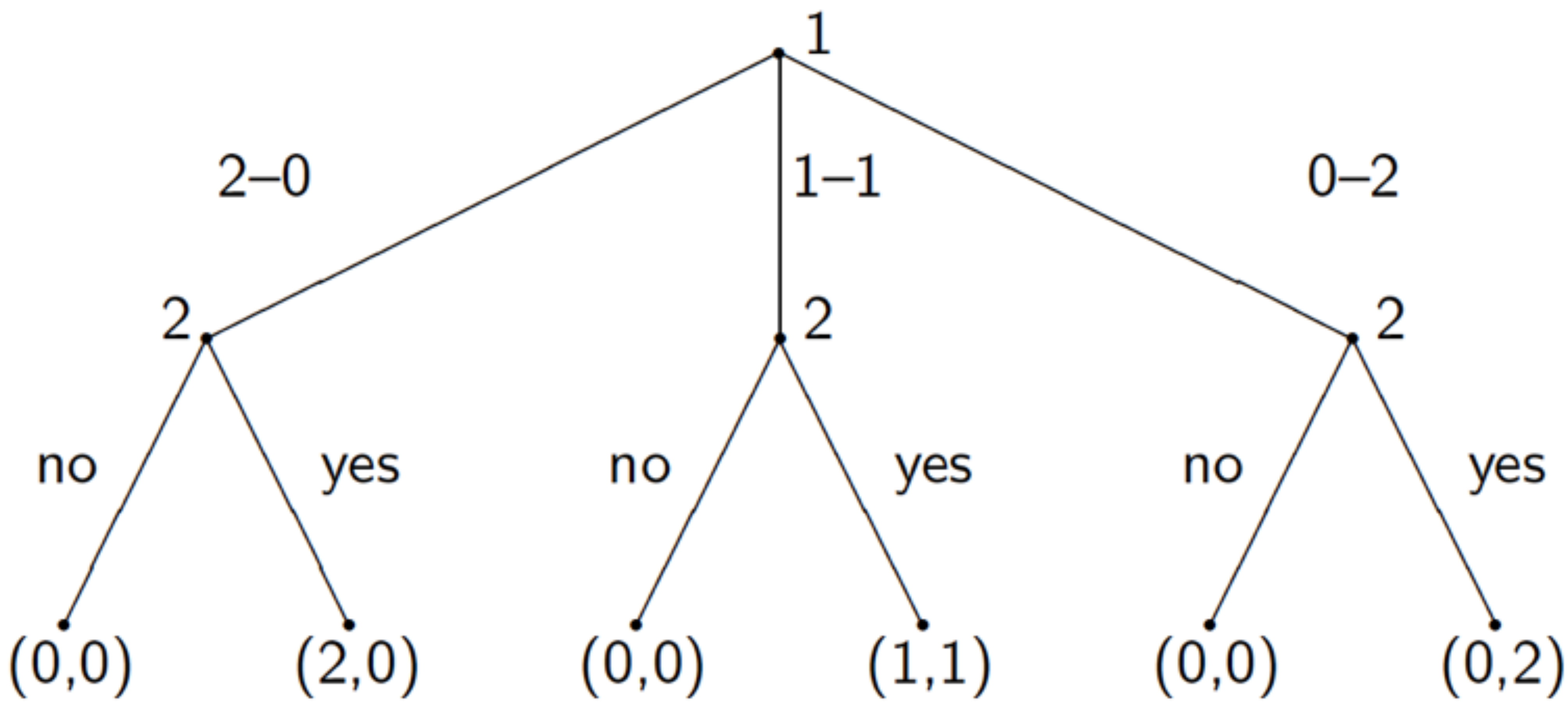
Let $G = (N, A, H, Z, \chi, \rho, \sigma, u)$ be a perfect-information extensive-form game. Then the pure strategies of player i consist of the cross product

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

Example: Sharing game



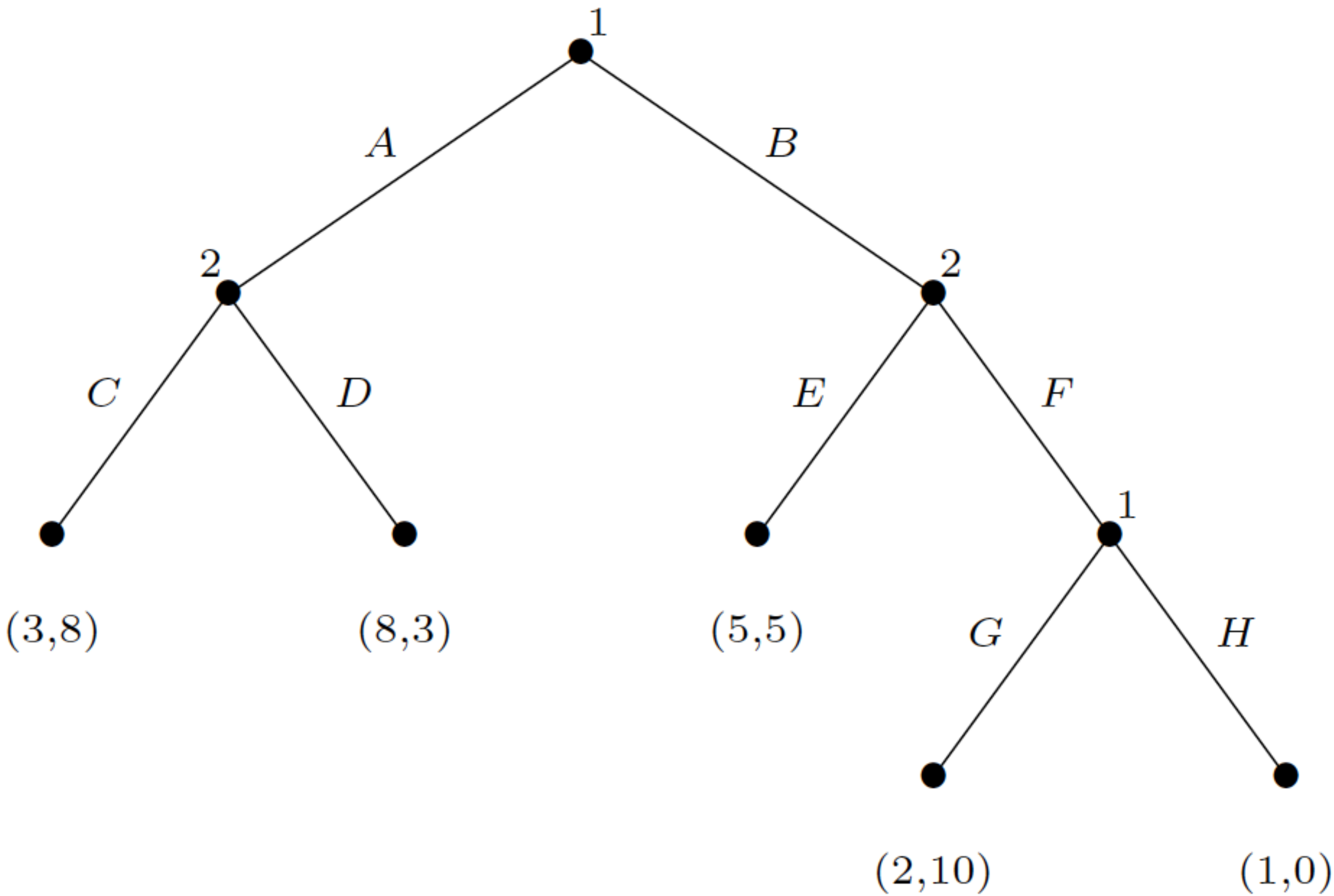
Example: Sharing game



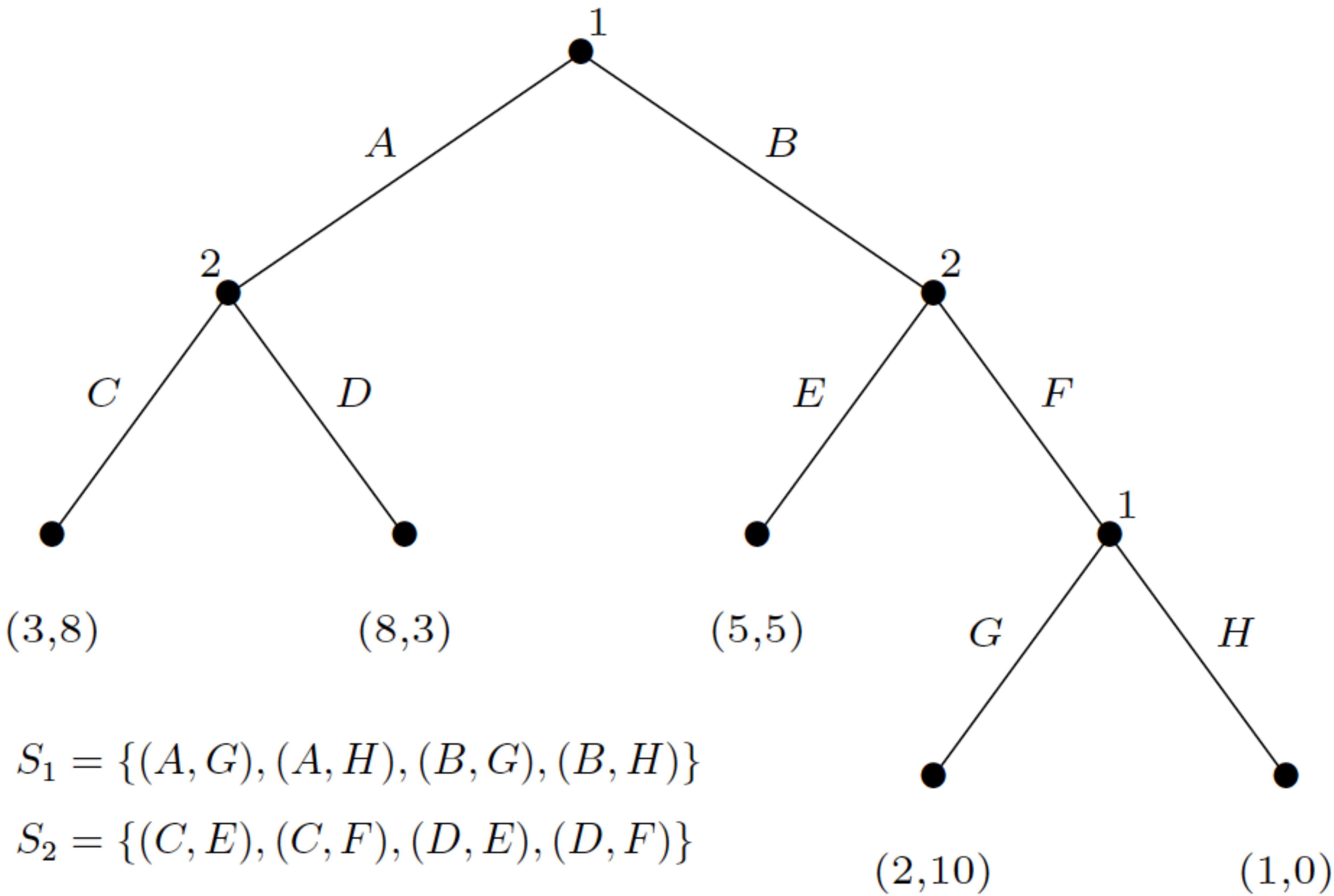
$S_1 = \{2\text{–}0, 1\text{–}1, 0\text{–}2\}$

$S_2 = \{(yes, yes, yes), (yes, yes, no), (yes, no, yes),$
$(no, yes, no), (no, no, yes), (no, no, no)\}$
$(yes, no, no), (no, yes, yes)\}$

Example



Example



$S_1 = \{(A, G), (A, H), (B, G), (B, H)\}$

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

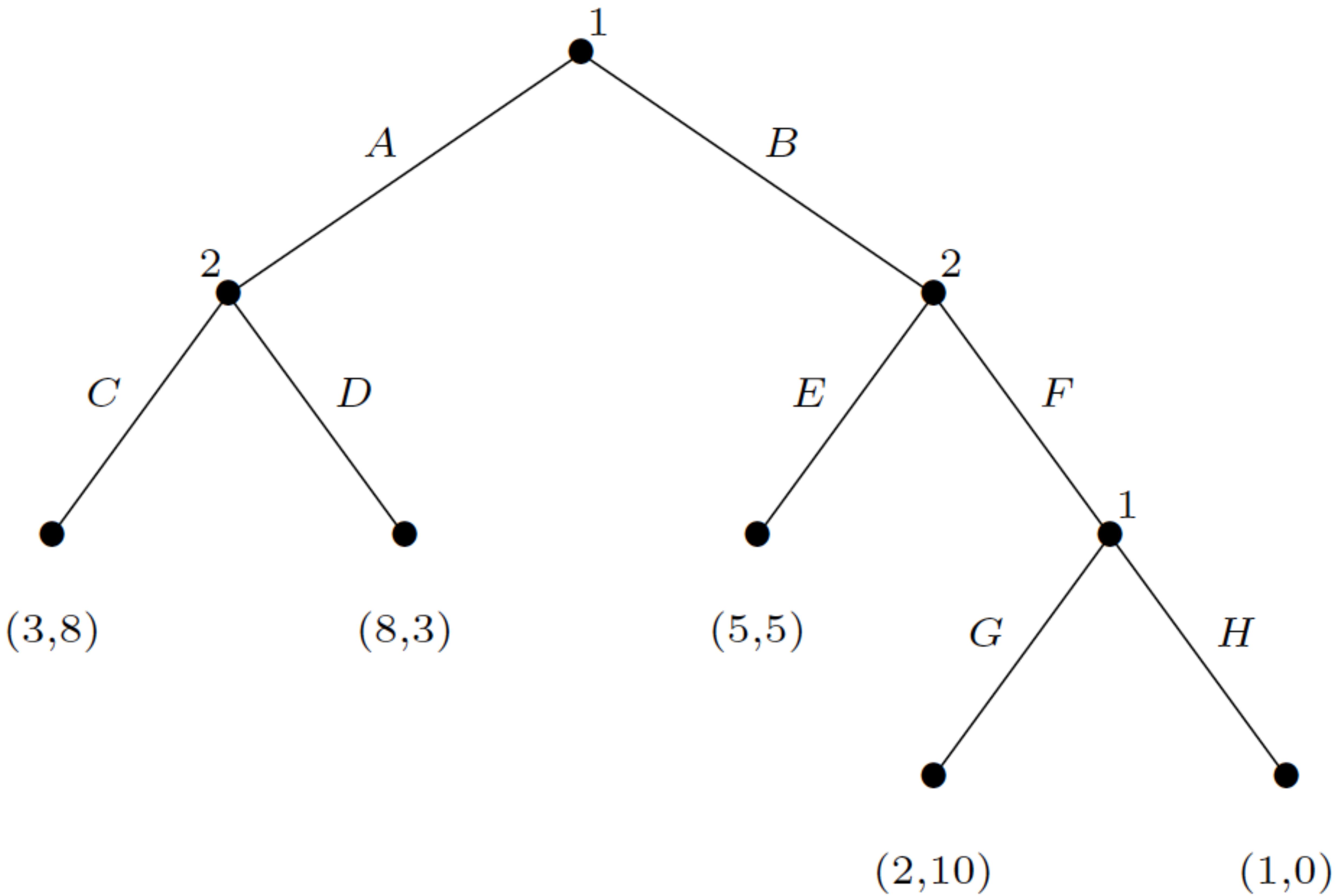
Nash Equilibrium

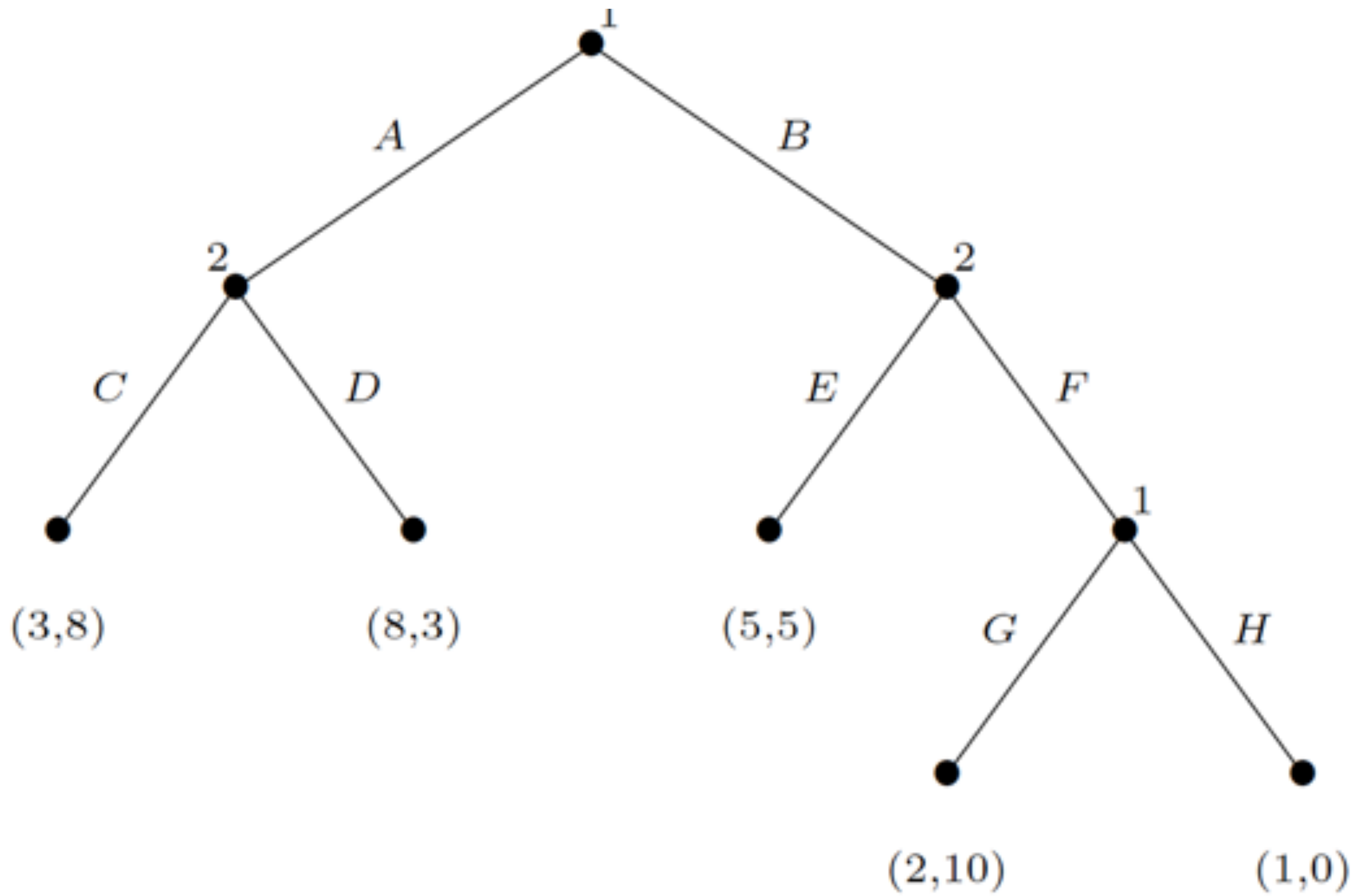
Theorem

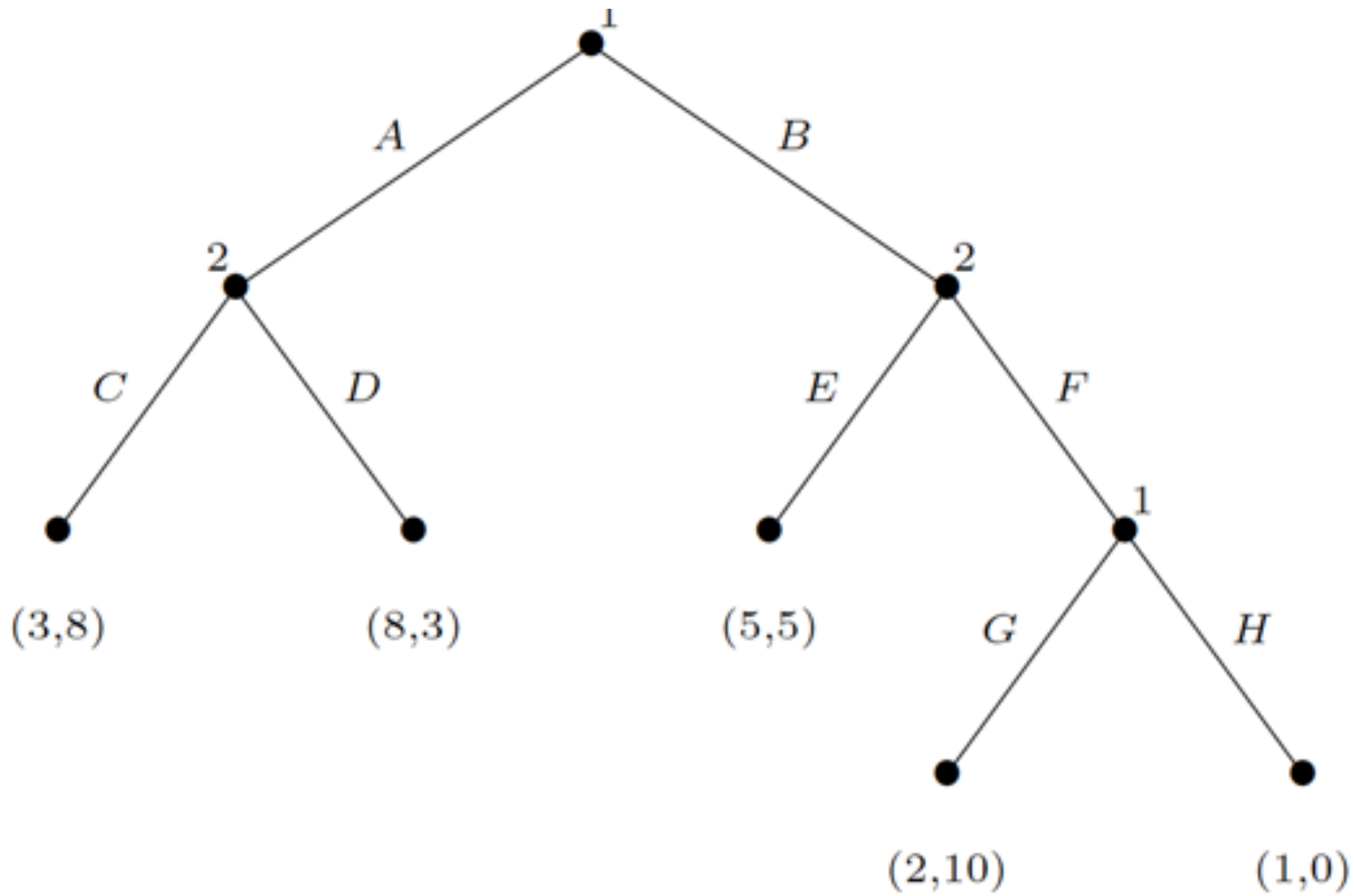
Every perfect information game in extensive form has a Pure Strategy Nash Equilibrium

- Why?

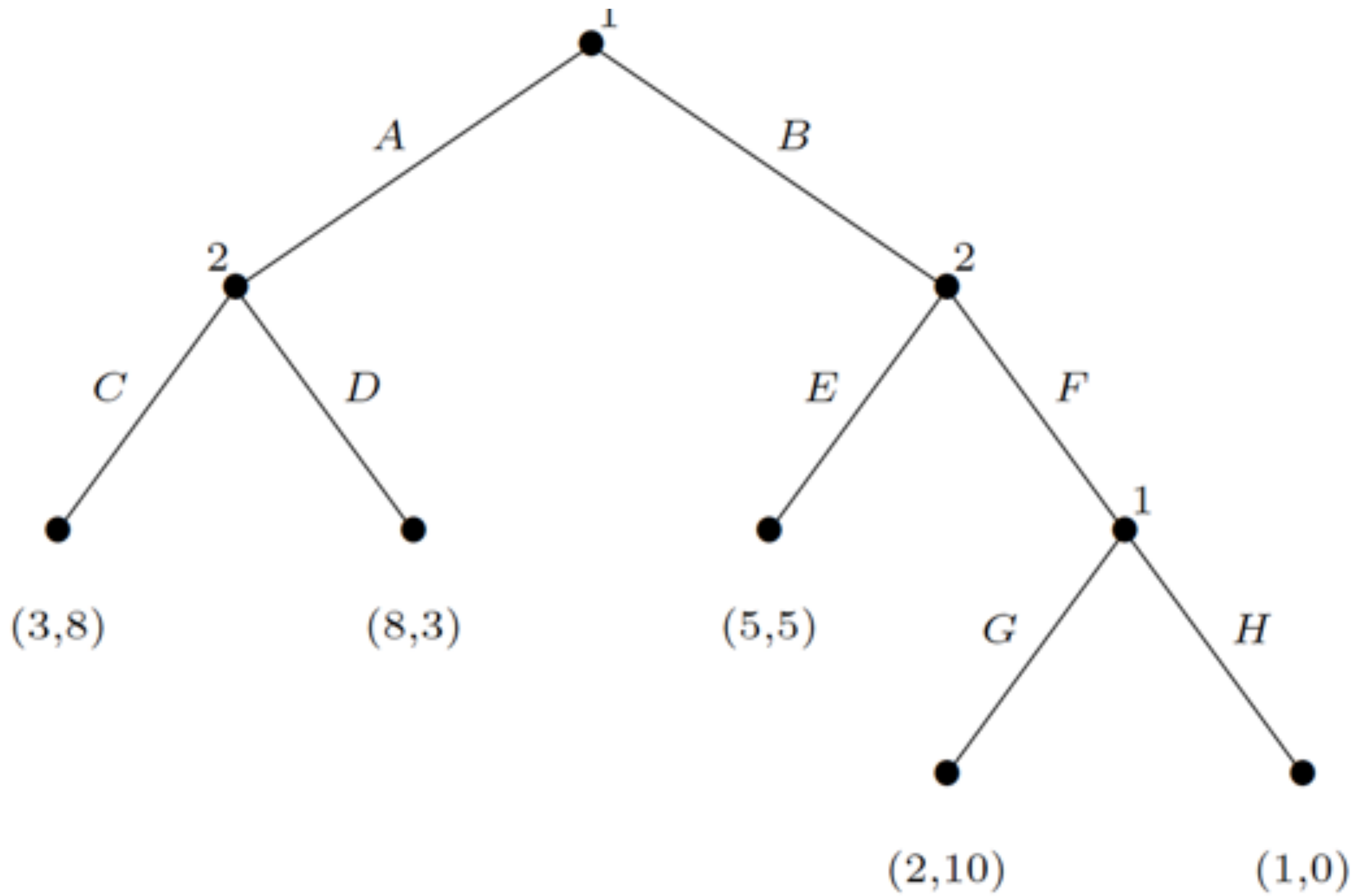
Induced normal



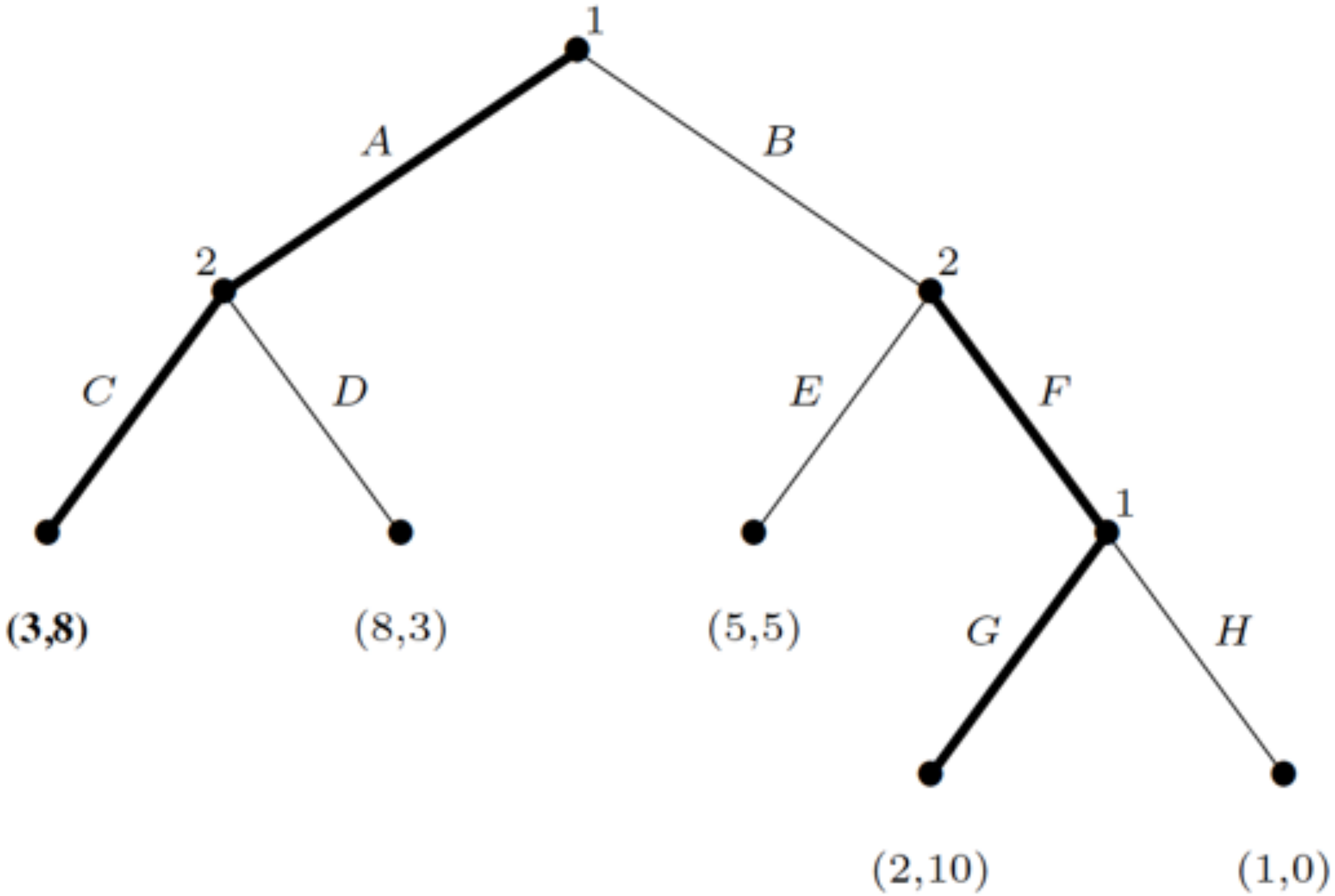

1
A
B
2
2
C
D
E
F
1
(3,8)
(8,3)
(5,5)
G
H
(2,10)
(1,0)



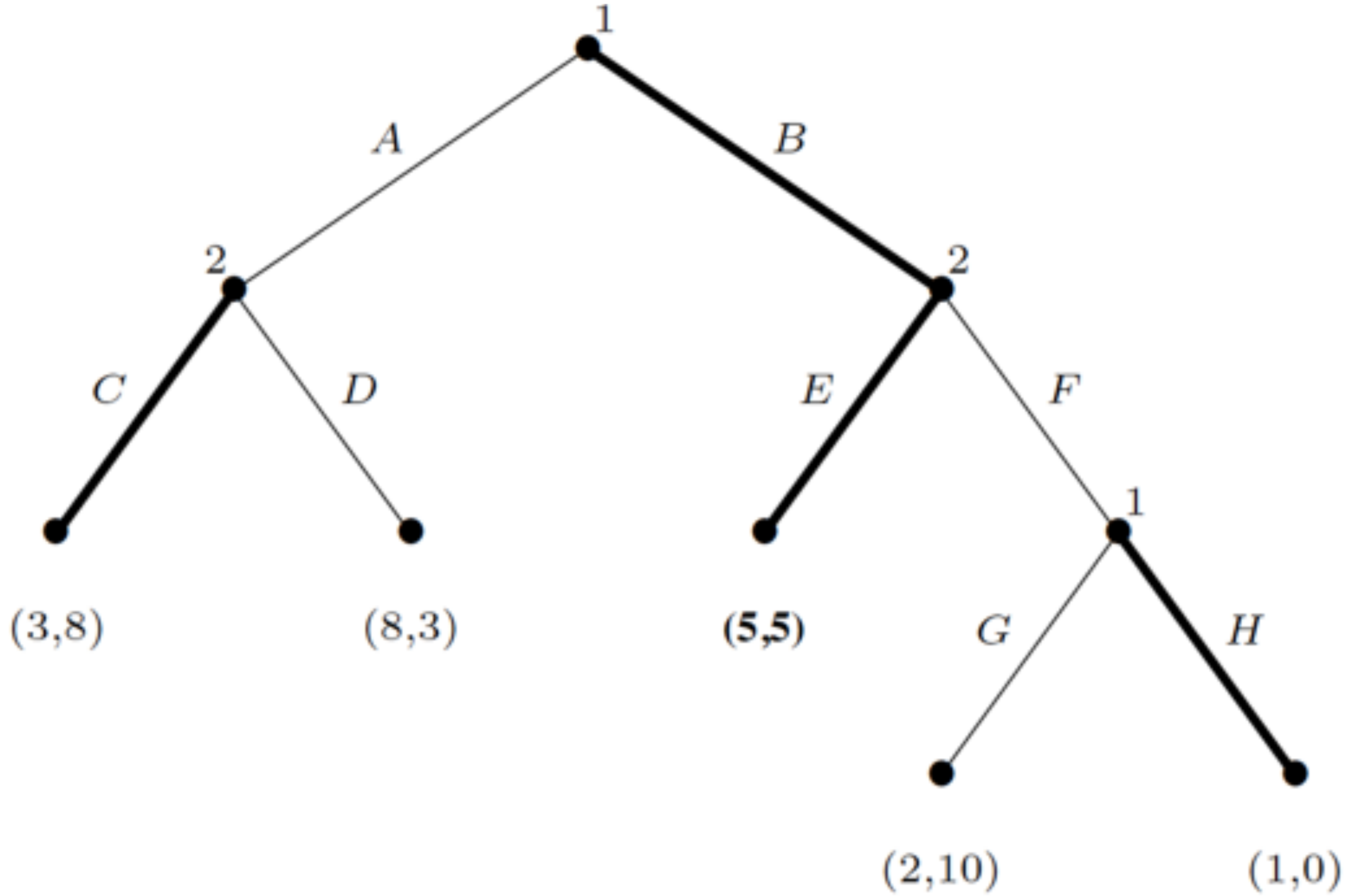
	(C,E)	(C,F)	(D,E)	(D,F)
(A,G)	3, 8	3, 8	8, 3	8, 3
(A,H)	3, 8	3, 8	8, 3	8, 3
(B,G)	5, 5	2, 10	5, 5	2, 10
(B,H)	5, 5	1, 0	5, 5	1, 0



	(C,E)	(C,F)	(D,E)	(D,F)
(A,G)	3, 8	3, 8	8, 3	8, 3
(A,H)	3, 8	3, 8	8, 3	8, 3
(B,G)	5, 5	2, 10	5, 5	2, 10
(B,H)	5, 5	1, 0	5, 5	1, 0

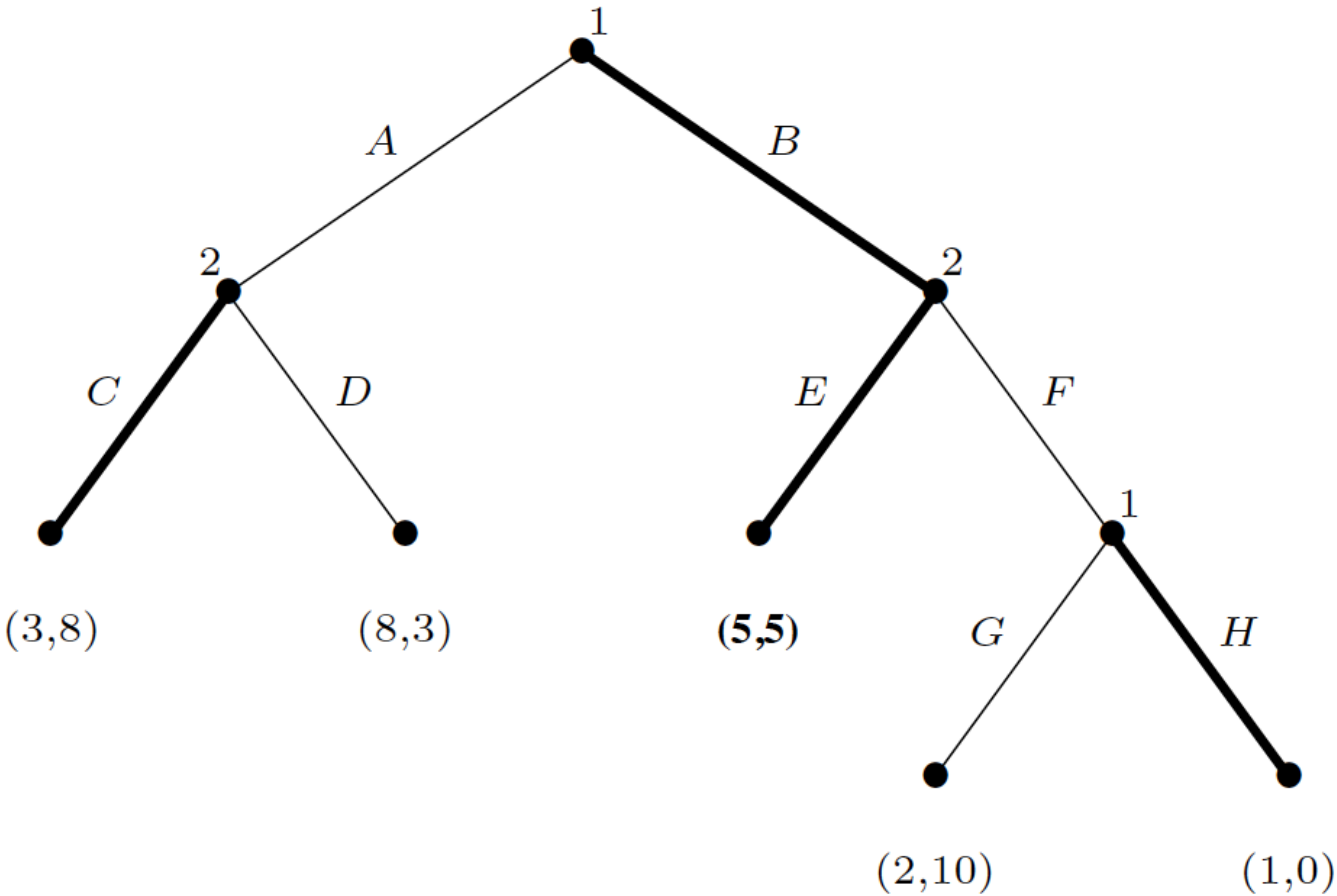


	(C,E)	(C,F)	(D,E)	(D,F)
(A,G)	3, 8	3, 8	8, 3	8, 3
(A,H)	3, 8	3, 8	8, 3	8, 3
(B,G)	5, 5	2, 10	5, 5	2, 10
(B,H)	5, 5	1, 0	5, 5	1, 0

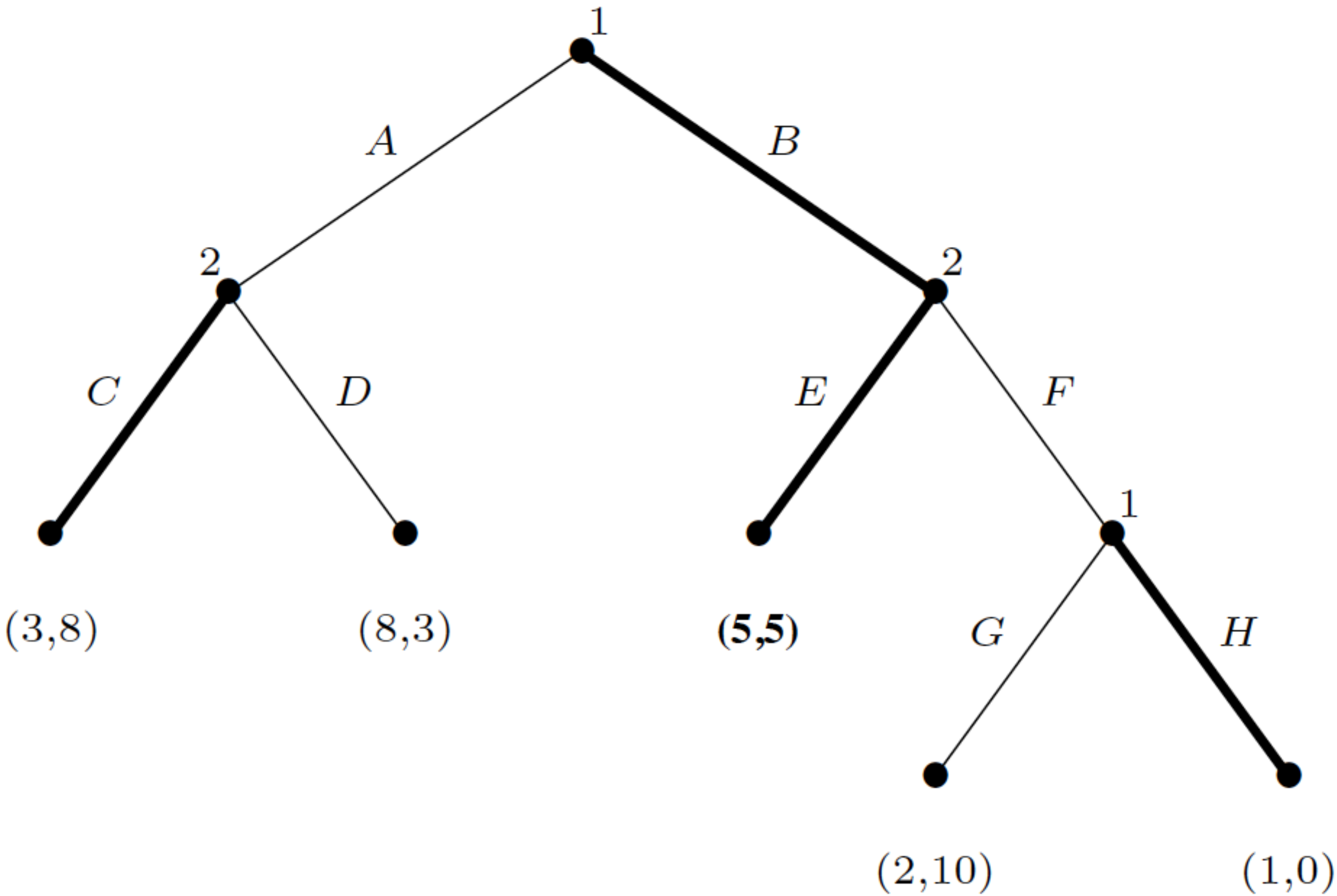


	(C,E)	(C,F)	(D,E)	(D,F)
(A,G)	3, 8	3, 8	8, 3	8, 3
(A,H)	3, 8	3, 8	8, 3	8, 3
(B,G)	5, 5	2, 10	5, 5	2, 10
(B,H)	5, 5	1, 0	5, 5	1, 0

Nash Equilibrium?



Subgame perfect NE?



Subgame perfect NE?

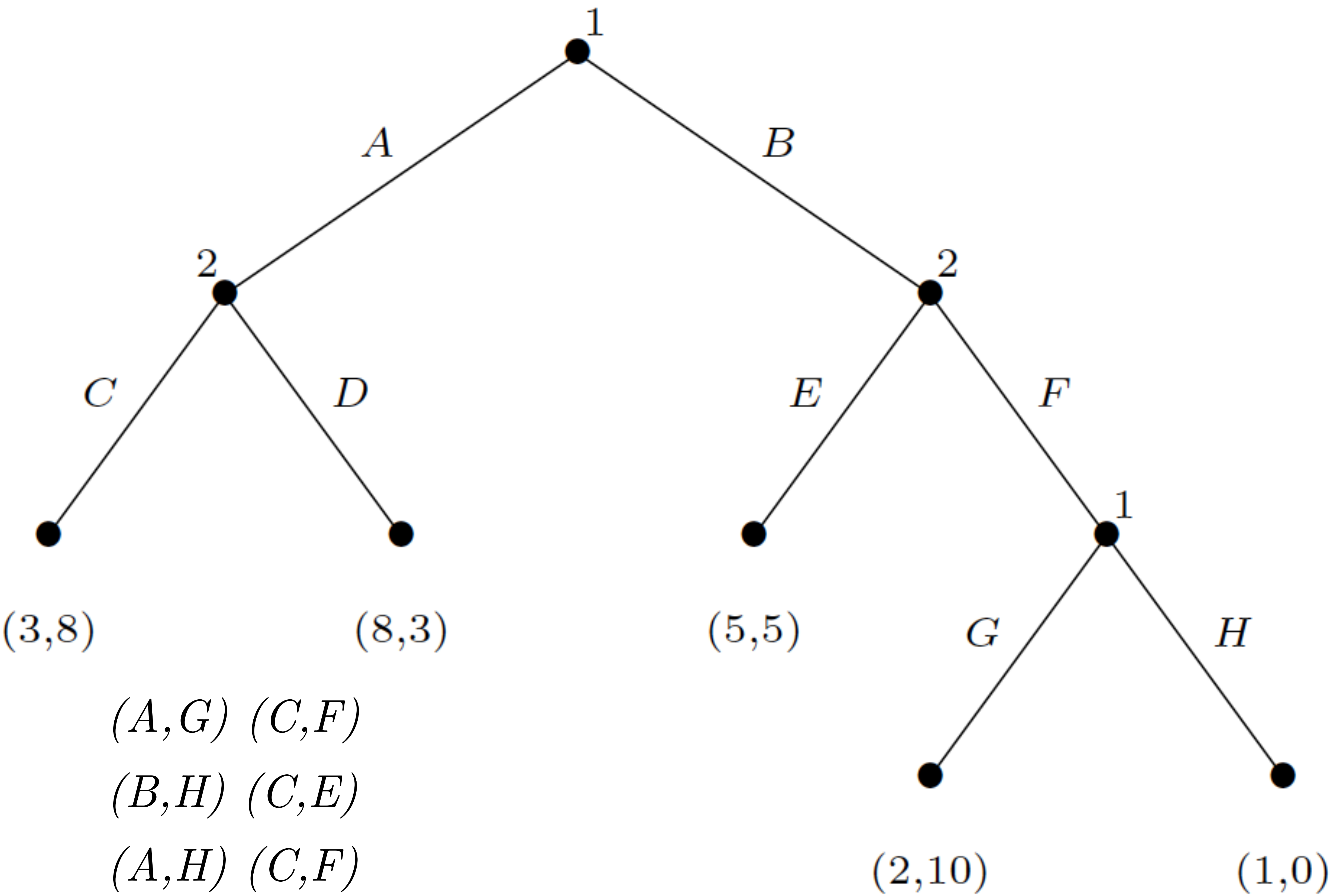
- The subgame of G rooted at h is the restriction of G to the descendents of H. The set of subgames of G is defined by the subgames of G rooted at each of the nodes in G.

Definition (Subgame perfect Nash Equilibrium):
s is a subgame perfect equilibrium of G i for any subgame G' of G, the restriction of s to G' is a Nash equilibrium of G'.

Notes:

- since G is its own subgame, every SPE is a NE.
- this definition rules out non-credible threats

Subgame Perfect Nash Equilibrium

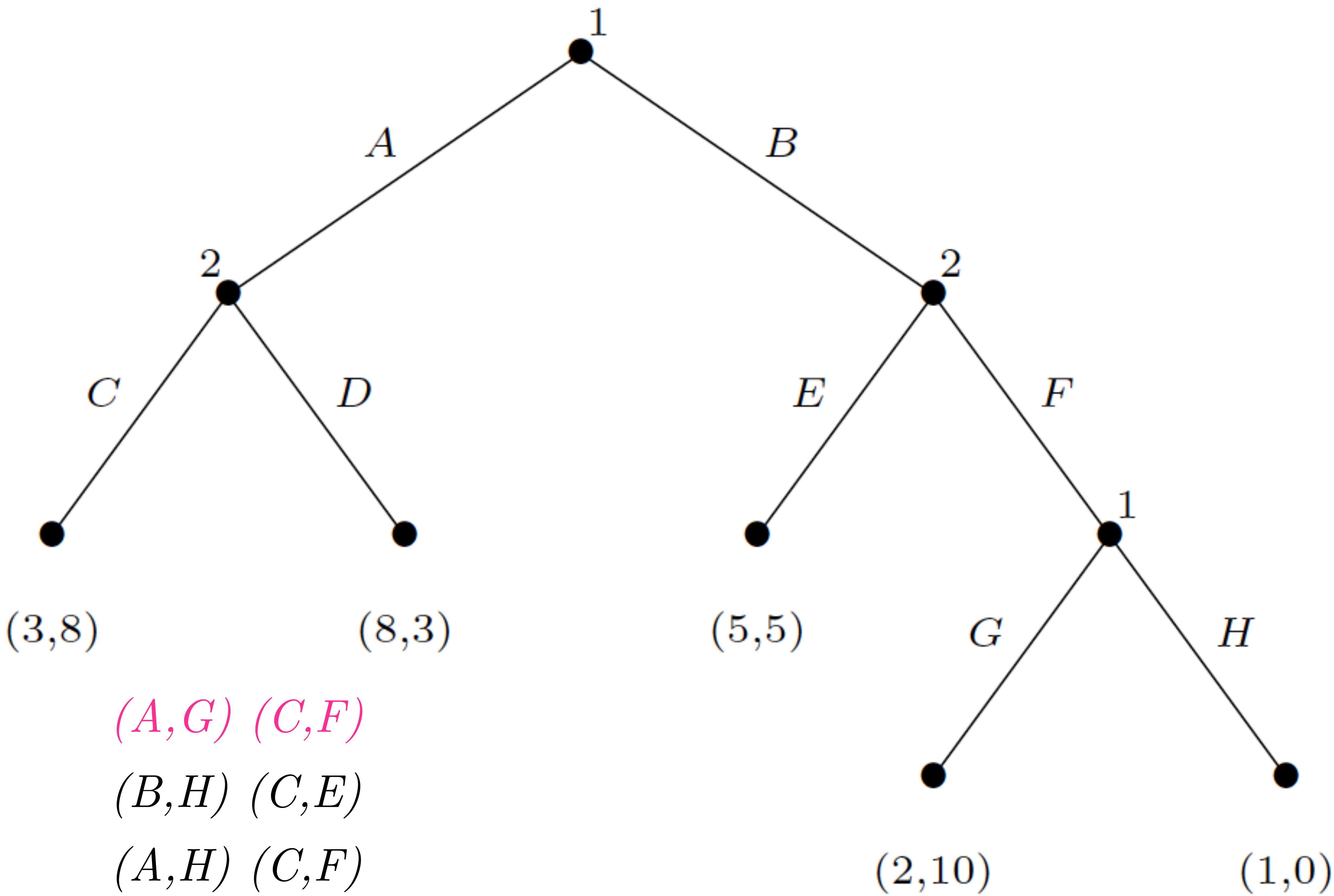


(A, G) (C, F)

(B, H) (C, E)

(A, H) (C, F)

Subgame Perfect Nash Equilibrium

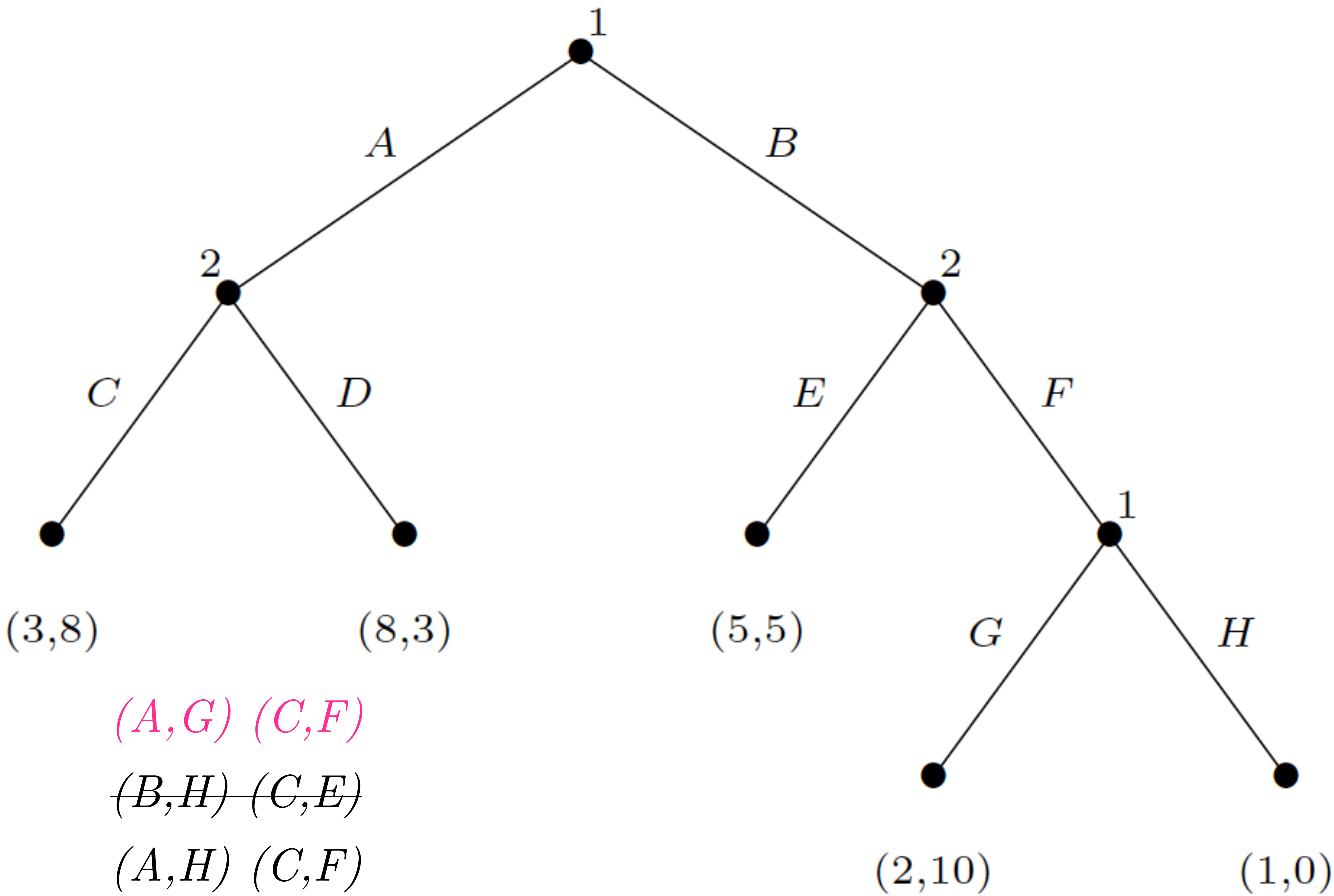


(A,G) (C,F)

(B,H) (C,E)

(A,H) (C,F)

Subgame Perfect Nash Equilibrium

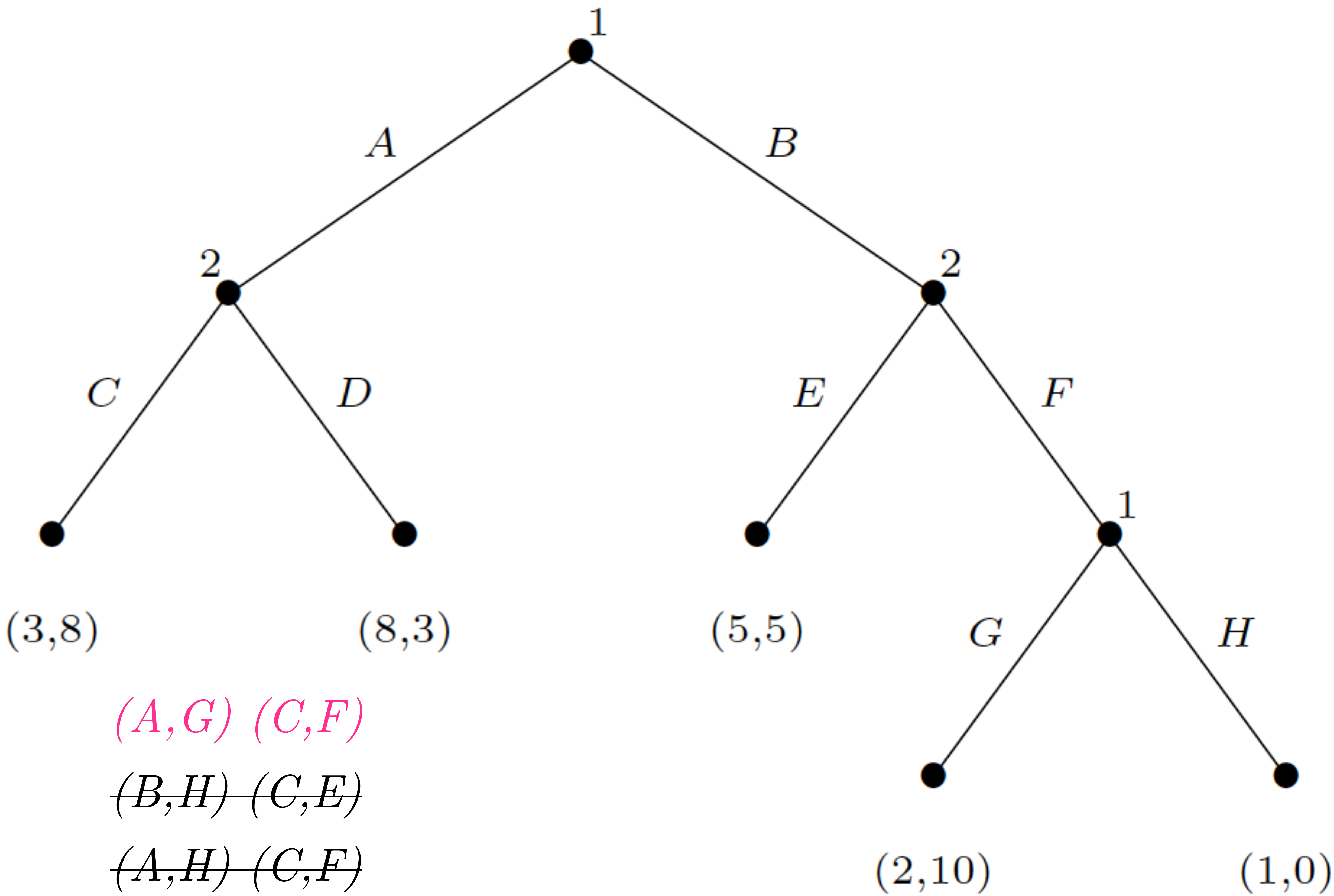


$(A,G)\ (C,F)$

~~$(B,H)\ (C,E)$~~

$(A,H)\ (C,F)$

Subgame Perfect Nash Equilibrium



$(A,G)\ (C,F)$

~~$(B,H)\ (C,E)$~~

~~$(A,H)\ (C,F)$~~

Computing SPE: Backward Induction

- Idea: Identify the equilibria in the bottom-most trees, and adopt these as one moves up the tree

function $\textsc{BackwardInduction}$ (node h) **returns** $u(h)$
if $h \in Z$ **then**
  return $u(h)$
$best\_util \leftarrow -\infty$
forall $a \in \chi(h)$ **do**
  $util\_at\_child \leftarrow \textsc{BackwardInduction}(\sigma(h, a))$
  if $util\_at\_child_{\rho(h)} > best\_util_{\rho(h)}$ **then**
    $best\_util \leftarrow util\_at\_child$
return $best\_util$

- The procedure doesn't return an equilibrium strategy, but rather labels each node with a vector of real numbers. This labeling can be seen as an extension of the game's utility function to the non-terminal nodes
- The equilibrium strategies: take the best action at each node

Computing SPE: Minimax Algorithm

- Idea: For zero-sum games, Backward Induction is called the Minimax algorithm.
- It is enough to store one number per nodeand speed things up by pruning nodes that will never be reached: alpha-beta pruning

Computing SPE: Minimax Algorithm

- It is enough to store one number per nodeand speed things up by pruning nodes that will never be reached: alpha-beta pruning

function $\textsc{AlphaBetaPruning}$ (node h, real α, real β) **returns** $u_1(h)$
if $h \in Z$ **then**
  return $u_1(h)$ // h is a terminal node
$best\_util \leftarrow (2\rho(h) - 3) \times \infty$ // $-\infty$ for player 1; ∞ for player 2
forall $a \in \chi(h)$ **do**
  if $\rho(h) = 1$ **then**
    $best\_util \leftarrow \max(best\_util, \textsc{AlphaBetaPruning}(\sigma(h,a), \alpha, \beta))$
    if $best\_util \geq \beta$ **then**
      return $best\_util$
    $\alpha \leftarrow \max(\alpha, best\_util)$
  else
    $best\_util \leftarrow \min(best\_util, \textsc{AlphaBetaPruning}(\sigma(h,a), \alpha, \beta))$
    if $best\_util \leq \alpha$ **then**
      return $best\_util$
    $\beta \leftarrow \min(\beta, best\_util)$
return $best\_util$

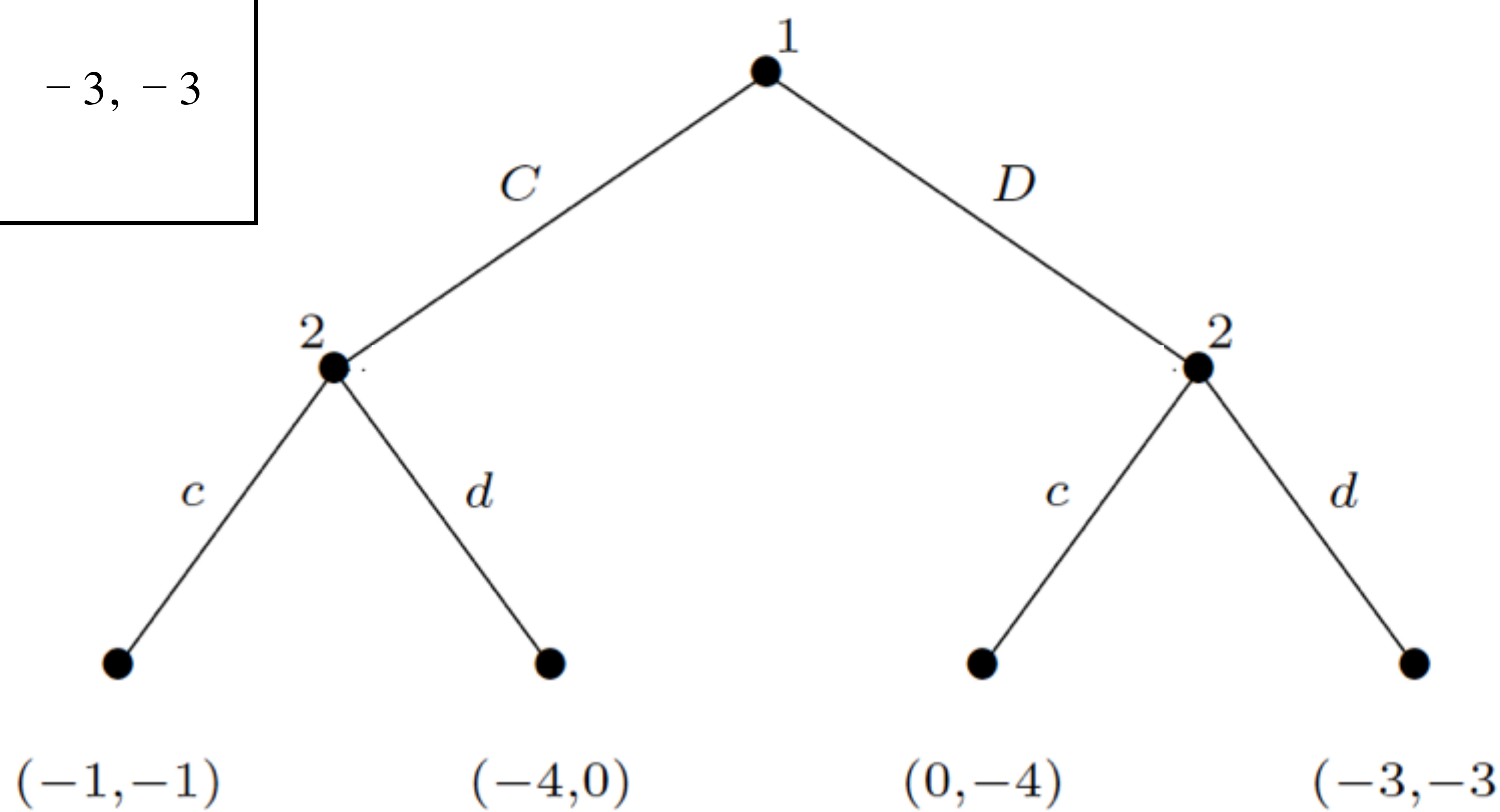
Imperfect Information EF Games

Imperfect Information EF Games

	Cooperate	Defect
Cooperate	$-1, -1$	$-4, 0$
Defect	$0, -4$	$-3, -3$

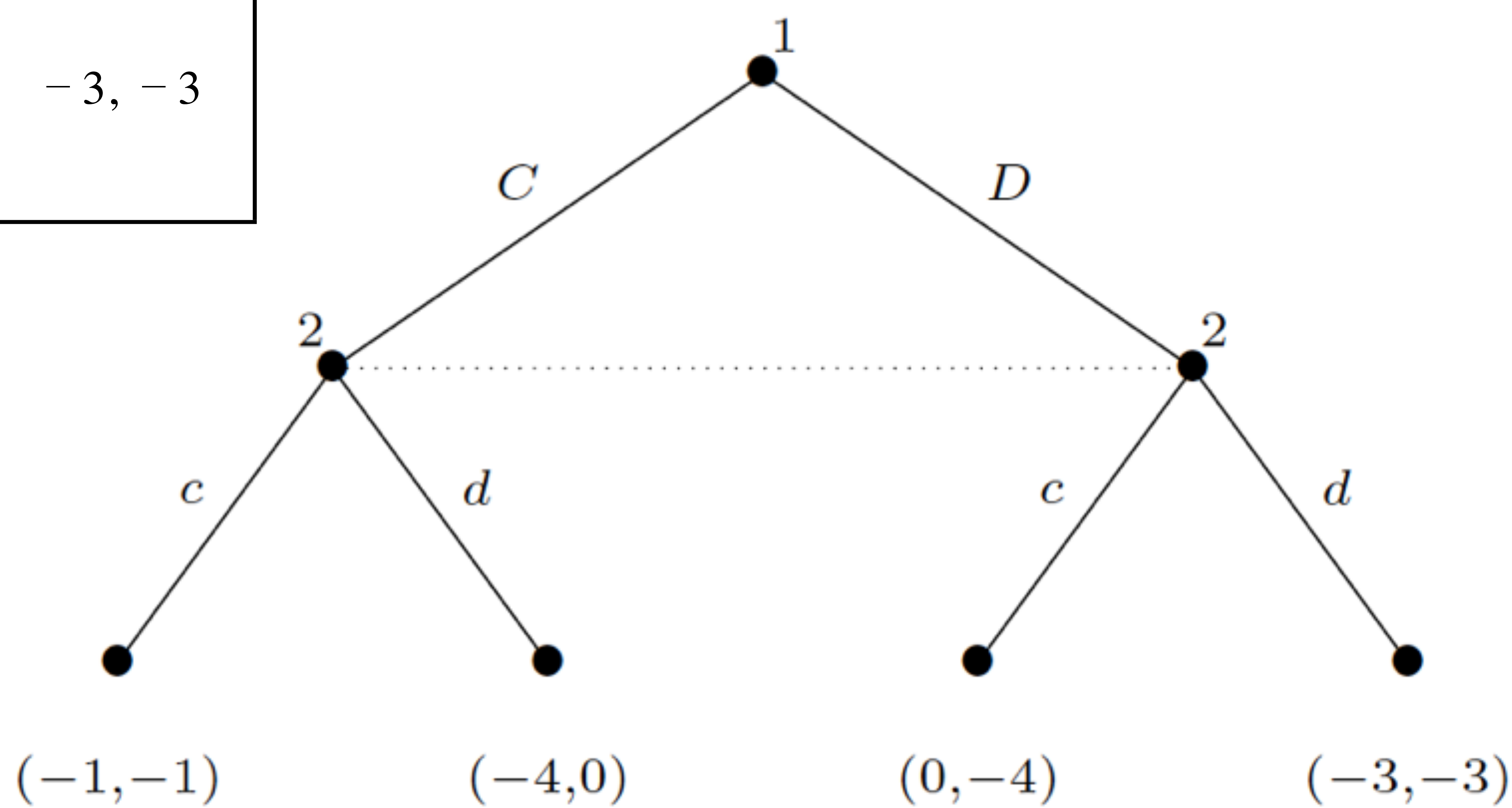
Imperfect Information EF Games

	Cooperate	Defect
Cooperate	$-1, -1$	$-4, 0$
Defect	$0, -4$	$-3, -3$



Imperfect Information EF Games

	Cooperate	Defect
Cooperate	$-1, -1$	$-4, 0$
Defect	$0, -4$	$-3, -3$



Imperfect Information EF Games

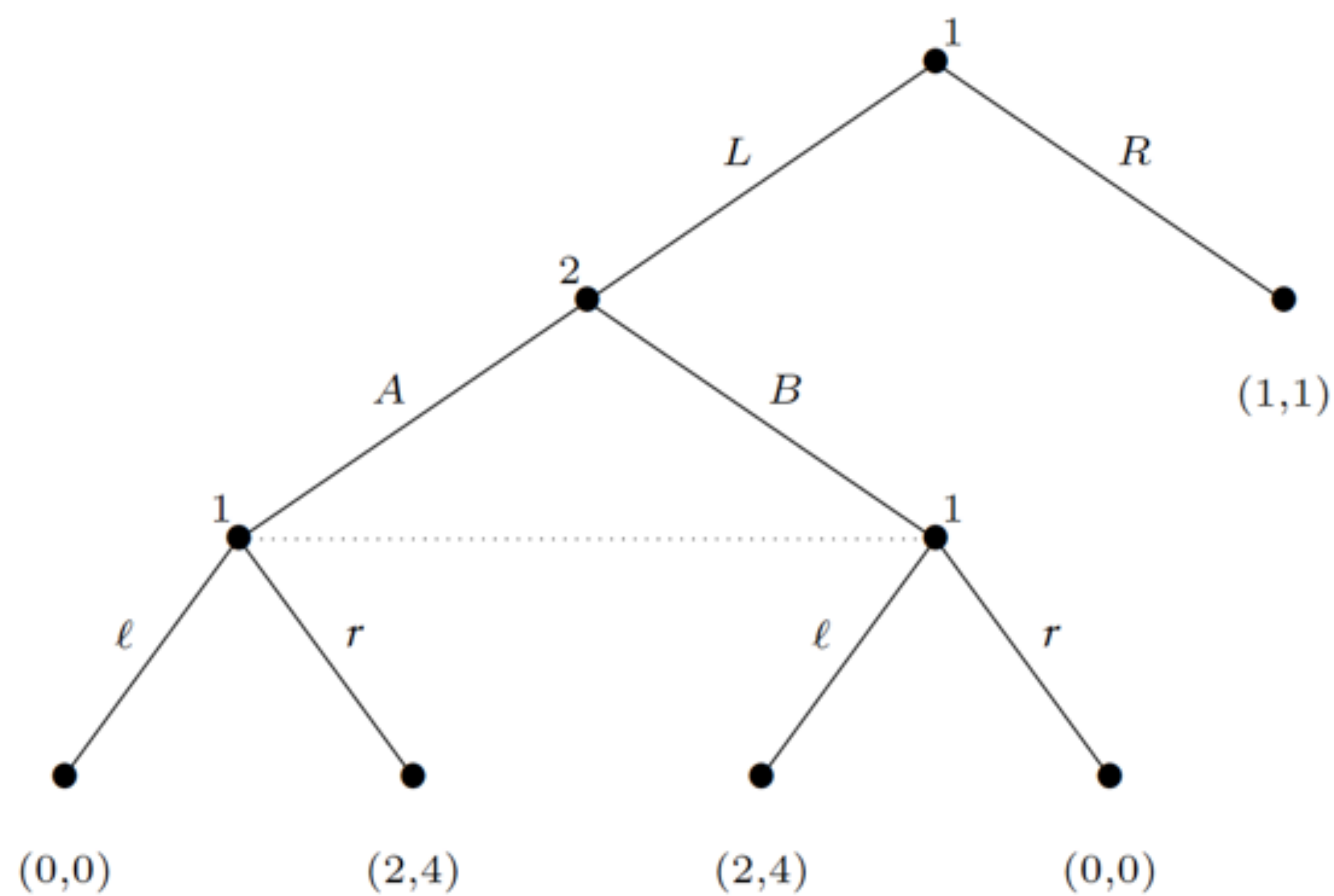
Definition 5.2.1 (Imperfect-information game) *An imperfect-information game (in extensive form) is a tuple* $(N, A, H, Z, \chi, \rho, \sigma, u, I)$, *where:*

- $(N, A, H, Z, \chi, \rho, \sigma, u)$ *is a perfect-information extensive-form game; and*
- $I = (I_1, \ldots, I_n)$, *where* $I_i = (I_{i,1}, \ldots, I_{i,k_i})$ *is a set of equivalence classes on (i.e., a partition of)* $\{h \in H : \rho(h) = i\}$ *with the property that* $\chi(h) = \chi(h')$ *and* $\rho(h) = \rho(h')$ *whenever there exists a* j *for which* $h \in I_{i,j}$ *and* $h' \in I_{i,j}$.

Imperfect Information EF Games

Definition 5.2.1 (Imperfect-information game) *An imperfect-information game (in extensive form) is a tuple $(N, A, H, Z, \chi, \rho, \sigma, u, I)$, where:*

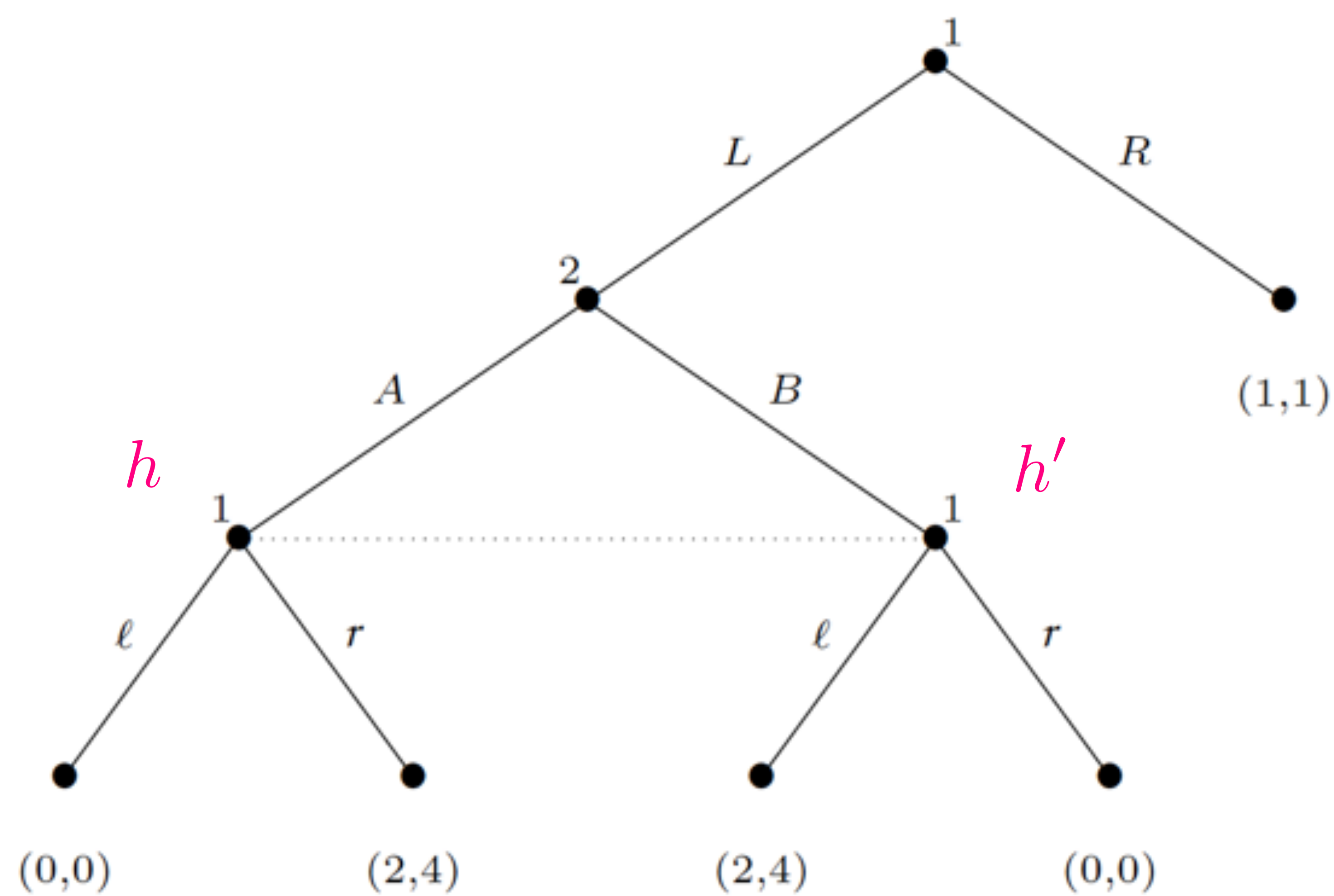
- $(N, A, H, Z, \chi, \rho, \sigma, u)$ *is a perfect-information extensive-form game; and*
- $I = (I_1, \ldots, I_n)$*, where* $I_i = (I_{i,1}, \ldots, I_{i,k_i})$ *is a set of equivalence classes on (i.e., a partition of)* $\{h \in H : \rho(h) = i\}$ *with the property that* $\chi(h) = \chi(h')$ *and* $\rho(h) = \rho(h')$ *whenever there exists a* j *for which* $h \in I_{i,j}$ *and* $h' \in I_{i,j}$.



Imperfect Information EF Games

Definition 5.2.1 (Imperfect-information game) *An imperfect-information game (in extensive form) is a tuple* $(N, A, H, Z, \chi, \rho, \sigma, u, I)$*, where:*

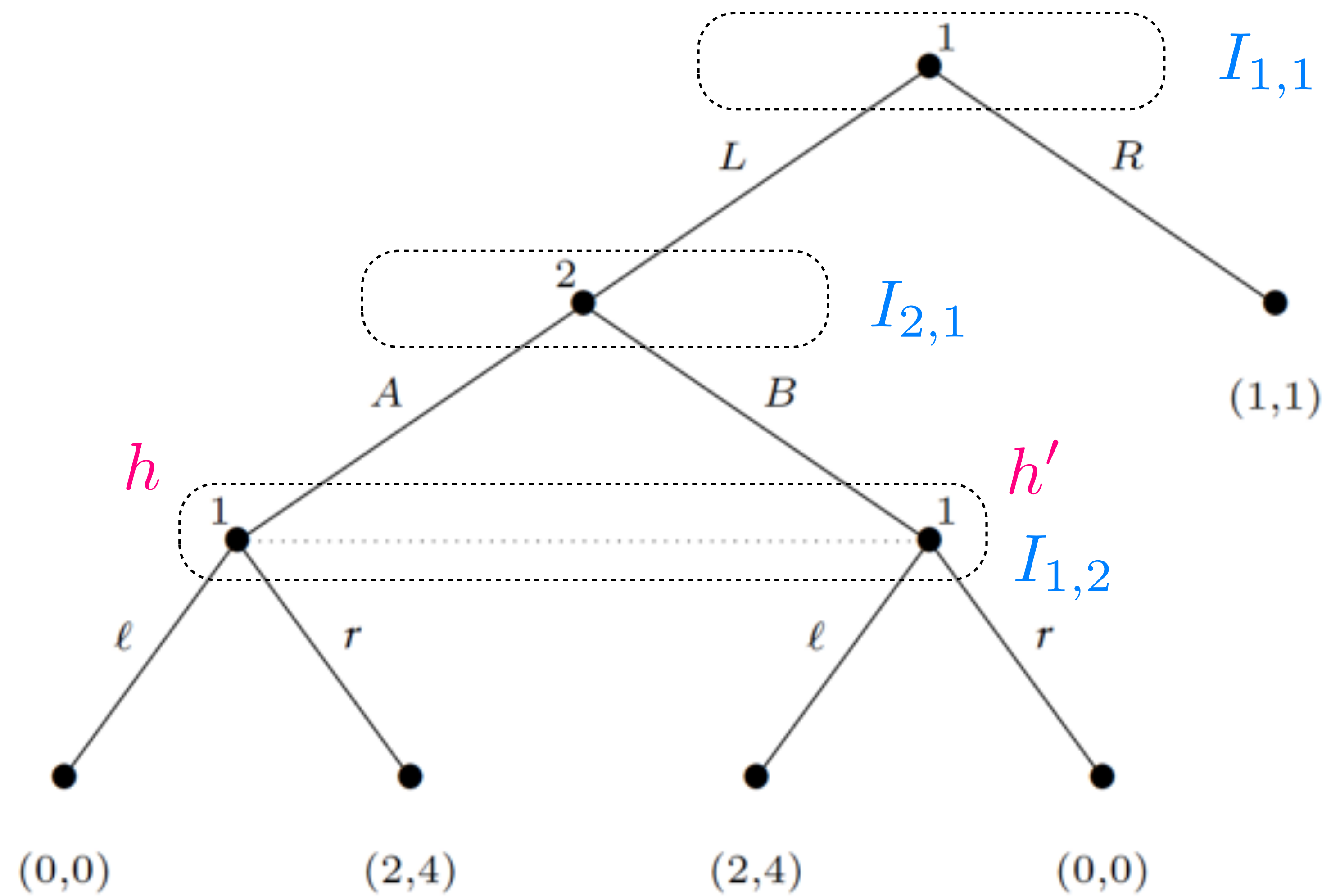
- $(N, A, H, Z, \chi, \rho, \sigma, u)$ *is a perfect-information extensive-form game; and*
- $I = (I_1, \ldots, I_n)$*, where* $I_i = (I_{i,1}, \ldots, I_{i,k_i})$ *is a set of equivalence classes on (i.e., a partition of)* $\{h \in H : \rho(h) = i\}$ *with the property that* $\chi(h) = \chi(h')$ *and* $\rho(h) = \rho(h')$ *whenever there exists a* j *for which* $h \in I_{i,j}$ *and* $h' \in I_{i,j}$.



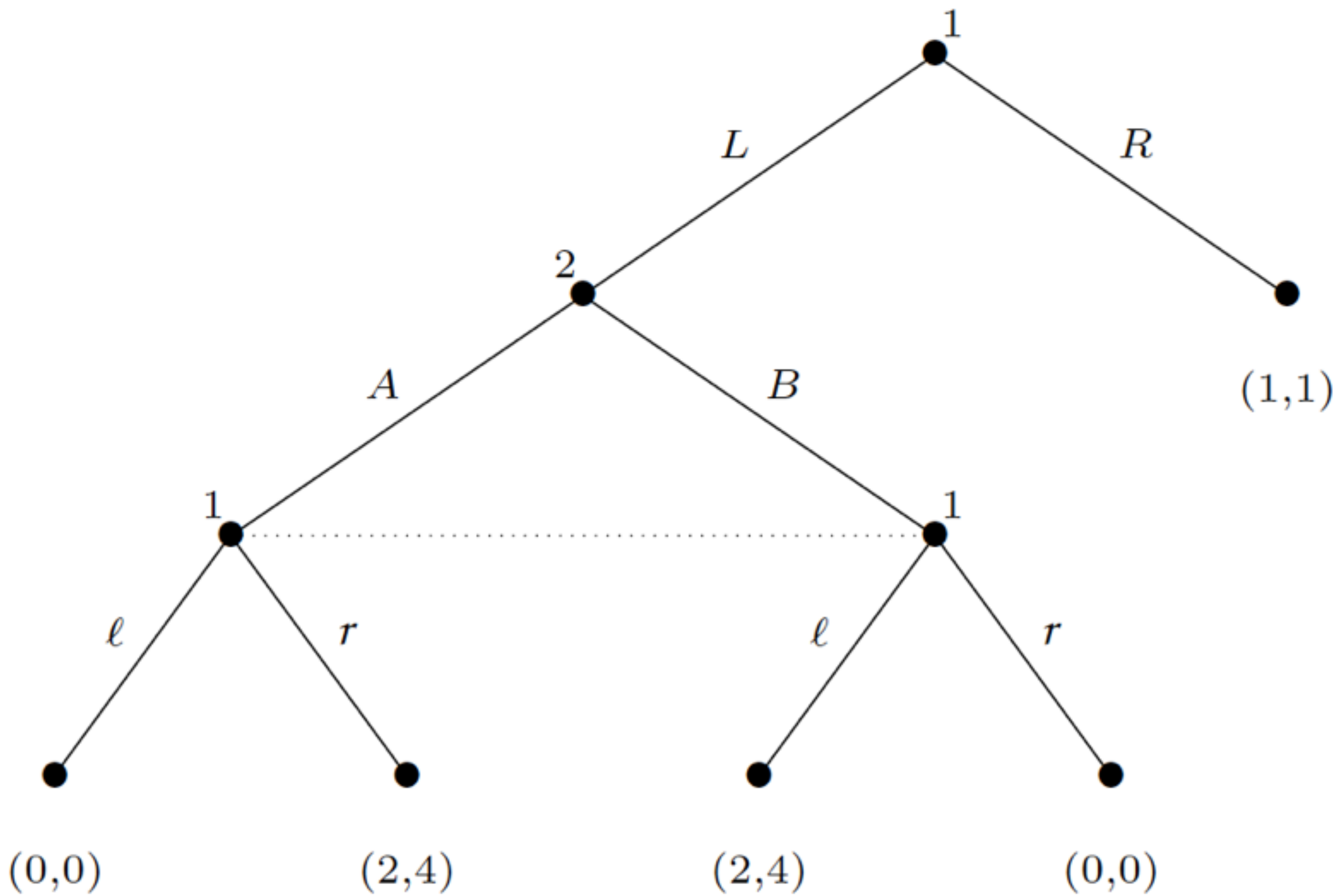
Imperfect Information EF Games

Definition 5.2.1 (Imperfect-information game) *An imperfect-information game (in extensive form) is a tuple* $(N, A, H, Z, \chi, \rho, \sigma, u, I)$, *where:*

- $(N, A, H, Z, \chi, \rho, \sigma, u)$ *is a perfect-information extensive-form game; and*
- $I = (I_1, \dots, I_n)$, *where* $I_i = (I_{i,1}, \dots, I_{i,k_i})$ *is a set of equivalence classes on (i.e., a partition of)* $\{h \in H : \rho(h) = i\}$ *with the property that* $\chi(h) = \chi(h')$ *and* $\rho(h) = \rho(h')$ *whenever there exists a* j *for which* $h \in I_{i,j}$ *and* $h' \in I_{i,j}$.



Imperfect Information EF Games

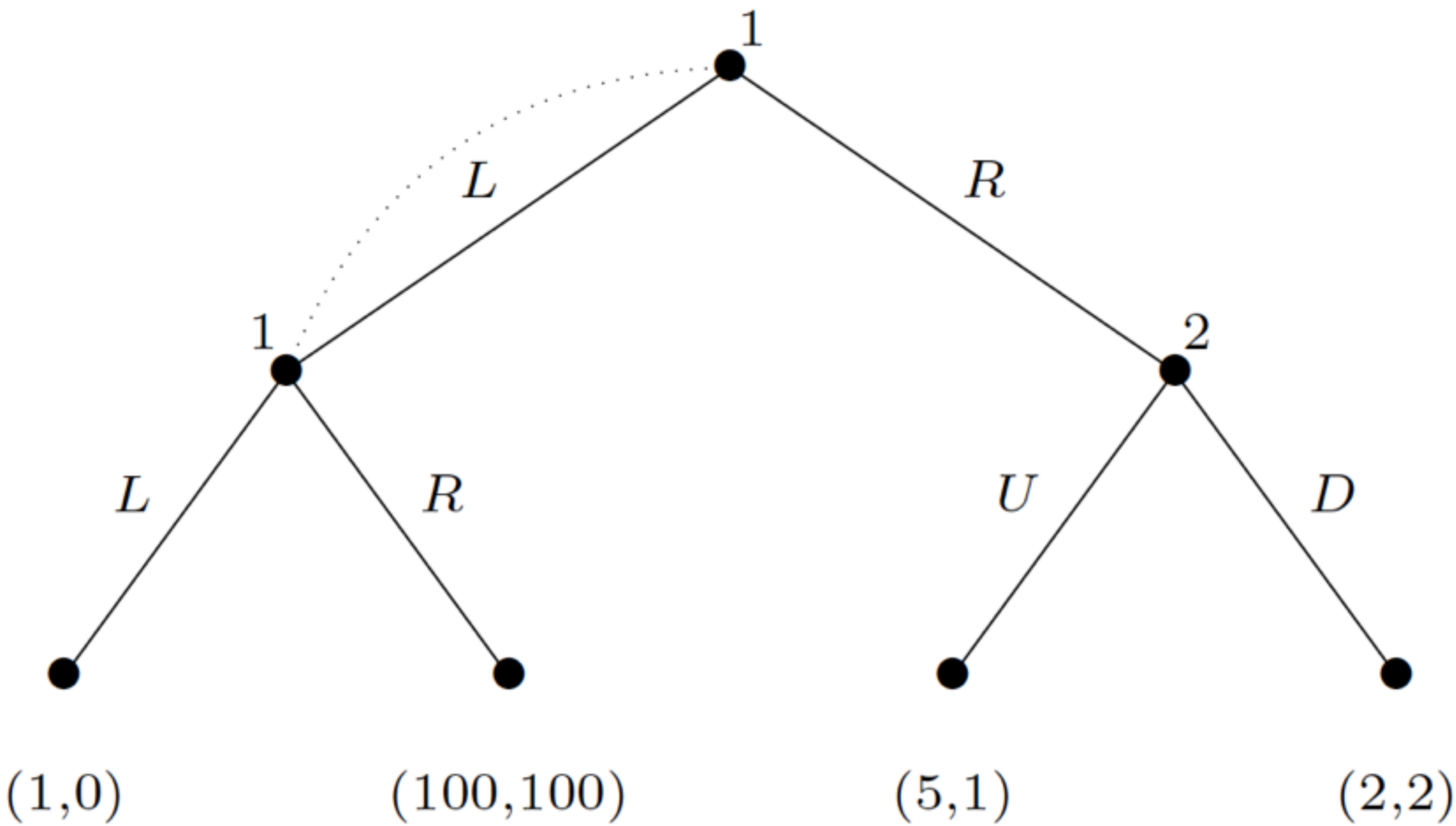


Imperfect Information EF Games

Definition 5.2.1 (Imperfect-information game) *An imperfect-information game (in extensive form) is a tuple* $(N, A, H, Z, \chi, \rho, \sigma, u, I)$, *where:*

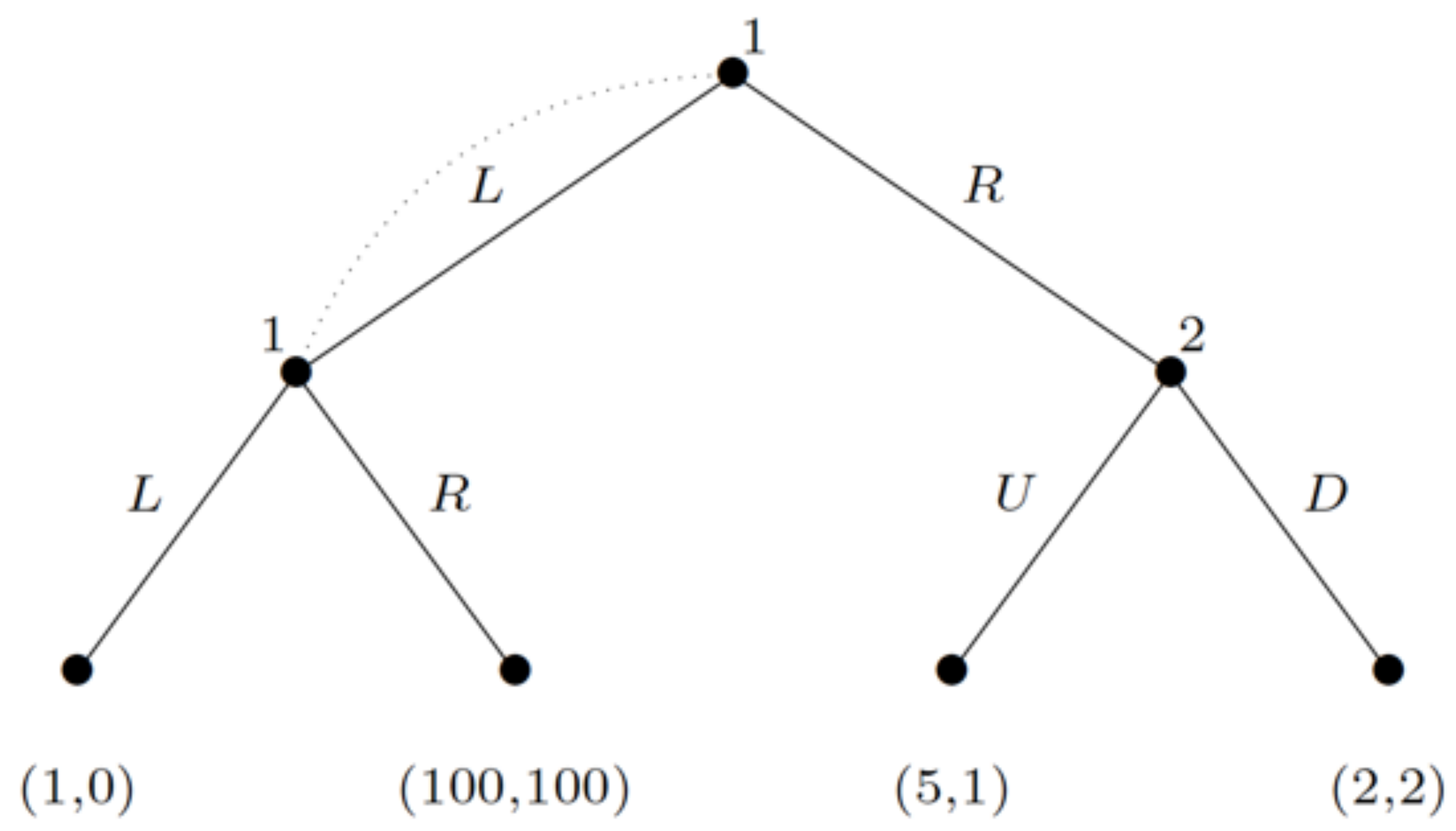
- $(N, A, H, Z, \chi, \rho, \sigma, u)$ *is a perfect-information extensive-form game; and*
- $I = (I_1, \ldots, I_n)$, *where* $I_i = (I_{i,1}, \ldots, I_{i,k_i})$ *is a set of equivalence classes on (i.e., a partition of)* $\{h \in H : \rho(h) = i\}$ *with the property that* $\chi(h) = \chi(h')$ *and* $\rho(h) = \rho(h')$ *whenever there exists a* j *for which* $h \in I_{i,j}$ *and* $h' \in I_{i,j}$.

- **Pure strategy:** Cartesian product over action functions for each equivalence class $\prod_{I_{i,j} \in I_i} \chi(I_{i,j})$.
- **Mixed strategy**: distribution over vector denoting pure strategies (mixed strategy randomly chooses a deterministic path through the game tree)
- **Behavioural strategy:** deterministically selected stochastic paths across the game tree.


1
L
R
1
2
L
R
U
D
(1,0)
(100,100)
(5,1)
(2,2)

Example

- mixed strategy: R is sitrictly dominant for player 1, D is best response to player 2. (R,D) is in Nash. $(R, D) = ((0, 1), (0, 1))$



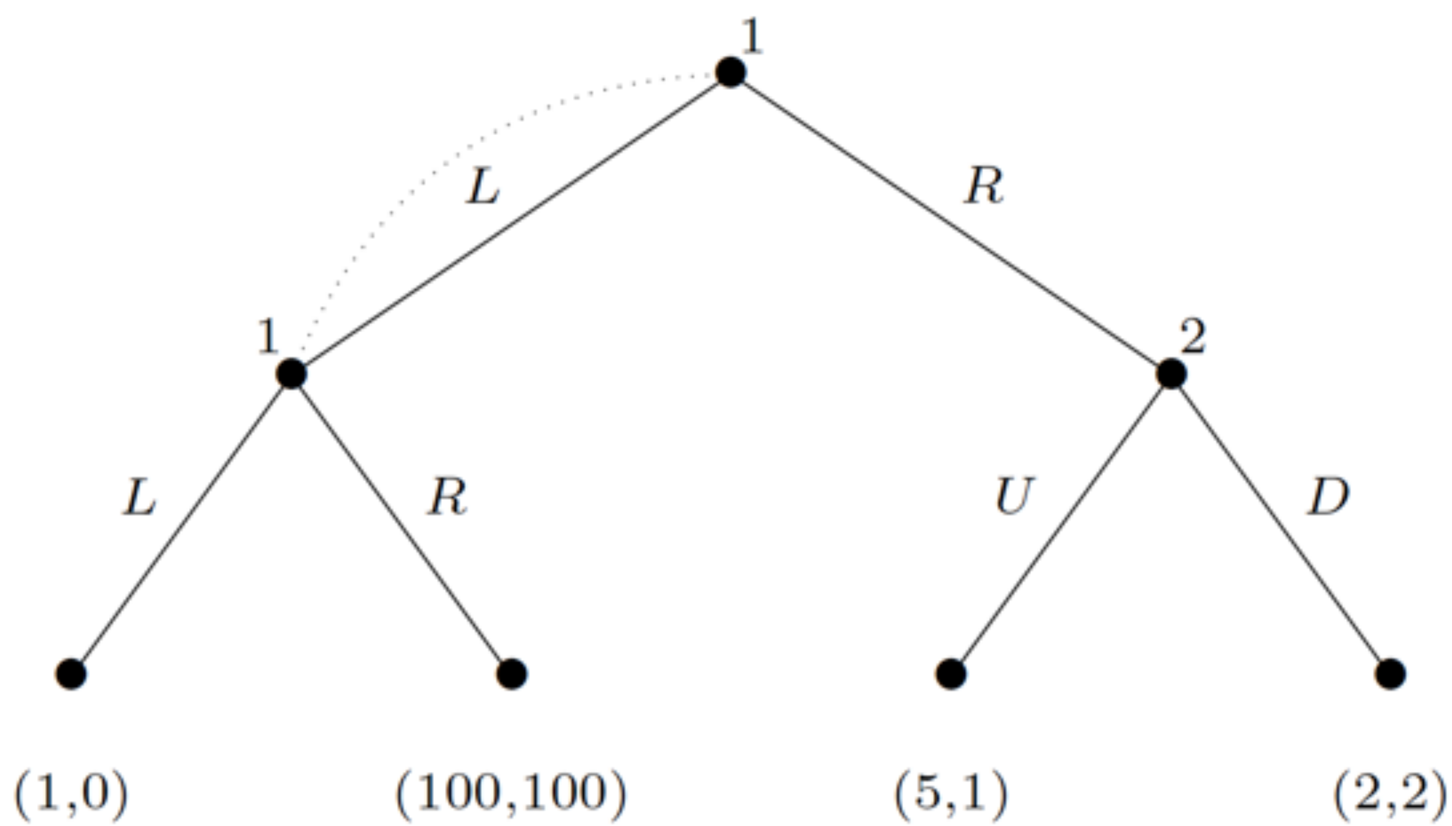
Example

- mixed strategy: $(R, D) = ((0,1),(0,1))$
- behavioural strategy: 1 choosing L with probability p each time in the information set. Utility obtained is than:

$$1 * p^2 + 100 * p(1-p) + 2 * (1-p)$$

maximizing $-99p^2 + 98p + 2$ at $p = \frac{98}{198}$

$$(R, D) = ((98/198, 100/198), (0,1))$$



Perfect recall

There is a broad class of imperfect-information games in which the expressive power of mixed and behavioral strategies coincides. This is the class of games of perfect recall. Intuitively speaking, in these games no player forgets any information he knew about moves made so far; in particular, he remembers precisely all his own moves.

Perfect recall

01

Definition

Player i has perfect recall in an imperfect-information game G if for any two nodes h, h' that are in the same information set for player i, for any path $h_0, a_0, h_1, a_1, h_2, \ldots, h_n, a_n, h$ from the root of the game to h (where the h_j are decision nodes and the a_j are actions) and any path $h_0, a'_0, h'_1, a'_1, h'_2, \ldots, h'_m, a'_m, h'$ from the root to h' it must be the case that:

1. $n = m$
2. For all $0 \leq j \leq n$, h_j and h'_j are in the same equivalence class for player i.
3. For all $0 \leq j \leq n$, if $\rho(h_j) = i$ (that is, h_j is a decision node of player i), then $a_j = a'_j$.

G is a game of perfect recall if every player has perfect recall in it.

Perfect recall

Theorem (Kuhn, 1953)

In a game of perfect recall, any mixed strategy of a given agent can be replaced by an equivalent behavioral strategy, and any behavioral strategy can be replaced by an equivalent mixed strategy. Here two strategies are equivalent in the sense that they induce the same probabilities on outcomes, for any fixed strategy profile (mixed or behavioral) of the remaining agents.

Corollary

In games of perfect recall the set of Nash equilibria does not change if we restrict ourselves to behavioral strategies.

OPPA European Social Fund
Prague & EU: We invest in your future.