Extension of backward induction for the enumeration of pure Nash equilibria outcomes

Paolo Zappalà, Amal Benhamiche, Matthieu Chardy, Francesco De Pellegrini, Rosa Figueiredo

Extension of backward induction for the enumeration of pure Nash equilibria outcomes

Paolo Zappalà[a,b], Amal Benhamiche[a], Matthieu Chardy[a], Francesco De Pellegrini[b], Rosa Figueiredo[b]

[a] *Orange Innovation, Orange, 44 Avenue de la République, Châtillon, 92320, France*
[b] *LIA, Avignon Université, 339 Chem. des Meinajaries, Avignon, 84000, France*

Abstract

Extensive-form games with perfect information admit at least one Nash equilibrium. The backward induction algorithm identifies in linear time a Nash equilibrium of the game, called subgame perfect. We introduce an extension of the backward induction algorithm which is the first to identify all the outcomes of pure Nash equilibria of the game in linear time with respect to the size of the game.

Keywords: Extensive-form games, backward induction, Nash equilibria, enumeration algorithm

1. Introduction

In extensive-form games with perfect information a finite set of agents, called *players*, observe in turns each other's actions and pick one of the possible subsequent actions available to them [1]. Extensive-form games are applied in different fields [2], such as, for instance, economics [3, 4, 5] and law [6]. A *strategy* of a player consists in picking one action whenever she ought to during the game. A *strategy profile* is a combination of strategies, one for each player, leading to an outcome of the game. A *Nash equilibrium* (NE) is a strategy profile such that no player has an incentive to change their own unilaterally. The number of Nash equilibria can be exponential in the size of the game, defined as the number of its outcomes [7]. The aim of this work is to identify all the outcomes that are reachable by actions picked by the players in a NE. The scope of our analysis is extensive-form games with perfect information and their pure Nash equilibria.

Every game admits at least one NE [8]. In particular, extensive-form games always admit a specific NE, called *subgame perfect equilibrium* (SPE) [9]. Such equilibrium can be computed by an algorithm called *backward induction* [10], which, however, does not give insights on other Nash equilibria. The SPE is considered the most appropriate solution for extensive-form games [11], because it is intuitive in its formulation and it can be computed in polynomial time with respect to the size of the game [12]. To our knowledge, in the literature there is no algorithm other than brute force to compute all the Nash equilibria in a generic extensive-form game [7]. Any brute-force algorithm requires exponential time to compute all the Nash equilibria, which makes the SPE also the only viable solution in practice. The adequacy of the SPE as a solution of extensive-form games has been longly debated since its introduction [13]. On one hand, the theoretical assumptions behind the SPE have ben questioned before [14]. On the other hand, empirical results show that even in basic examples other Nash equilibria are attained [15]. To some extent, the lack of convenient methods for computing Nash equilibria in extensive-form games did not allow to advance further in the field.

Contribution. In this work, we introduce a new algorithm for the enumeration of all outcomes of pure Nash equilibria of an extensive-form game, which works recursively like the backward induction. We show that it has the same complexity as the one used to compute SPE outcomes.

The manuscript is organized as follows. In Section 2, we describe some examples of extensive-form games in which the SPE is not accepted as a solution. Then, we formally define the extensive-form game with perfect information, the Nash equilibria and the backward induction algorithm. In Section 3, we introduce the algorithm for the enumeration of the outcomes of the Nash equilibria and we illustrate it with an example. Then, we prove in Section 4 that the algorithm provides the enumeration of the outcomes of the Nash equilibria in linear time with the size of the game. Finally, Section 5 ends the paper with possible directions of future research.

2. Extensive-form games

In this section, we provide two examples of extensive-form games with perfect information and their formal definitions. These examples are used to illustrate how, in some cases, the SPE may not be the most adequate solution of an extensive form game.

Chain store. This example is provided in [16] by the author of the work introducing the SPE. Here, we provide a simplified version of the game, which retains its basic structure. A chain store (*entrant* company) wants to expand in a new town, where a local competitor (*incumbent* company) has a monopoly. If the chain store enters the market, the incumbent can either keep the

Email addresses: paolo.zappala@orange.com (Paolo Zappalà), amal.benhamiche@orange.com (Amal Benhamiche), matthieu.chardy@orange.com (Matthieu Chardy), francesco.de-pellegrini@univ-avignon.fr (Francesco De Pellegrini), rosa.figueiredo@univ-avignon.fr (Rosa Figueiredo)

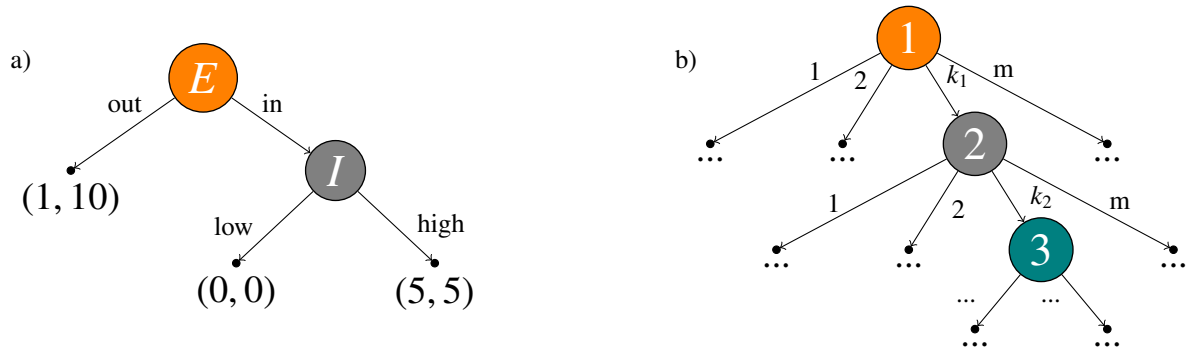

Figure 1: **a) Chain store**. The entrant (E) can either open a store in the town (in) or not (out). The incumbent (I) can either lower the price (low) or keep them high (high). The numerical values represent the profits (in $) made respectively by the entrant and the incumbent. **b) Scheduling problem**. Every player $i \in \{1, \dots, n\}$ observes the decisions of the preceeding players and chooses a machine $k_i \in \{1, \dots, m\}$.

prices high, and thus share the profits with the entrant, or lower the prices, and nullify the profits. Let us consider the tree of Figure 1a. First, the entrant takes the decision on whether to enter the market or to stay out. If she stays out, the incumbent gets all the profits (10$) and the entrant saves the money for the investment (1$). If she enters the market, the incumbent can either share the profits (5$ each) or nullify them (0$ each). The backward induction argument suggests that if the chain store enters the market, the incumbent has no incentive to lower the prices. Therefore the intuitive solution is that the entrant creates a new store in order to share the profits. Actually, this solution corresponds to a *subgame perfect equilibrium*. However, in real scenarios, the incumbent often reduces the prices in order to discourage the chain store from entering the market [17]. In fact, such scenario matches a different, non subgame-perfect Nash equilibrium which results in outcome $(1, 10)$. This example has two equilibria, and they are easy to identify by inspection. The original chain-store game [16], on the other hand, has 20 towns rather than one, so that it is way harder to list all the strategies of the players and identify the equilibria. Our algorithm, however, enables to find all the outcomes of the equilibria in linear time with respect to the size of the game.

Scheduling. We consider the scheduling problem introduced in [4]. There are n ordered agents, each of whom has to perform a different job on one single machine, chosen out of m distinct machines. Agents sequentially choose a machine, being aware of the choice of their own predecessors. As soon as a machine completes all the jobs assigned to it, they are delivered to the agents. The goal of each agent is to have her job delivered at the earliest possible time. Let us consider the tree of the game in Figure 1b). The first agent chooses the machine $k_1 \in \{1, \dots, m\}$. The second agent observes this decision and chooses the machine $k_2 \in \{1, \dots, m\}$. Every agent thus observes the previous decisions and chooses a machine. We do no represent the full tree, since it would have m^n leaves. The criterion used to evaluate this game is the sequential Price of Anarchy (SPoA) [4], which recurs to the SPE, rather than the more common Price of Anarchy (PoA) [18], which relies on the worse Nash equilibrium in order to evaluate the maximum performance loss over the social optimum. The reason for this choice is that the backward induction algorithm allows to compute the SPE in linear time with respect to the size of the game (m^n). On the other hand, to the best of our knowledge, there are no methods in the literature to compute efficiently the other Nash equilibria. Our method to identify all the outcomes of the Nash equilibria allows to extend the analysis in [4] to the PoA, rather than the unusual SPoA.

Formalism. This work deals with recursive algorithms. We next provide a notation framework for extensive-form games which appears natural and convenient for the use of recursive algorithms. More specifically, an extensive-form game with perfect information is defined as the tree of the possible actions that players can take [1]. We observe that every stage of the game is a game itself, called *subgame*. We thus define recursively a game as its first stage and all the subgames that follow from it. In order to achieve the classical representation of games as trees, it is sufficient to define the set of nodes of the tree as a bijection of the set of all subgames. The outcomes of the games correspond to the leaves of the tree.

We consider a game with a finite number N of players $\mathcal{I} = \{1, \dots, N\}$. A finite extensive-form game develops in a finite number of stages. At every stage, the designated player $P \in \mathcal{I} \cup \{\emptyset\}$ is chosen based on the sequence of actions occurring up to the stage. This designated player has available a new set of actions A at this stage. Every action $a \in A$ leads to a different new stage of the game, that we denote by $\Gamma(a)$. At this stage, a new player can observe all previous actions; therefore we can consider the new stage $\Gamma(a)$ as a subgame. We also say that the player acts at subgame $\Gamma(a)$.

When the set of available actions is empty ($A = \emptyset$) the game ends, and we obtain what is called an *outcome*. The corresponding subgame is called a terminal subgame. We call H the set of the outcomes of the game. Every outcome is evaluated by every player $i \in \mathcal{I}$ through a function, called *utility function*, $u_i : H \to \mathbb{R}$. We also write $h_1 \succ_i h_2$ for $h_1, h_2 \in H$ when $u_i(h_1) > u_i(h_2)$, or $h_1 \sim_i h_2$ when $u_i(h_1) = u_i(h_2)$. Next, we provide the recursive definition of an extensive-form game.

Definition 1 (extensive-form game)**.** *An extensive-form game is a tuple* $\Gamma = \langle \mathcal{I}, A, H, P, u \rangle$*, where:*

- $\mathcal{I} = \{1, \dots, N\}$ *is a finite set of players;*
- A *is a finite set of actions;*

- *H is the set of outcomes of the game;*
- *$P \in \mathcal{I}$ is the player acting;*
- *If $A = \emptyset$, then the game ends, $|H| = 1$.*
- *If $A \neq \emptyset$, then every action $a \in A$ leads to a subgame, that we denote by $\Gamma(a)$.*
- *$u = (u_i)_{i \in \mathcal{I}}$, with $u_i : H \to \mathbb{R}$ the utility function of player $i \in \mathcal{I}$.*

Remark. The above recursive definition does not require to introduce the notion of history or to refer to sub-games as trees, as done usually in the literature. Hence, throughout the text, the compact notation $(A_\Gamma, H_\Gamma, P_\Gamma)$ will uniquely determine any object (A, H, P) of a game Γ.

We only consider *pure strategies* [8], from now simply called *strategies*, as they are sufficient to compute all equilibria (cf. [19] and Theorem 1 of [20]). As stated in the next definition, a strategy is a function that shows which action would be taken by a player at every stage.

Definition 2 (strategy)**.** *Given a game $\Gamma = \langle \mathcal{I}, A, H, P, u \rangle$ and a player $i \in \mathcal{I}$, let us consider G_i the subgames at which player i acts. A strategy $s_i \in S_i$ is a function $s_i : \Gamma' \in G_i \mapsto a \in A_{\Gamma'}$ that maps every subgame $\Gamma' \in G_i$ to one of the actions $a \in A_{\Gamma'}$ available to the player.*

We call *strategy profile* a N-tuple of strategies $s = \langle s_1, s_2, \ldots, s_N \rangle$, one for each player. We denote by S_i the set of strategies of player i and by $S = S_1 \times S_2 \times \cdots \times S_N$ the set of all strategy profiles. If every player chooses a strategy, one single action is picked at every node; therefore, given a strategy profile, the actions chosen by the players lead to a single outcome. We denote by $s \mapsto h$ the outcome $h \in H$ of a strategy profile $s \in S$. With some abuse of notation, let $u_i(s) := u_i(s \mapsto h)$ denote the utility of player i under a certain strategy profile s. Furthermore, we write $s_{-i} = \langle s_j \rangle_{j \in \mathcal{I} \setminus \{i\}}$.

A strategy profile is a Nash equilibrium if no player can increase her utility by changing unilaterally her strategy.

Definition 3 (Nash equilibrium)**.** *Given a game $\Gamma = \langle \mathcal{I}, A, H, P, u \rangle$, a strategy profile $\langle s_i \rangle_{i \in \mathcal{I}}$ is a Nash equilibrium if for every $i \in \mathcal{I}$ and for all $s'_i \in S_i$ the condition $u_i(s_i, s_{-i}) \geq u_i(s'_i, s_{-i})$ holds.*

We now introduce the fundamental question that leads the analysis developed in this work.

Problem 1 (Outcome of a Nash equilibrium)**.** *Given a game $\Gamma = \langle \mathcal{I}, A, H, P, u \rangle$ and an outcome $h \in H$, is there a Nash equilibrium $s \in S$ having h as its realisation ($s \mapsto h$)?*

The object of our analysis is an algorithm able to answer the above question for all the outcomes of an extensive-form game. The proposed enumeration algorithm is inspired by the backward induction (BI) algorithm, which is the most known algorithm to compute outcomes of Nash equilibria in extensive-form games [10]. For the sake of the subsequent development, rather than the standard tree-folding procedure customary in the literature, we have reported a recursive, outcome-oriented version of the BI procedure in Algorithm 1.

ALGORITHM 1: Backward induction (BI)

Input: A game $\Gamma = \langle \mathcal{I}, A, H, P, u \rangle$.
Output: The set of outcomes of the subgame perfect equilibria $SPE \subseteq H$.

```
SPE ← ∅ ;                    // Initialising the output
if |H| = 1 then
    SPE = H ;
else
    ⟨Γ^a, SPE^a, h^a⟩_{a∈A} ← ∅;
    for a ∈ A do
        SPE^a = BI(Γ(a)) ;  // Recursive call on the
         subgame Γ(a) following action a
        h^a ∈ arg min_{h'∈SPE^a} u_P(h') ;   // An outcome of
         a SPE achieving the lowest utility in
         Γ(a) for the acting player P
    end
    SPE = {h ∈ ∪_{a∈A} SPE^a | ∀a ∈ A, h ≥_P h^a} ;
     // Outcome h is preferred by player P
end
```

The BI algorithm provides only a specific subset of Nash equilibria, i.e., the *subgame perfect equilibria* (SPE). A strategy profile is a subgame perfect equilibrium if it defines a Nash equilibrium for every subgame [9]. Every game with perfect information admits a subgame perfect equilibrium [9].

The backward induction selects, starting from the leaves of the tree of the game, the outcomes that are most preferred by the player acting at a given node. Algorithm 1 descends along the tree through recursive calls, then it selects at every subgame the outcomes preferred by the acting player. The SPE might not be unique, so the candidate outcome h must be preferred among at least one possible selection of candidate outcomes $\{h^a \in SPE^a\}$ of the other subgames $\Gamma(a)$ for each $a \in A$. Here, SPE^a denotes the set of SPE outcomes of the subgame following a. Showing $u_P(h)$ is larger than at least one $u_P(h')$ with $h' \in SPE^a$ is equivalent to showing that it is larger than the minimum of such values. The candidate outcomes thus propagate upwards, towards the root of the tree, as exemplified next.

Example. Let us consider the game Γ represented by the tree of Fig. 2a). The preferences of the players w.r.t. the outcomes are indicated in the caption. Let us compute the subgame perfect equilibria of the game by applying the BI algorithm. Once the algorithm reaches a terminal subgame, the backward selection of the SPE is executed. Player 1 strictly prefers h_2 to h_1 ($h_2 \succ_1 h_1$), player 2 has no strict preference between h_3 and h_4 ($h_3 \sim_2 h_4$) and player 3 prefers h_8 to h_7 ($h_8 \succ_3 h_7$). The outcomes h_2, $\{h_3, h_4\}$ and h_8 correspond to the SPE of the respective subgames, as shown in Fig. 2b). At the second level of the tree, player 3 prefers h_2 to h_3 and h_4 ($h_2 \succ_3 h_3 \sim_3 h_4$) and player 2 prefers h_5 to h_6 and h_8 ($h_5 \succ_2 h_8 \succ_2 h_6$). Finally, at the root of the tree, player 1 prefers h_2 to h_5 ($h_2 \succ_1 h_5$). The (here unique) outcome of a subgame perfect equilibrium of the game is h_2.

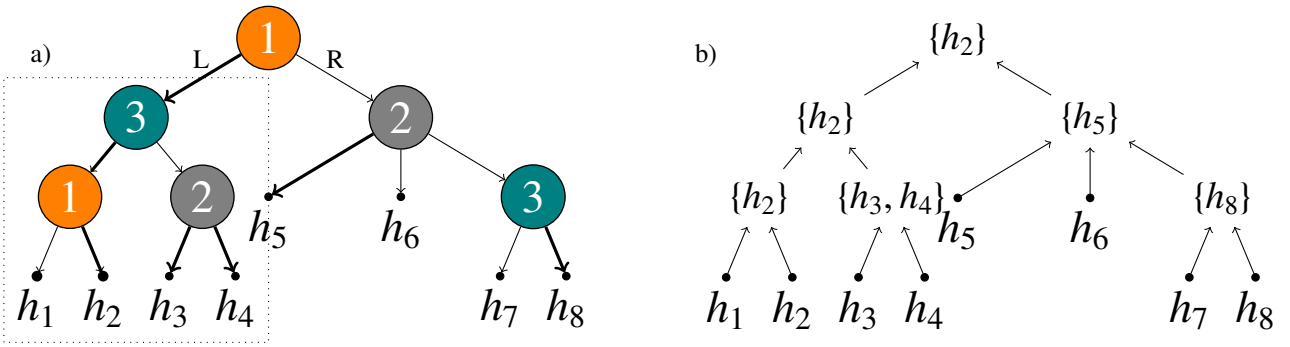

Figure 2: **Example.** a) 3-player game in extensive form. Preferences of the players over the outcomes are respectively: $u_1 : h_6 \succ_1 h_7 \succ_1 h_8 \succ_1 h_3 \succ_1 h_4 \succ_1 h_2 \succ_1 h_1 \succ_1 h_5$, $u_2 : h_5 \succ_2 h_8 \succ_2 h_7 \succ_2 h_6 \succ_2 h_2 \succ_2 h_3 \sim_2 h_4 \succ_2 h_1$ and $u_3 : h_8 \succ_3 h_7 \succ_3 h_6 \succ_3 h_2 \succ_3 h_5 \succ_3 h_3 \succ_3 h_1 \succ_3 h_4$. The subgame $\Gamma(L)$ is highlighted. Player $P_{\Gamma(L)} = 3$ observes action L. b) Application of the backward induction to the game. The (here) unique outcome of a subgame perfect equilibrium of the game is h_2.

3. Enumeration of the outcomes of Nash equilibria

In order for an outcome to be the realisation of a subgame perfect equilibrium, at every stage it must have a utility higher than the lowest valued SPE h^a from every subgame $\Gamma(a)$ for the acting player. Given i the player acting at the stage and A the set of actions to be taken, this condition can be formulated as such:

$$h \succeq_i \underset{h^a \in SPE^a_{min}, a \in A}{\arg\max} \; u_i(h^a),$$

where we define $SPE^a_{min} = \arg\min_{h' \in SPE^a} u_i(h')$ for $a \in A$.

Since the set A is finite, there is an outcome $\theta \in H$ such that $\theta \in \arg\max_{h^a \in SPE^a_{min}, a \in A} u_i(h^a)$. In other words, for every stage, there is an outcome $\theta \in H$ whose value of utility $u_i(\theta)$ represents the threshold that the utility of the player i must exceed in order for h to be an outcome of the SPE. The BI algorithm enables to compute the threshold outcome θ at every stage, and then compares its utility with the one of the candidate equilibrium outcomes.

In this work, we present an algorithm that identifies all black the outcomes of the Nash equilibria. Analogously to BI, the Nash Backward Induction algorithm (cf. Algorithm 2) computes recursively at every subgame a threshold and compares its utility to the one of the candidate outcomes. The main difference between the backward induction (cf. Algorithm 1) and the NE enumeration algorithm (cf. Algorithm 2) is in the computation of the threshold. Indeed, let us observe Algorithm 2: at every stage it stores not a single threshold outcome, but as many threshold outcomes as players, i.e., a vector of threshold outcomes $\langle \theta_i \rangle_{i \in \mathcal{I}}$. At every stage, by applying the principle of backward induction, these threshold outcomes are computed recursively based on the threshold outcomes $\langle \theta^a_i \rangle_{i \in \mathcal{I}}$ of the subgames $\Gamma(a), a \in A$, by maximising the value of the utility for the player P acting at the considered stage, while minimising the utility for the other players:

$$\theta_i \in \underset{\theta^a_i, a \in A}{\arg\max} \, u_i(\theta^a_i), \quad \text{if} \quad i = P,$$

$$\theta_i \in \underset{\theta^a_i, a \in A}{\arg\min} \, u_i(\theta^a_i), \quad \text{if} \quad i \neq P.$$

Finally, the threshold θ_P corresponding to the player $i = P$ acting at the stage is compared with the candidates outcomes $\cup_{a \in A} NE^a$ (NE^a denoting the set of NE outcomes of the subgame $\Gamma(a)$).

As we prove in Section 4, adopting such threshold definition for the outcomes enables the enumeration of the outcomes of the Nash equilibria, corresponding to both SPE and not SPE of the game.

ALGORITHM 2: (NBI) Nash Backward Induction

Input: A game $\Gamma = \langle \mathcal{I}, A, H, P, u \rangle$.
Output: $\langle NE, \langle \theta_i \rangle_{i \in \mathcal{I}} \rangle$: the set of outcomes which are realisations of all Nash equilibria $NE \subseteq H$ and the thresholds $\langle \theta_i \rangle_{i \in \mathcal{I}}$ for every player.

$\langle NE, \langle \theta_i \rangle_{i \in \mathcal{I}} \rangle \leftarrow \emptyset$; `// Initialising the output`
if $|H| = 1$ **then**
  $NE = H$; `// Terminal game`
  $\theta_i = h$ for all $i \in \mathcal{I}$, where $H = \{h\}$; `// Threshold outcomes for all players`
else
  $\langle NE^a, \langle \theta^a_i \rangle_{i \in \mathcal{I}} \rangle_{a \in A} \leftarrow \emptyset$; `// Subgames output`
  for $a \in A$ **do**
    $\langle NE^a, \langle \theta^a_i \rangle_{i \in \mathcal{I}} \rangle = NBI(\Gamma(a))$; `// candidate outcomes and thresholds of` $\Gamma(a)$
  end
  $\theta_P \in \arg\max_{\theta^a_P, a \in A} u_P(\theta^a_P)$; `// Outcome maximising the utility for the acting player`
  $NE = \{h \in \cup_{a \in A} NE^a | h \succeq_P \theta_P\}$; `// Outcomes exceeding the acting player's threshold`
  for $i \in \mathcal{I} \setminus \{P\}$ **do**
    $\theta_i \in \arg\min_{\theta^a_i, a \in A} u_i(\theta^a_i)$; `// Outcome minimising the other players' utility`
  end
end

The application of the algorithm to the game provided in Fig. 2 is given in Fig. 3. The left hand side shows the threshold computed for every subgame, while the right side provides the outcomes of the Nash equilibria of every subgame. Algorithm 2 computes the threshold outcomes and checks which el-

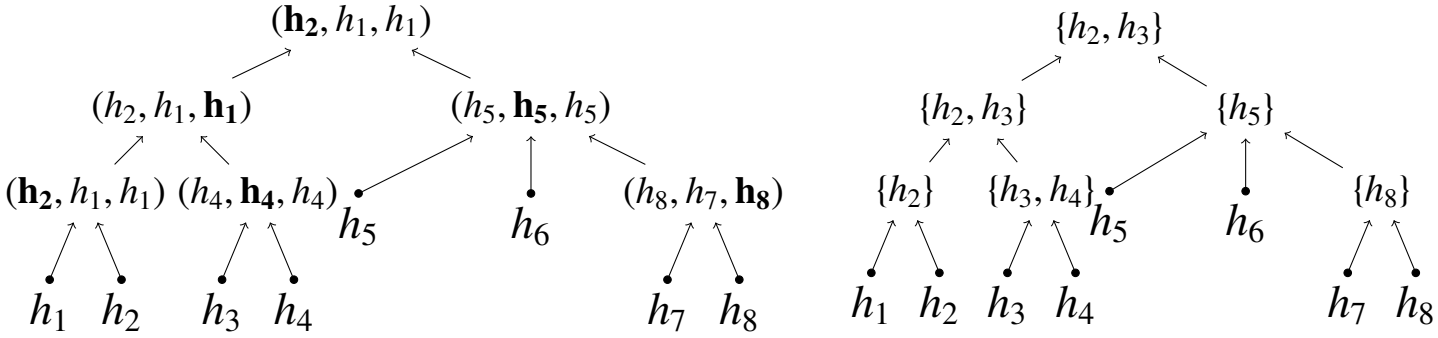

Figure 3: **Application of Algorithm 2** to the game of Fig. 2. Left: Thresholds $\theta = (\theta_1, \theta_2, \theta_3)$ at every node. Right: Nash Equilibria NE of every subgame.

ements meet them at the same time. For sake of clarity, in the following, we first illustrate the computation of the thresholds and then their use to identify the outcomes of Nash equilibria.

Threshold outcomes. We identify now the thresholds for the game of Fig. 2. Let us start from the bottom nodes on the third stage. Player 1 must choose between h_1 and h_2. Following Algorithm 2, we identify the outcome with largest utility for player 1 ($\theta_1 = h_2$, because $h_2 \succ_1 h_1$) and the lowest utility for the other players ($\theta_2 = \theta_3 = h_1$ in both cases, since $h_1 \prec_2 h_1$ and $h_1 \prec_3 h_2$). Player 2 chooses between h_3 and h_4, which are equivalent for her $h_3 \sim_2 h_4$, therefore $\theta_2 = h_3 = h_4$ (since both can be chosen, for clarity in Fig. 3 we write only $\theta_2 = h_4$). For the other two players 1 and 3 it is necessary to minimise the utility: indeed, we have $\theta_1 = \theta_3 = h_4$, because $h_4 \prec_1 h_3$ and $h_4 \prec_3 h_3$. Player 3 has to choose between the h_7 and h_8. Maximising for the utility of player 3 and minimising for the other two players ($h_8 \prec_1 h_7$, $h_7 \prec_2 h_8$ and $h_8 \succ_3 h_7$), we obtain $(\theta_1, \theta_2, \theta_3) = (h_8, h_7, h_8)$. Let us now compute the thresholds for the nodes at the second stage. Player 3 must choose between left and right, whose thresholds are respectively $\theta = (h_2, h_1, h_1)$ and $\theta = (h_4, h_4, h_4)$. We maximise for player 3 and thus have $\theta_3 = h_1$, because $h_1 \succ_3 h_4$, while we minimise for player 1 and 2, getting $\theta_1 = \arg\min_{h \in \{h_2, h_4\}} u_1(h) = h_2$ and $\theta_2 = \arg\min_{h \in \{h_1, h_4\}} u_2(h) = h_1$. At the second stage player 2 has three actions available (left, centre, right). Left and centre actions lead to outcomes of the game h_5 and h_6 and thus we fix for them the thresholds $\theta = (h_5, h_5, h_5)$ and $\theta = (h_6, h_6, h_6)$. The thresholds to be compared are thus $\theta = (h_5, h_5, h_5)$, $\theta = (h_6, h_6, h_6)$ and $\theta = (h_8, h_7, h_8)$. Let us maximise for the utility of player 2, getting $\theta_2 = \arg\max_{h \in \{h_5, h_6, h_7\}} u_2(h) = h_5$, and minimise for the utility of the other players 1 and 3, leading thus to $\theta_1 = \arg\min_{h \in \{h_5, h_6, h_8\}} u_1(h) = h_5$ and $\theta_3 = \arg\min_{h \in \{h_5, h_6, h_8\}} u_3(h) = h_5$. Finally at the root player 1 has to choose between left and right actions, whose thresholds are (h_2, h_1, h_1) and (h_5, h_5, h_5). For player 1 we have $h_2 \succ_1 h_5$ and thus $\theta_1 = h_2$ (utility is maximised), while for players 2 and 3 we have $\theta_2 = \theta_3 = h_1$, since $h_5 \succ_2 h_1$ and $h_5 \succ_3 h_1$.

Outcomes of Nash equilibria. Simultaneously with the computation of the threshold values at each stage, a backward propagation argument determines which outcomes are realisations of a Nash equilibrium. For an outcome to be a realisation of a Nash equilibrium, it should avoid unilateral deviations at every stage from the root to the leaf. The threshold at every stage is the lowest value a unilateral deviation of the acting player can achieve in a subgame which is not explored: if an outcome does not meet a threshold at a certain stage, it means that the player acting at that stage has an incentive to deviate unilaterally. Once the thresholds are computed, we can thus verify which outcomes meet all the thresholds at every stage.

We will analyse each outcome and verify if the Algorithm 2 identifies it as an outcome of a NE. Fig. 3 shows on the right which outcomes meet the threshold at every stage, starting from the leaves. Let us start from h_1. Starting from the bottom to the root the thresholds are $\theta_1 = h_2$ (player 1 on the third stage), $\theta_3 = h_1$ (player 3 on the second stage) and $\theta_1 = h_2$ (player 1 at the root). The thresholds are highlighted on the left of Fig. 3. The outcome h_1 does not meet two thresholds, indeed $h_1 \prec_1 \theta_1 = h_2$, $h_1 \sim_3 \theta_3 = h_1$ and $h_1 \prec_1 \theta_1 = h_2$, and then it cannot be selected by the algorithm as a NE. Therefore it is not the realisation of a Nash equilibrium. On the other hand, outcome h_2 has the same thresholds and it does meet them all: $h_2 \sim_1 \theta_1 = h_2$, $h_2 \succ_3 \theta_3 = h_1$ and $h_2 \sim_1 \theta_1 = h_2$. Therefore it is the realisation of a Nash equilibrium; indeed, we verified before (cf. Fig. 2b) that it is the outcome of the subgame perfect equilibrium. The outcomes h_3 and h_4 confront the following thresholds: $\theta_2 = h_4$, $\theta_3 = h_1$ and $\theta_1 = h_2$, from the bottom to the root of the tree. Outcome h_3 meets all the thresholds: $h_3 \sim_2 \theta_2 = h_4$, $h_3 \succ_3 \theta_3 = h_1$ and $h_3 \succ_1 \theta_1 = h_2$. It is thus the realisation of a Nash equilibrium. Outcome h_4 fails to meet threshold $h_4 \prec_3 \theta_3 = h_1$, and therefore it is not pulled up by the algorithm as a Nash equilibrium. Outcome h_5 and h_6 confront two thresholds: $\theta_2 = h_5$ and $\theta_1 = h_2$. Outcome h_6 fails to meet the first one, $h_6 \prec_2 \theta_2 = h_5$, while outcome h_5 fails to meet the second one, $h_5 \prec_1 \theta_1 = h_2$. Finally, outcomes h_7 and h_8 confront three thresholds: $\theta_3 = h_8$, $\theta_2 = h_5$ and $\theta_1 = h_2$. Both fail to meet $h_7 \prec_2 h_8 \prec_2 \theta_2 = h_5$. Therefore none of them is a realisation of Nash equilibria. The only outcomes of Nash equilibria are $NE = \{h_2, h_3\}$. The right part of Fig. 3 displays each subset of outcomes identified at each stage by Algorithm 2.

Nash equilibria. Fig. 4 shows two different strategy profiles, having the actions chosen by the players marked in bold. Fig. 4a) corresponds to a SPE of the game, having outcome h_2, which has also been identified previously by the BI algorithm.

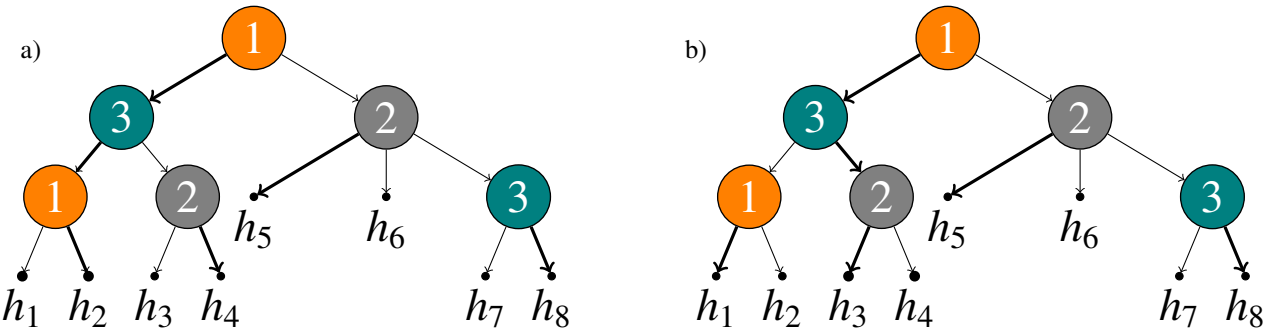

Figure 4: **Nash equilibria.** The strategy profiles of the Nash equilibria having respectively a) h_2 and b) h_3 as outcomes. The players' actions are highlighted by bolder edges.

Fig. 4b) shows another strategy profile, having outcome h_3. This strategy profile is a Nash equilibrium: indeed, every player cannot get better off by deviating unilaterally. If player 1 goes right, she achieves outcome h_5, with $h_3 \succ_1 h_5$. If player 3 goes left, she reaches h_1, with $h_3 \succ_3 h_1$. Finally, if player 2 goes right, she obtains the same utility with h_4: $h_3 \sim_2 h_4$. Such Nash equilibrium cannot be obtained by the BI algorithm. Algorithm 2 not only identifies h_3 as another outcome of a Nash equilibrium, but also proves that no other outcome than h_2 and h_3 holds this property for the game in the example.

Remark. The intuition behind the algorithm is that, in an extensive-form game with perfect information, the unilateral deviation of a player from a pure Nash equilibrium would lead the player to an outcome of the part of the game not explored as part of her strategy. The player does not deviate if and only if the value of the utility of such outcome is lower than the utility of the outcome of the Nash equilibrium. The threshold computes the lowest value a deviation can achieve: as proven next this is a sufficient and necessary condition for preventing an unilateral deviation. The threshold outcomes, in fact, correspond to the strategy profiles where the other players minimise the utility of a deviating player, while she maximizes her own.

4. Proofs and complexity results

Correctness. In the proof that the BI algorithm provides the outcomes of the SPE, the strategy profile defining an SPE is built by following the backward step of the algorithm [9]. The argument of the proof is that, according to BI, all players are dictated to choose at every stage an action that leads to one of the subgame perfect equilibria.

In this section, we prove that Algorithm 2 enumerates the outcomes of the pure Nash equilibria. In order to do that, we first show that if $h \in H$ is an outcome of a Nash equilibrium, it is possible to build a strategy profile s such that h is its realisation ($s \mapsto h$) and s is a Nash equilibrium. Following the scheme of the proof of correcteness of the backward induction, we build the strategies by following the backward phase of Algorithm 2.

Definition 4 (Strategy induced by an outcome)**.** *Let us consider a game $\Gamma = \langle \mathcal{I}, A, H, P, u \rangle$, one of its outcome $h \in H$ and the thresholds $\langle \theta_i \rangle_{i \in \mathcal{I}}$ computed at every node by Algorithm 2. A strategy $s_i^h \in S_i$ for some player $i \in \mathcal{I}$ is said to be induced by the outcome h if at every node where player i acts, she chooses, if available, 1) the action that leads to h, or, if not available, 2) the action that leads to the threshold θ_j, where $j \in \mathcal{I}$ is the player acting at the root of the smallest subgame including both the node and h.*

Example. We consider the game of Fig. 2 and the application of Algorithm 2. Let us analyse the target outcome $h_3 \in H$ returned as a NE by the algorithm.

The induced strategy $s_1^{h_3} \in S_1$ of player 1 is going left (L) at the root, because it leads to h_3, and going left at the other node where player 1 acts, because it leads to h_1, which is the threshold $\theta_3 = h_1$. Indeed, player 3 acts in the subgame $\Gamma(L)$ including both the node where player 1 acts and h_3.

The induced strategy $s_2^{h_3} \in S_2$ of player 2 is going left in the bottom node (to reach h_3) and going left in the other node ($\theta_1 = h_5$). The induced strategy $s_3^{h_3} \in S_3$ of player 3 is going right in the upper node (to reach h_3) and going right in the bottom node ($\theta_1 = h_8$).

The following two results state the correctness of Algorithm 2 for a generic input game $\Gamma = \langle \mathcal{I}, A, H, P, u \rangle$. Let $\langle NE, \langle \theta_i \rangle_{i \in \mathcal{I}} \rangle$ be its output.

Theorem 1. *All the outcomes NE produced by Algorithm 2 are the realisation of a Nash equilibrium.*

Proof. It is enough to prove that, for $h \in NE$, the induced strategy profile $s^h = \langle s_i^h \rangle_{i \in \mathcal{I}}$ is a Nash equilibrium. By construction, h is the realisation of the strategy profile s^h. Let us consider a unilateral deviation $s' \neq s^h$ for some player $j \in \mathcal{I}$. It holds $s'_{-j} = s^h_{-j}$. We prove that $u_j(s^h) \geq u_j(s')$. Let us consider the outcome h' realisation of s'. Since it is a unilateral deviation, it is a node where player j acts that separates the paths from the root to h and h'. By construction of the threshold of such node θ_j, we have that $u(s') = u_j(h') \leq u_j(\theta_j)$. Since $u_j(s^h) \geq u_j(\theta_j)$, we have the proof. □

Algorithm 2 enables to identify a strategy profile s_{-j} of players $\mathcal{I} \setminus \{j\}$ such that player j has no incentive to deviate.

We are now left to prove that if an outcome is not an output for Algorithm 2, i.e., it does not meet a threshold at some node of the game, then it is not the realisation of a Nash equilibrium. The proof is based on the fact that if, for a given player $j \in \mathcal{I}$,

the induced strategy profile of players $\mathcal{I} \setminus \{j\}$, built according to (the thresholds output of) Algorithm 2 does not permit player j to reach outcome h, no Nash strategy profile can do it.

Theorem 2. *If $h \notin NE$, then such outcome is not the realisation of a Nash equilibrium.*

Proof. We prove it by contradiction. Let us assume that h does not meet a threshold θ_j at which player j acts and yet h is the realisation of a Nash equilibrium $s \in S$. By hypothesis, $u(s) = u(h) < u(\theta_j)$. Let us consider the strategy $s_j^{\theta_j} \in S_j$ induced by θ_j. By construction of θ_j, we observe that it is the lowest value player j can achieve with a deviation, i.e., $u(s_j^{\theta_j}, s_{-j}) \geq u(\theta_j)$. Since s is a Nash equilibrium, for any $s'_j \in S_j$ it holds $u(s'_j, s_{-j}) \leq u(s) < u(\theta_j)$. Hence the contradiction $u(s_j^{\theta_j}, s_{-j}) < u(\theta_j)$. □

Complexity. We now show that, in a non-degenerate game, Algorithm 2 has the same complexity as Algorithm 1. A game is degenerate if there are nodes with only one action. In a non-degenerate game, the set of all strategy profiles is in exponential size with respect to the number of outcomes [1]. As discussed in Section 1, a brute-force algorithm to enumerate the Nash equilibria of the game is therefore not practical in real-case scenarios. The backward induction has linear complexity with respect to the number of outcomes, and it is thus preferred to brute force.

Lemma 1 ([12])**.** *Given a non-degenerate game $\Gamma = \langle \mathcal{I}, A, H, P, u \rangle$, the complexity of Algorithm 1 is $O(|H|)$.*

Proof. The result is proven by induction. If $|H| = 1$, the game ends and the backward induction has complexity $O(1)$. If $|H| > 1$, then let us consider the set of actions A. For every action $a \in A$, there is a subgame $\Gamma(a)$ with $|H_{\Gamma(a)}|$ outcomes. By construction, $\sum_{a \in A} |H_{\Gamma(a)}| = |H|$. The complexity of the backward induction is $\sum_{a \in A} O(|H_{\Gamma(a)}|) = O(\sum_{a \in A} |H_{\Gamma(a)}|) = O(|H|)$. □

In the theory of extensive-form games, players are assumed to be finite and in a fixed number $N = O(1)$. We now prove that the complexity of Algorithm 2 is $O(N \cdot |H|)$, which is thus equivalent to $O(|H|)$.

Theorem 3. *Given a non-degenerate game $\Gamma = \langle \mathcal{I}, A, H, P, u \rangle$, the complexity of Algorithm 2 is $O(N \cdot |H|)$.*

Proof. The proof is analogous to that of Lemma 1. If $|H| = 1$, Algorithm 2 has complexity $O(N)$. If $|H| > 1$, the complexity of Algorithm 2 is $\sum_{a \in A} O(|H_{\Gamma(a)}|) + O(N \cdot |H|) = O(N \cdot |H|)$. □

A corollary of Theorem 3 is that the problem of the enumeration of the outcomes of the Nash equilibria is as complex as the problem of identifying the outcomes of the subgame perfect equilibria of the game.

5. Conclusions

In this paper, we introduced a new method to compute all outcomes of the pure Nash equilibria of an extensive-form game. Notably, it operates in linear time with respect to the size of the game. Our starting point is the backward induction algorithm, which dates back to the 19th century and provides only subgame-perfect Nash equilibria. Our extension of the backward induction technique, based on the new notion of a threshold outcome, enables to enumerate all the outcomes of pure Nash equilibria of an extensive form game. Until now, the absence of efficient methods to compute Nash equilibria has limited the analysis of models relying on extensive-form games, which find application in various fields in the literature. We believe that this new method for the enumeration of the solutions of a game can support further theoretical insights in all these domains.

References

[1] H. W. Kuhn, A. W. Tucker, Contributions to the Theory of Games, no. 28 in II, Princeton University Press, 1953.

[2] P. K. Dutta, Strategies and games: theory and practice, MIT press, 1999.

[3] F. M. Fisher, Games economists play: a noncooperative view, The Rand journal of economics 20 (1) (1989) 113–124.

[4] R. Hassin, U. Yovel, Sequential scheduling on identical machines, Operations Research Letters 43 (5) (2015) 530–533.

[5] P. Zappalà, A. Benhamiche, M. Chardy, F. De Pellegrini, R. Figueiredo, A timing game approach for the roll-out of new mobile technologies, in: Proc. of IEEE WiOpt, 2022, pp. 217–224.

[6] D. G. Baird, R. H. Gertner, R. C. Picker, Game theory and the law (1998).

[7] D. Koller, N. Megiddo, The complexity of two-person zero-sum games in extensive form, Games and economic behavior 4 (4) (1992) 528–552.

[8] J. F. Nash Jr, Equilibrium points in n-person games, Proceedings of the national academy of sciences 36 (1) (1950) 48–49.

[9] R. Selten, Spieltheoretische behandlung eines oligopolmodells mit nachfrageträgheit: Teil i: Bestimmung des dynamischen preisgleichgewichts, Zeitschrift für die gesamte Staatswissenschaft/Journal of Institutional and Theoretical Economics 2 (H.) (1965) 301–324.

[10] A. Cayley, Mathematical questions with their solutions, The Educational Times 23 (1875) 18–19.

[11] R. J. Aumann, Backward induction and common knowledge of rationality, Games and economic Behavior 8 (1) (1995) 6–19.

[12] J. Szymanik, Backward induction is ptime-complete, in: Logic, Rationality, and Interaction: 4th International Workshop, LORI 2013, Hangzhou, China, October 9-12, 2013, Proceedings 4, Springer, 2013, pp. 352–356.

[13] K. Binmore, Rationality and backward induction, Journal of Economic Methodology 4 (1) (1997) 23–41.

[14] P. Pettit, R. Sugden, The backward induction paradox, The Journal of Philosophy 86 (4) (1989) 169–182.

[15] K. Binmore, J. McCarthy, G. Ponti, L. Samuelson, A. Shaked, A backward induction experiment, Journal of Economic theory 104 (1) (2002) 48–88.

[16] R. Selten, The chain store paradox, Theory and decision 9 (2) (1978) 127–159.

[17] D. Simon, Incumbent pricing responses to entry, Strategic Management Journal 26 (13) (2005) 1229–1248.

[18] M. Haviv, T. Roughgarden, The price of anarchy in an exponential multi-server, Operations Research Letters 35 (4) (2007) 421–426.

[19] S. Demichelis, K. Ritzberger, J. M. Swinkels, The simple geometry of perfect information games, International Journal of Game Theory 32 (2004) 315–338.

[20] C. Audet, S. Belhaiza, P. Hansen, A new sequence form approach for the enumeration and refinement of all extreme nash equilibria for extensive form games, International Game Theory Review 11 (04) (2009) 437–451.