Review of Economic Studies (2015) **0**, 1–46 doi:10.1093/restud/rdv046

Recursive Lexicographical Search: Finding All Markov Perfect Equilibria of Finite State Directional Dynamic Games

FEDOR ISKHAKOV
CEPAR, University New South Wales
JOHN RUST
Georgetown University
and
BERTEL SCHJERNING
University of Copenhagen

First version received January 2014*; final version accepted August* 2015 (*Eds.*)

We define a class of dynamic Markovian games, *directional dynamic games* (DDG), where directionality is represented by a strategy-independent partial order on the state space. We show that many games are DDGs, yet none of the existing algorithms are guaranteed to find *any* Markov perfect equilibrium (MPE) of these games, much less *all* of them. We propose a fast and robust generalization of backward induction we call *state recursion* that operates on a decomposition of the overall DDG into a finite number of more tractable *stage games,* which can be solved recursively. We provide conditions under which state recursion finds at least one MPE of the overall DDG and introduce a *recursive lexicographic search* (RLS) algorithm that systematically and efficiently uses state recursion to find *all* MPE of the overall game in a finite number of steps. We apply RLS to find all MPE of a dynamic model of Bertrand price competition with cost-reducing investments which we show is a DDG. We provide an *exact non-iterative algorithm* that finds all MPE of every stage game, and prove there can be only 1, 3, or 5 of them. Using the stage games as building blocks, RLS rapidly finds and enumerates all MPE of the overall game. RLS finds a unique MPE for an alternating move version of the leapfrogging game when technology improves with probability 1, but in other cases, and in any simultaneous move version of the game, it finds a huge multiplicity of MPE that explode exponentially as the number of possible cost states increases.

Key words: Dynamic games, Directional dynamic games, Markov perfect equilibrium, Subgame perfect equilibrium, Multiple equilibria, Partial orders, Directed acyclic graphs, d-subgames, Generalized stage games, State recursion, Recursive lexicographic search algorithm, Variable-base arithmetic, Successor function.

JEL Codes: D92, L11, L13

1. INTRODUCTION

Dynamic games have had a major impact on both economic theory and applied work over the last four decades, much of it inspired by the Markov perfect equilibrium (MPE) solution concept of Maskin and Tirole (1988). There has been considerable progress in the development of algorithms for computing MPE, including the pioneering work by Pakes and McGuire (1994) and recent progress on homotopy methods for finding multiple equilibria of both static and dynamic games (Besanko *et al*. 2010; Borkovsky *et al*. 2010). There has been progress on algorithms that can provably find *all Nash equilibria* of certain classes of static games. Examples include algebraic approaches for finding all equilibria in cases where the equilibrium conditions can be reduced to paired linear programmes (used to find all Nash equilibria of bi-matrix games by Audet *et al*. 2001), or certain types of polynomial equations (Datta, 2010; Judd *et al*., 2012a, 2012b). However, it remains an extremely challenging problem to find even a *single* MPE of a dynamic game, much less *all* of them. As Hörner *et al*. (2011) note, "Dynamic games are difficult to solve. In repeated games, finding some equilibrium is easy, as any repetition of a stage game Nash equilibrium will do. This is not the case in stochastic games. The characterization of even the most elementary equilibria for such games, namely (stationary) Markov equilibria, in which continuation strategies depend on the current state only, turns out to be often challenging" (p. 1277).

This article reports progress on a new approach for computing MPE that is applicable to a class of dynamic Markovian games that we call *directional dynamic games* (DDGs). We show that many dynamic games exhibit a type of directionality that is not directly linked to the passage of calendar time (which of course makes every dynamic game inherently directional), but rather pertains to the stochastic evolution of the state of the game. In this article we formalize this concept and introduce an algorithm we call *state recursion* that generalizes the standard concept of backward induction in time.[1] Our algorithm is based on a decomposition of the overall DDG into a finite number of simpler, more tractable *stage games*. Assuming that all or some equilibria of these stage games can be computed, we show how it is possible to recursively construct all or some MPE of the overall game. While finding the equilibria of a stage game is a potentially non-trivial problem, it is a lot easier in practice than finding all MPE of the full game, and can be handled in a number of ways, including the homotopy and algebraic methods mentioned above, which are made feasible by the much smaller size of the problem.

When the overall game has multiple MPE, the state recursion algorithm can be used to find a MPE corresponding to any specified *equilibrium selection rule* (ESR). An ESR picks a particular equilibrium in every stage game and thereby constructs a particular MPE of the overall game. We introduce a *recursive lexicographic search* algorithm (RLS) that systematically and efficiently cycles through all feasible ESR and combines the equilibria of the stage games to construct and enumerate all MPE of the overall game in a finite number of steps.[2] RLS checks the sequence

1. The idea of exploiting the directionality of the state space had been used in specific applications before *e.g.* Cabral and Riordan (1994); Cabral (2011); Judd *et al*. (2012a) and Judd *et al*. (2012b). The latter paper describes an "upwind procedure" to solve their duopoly game that exploits "a special structure that allows for much faster computation" (p. 49) that is identical to state recursion. However, they do not provide a general definition of "directionality" or discuss whether the upwind procedure applies to other games, or can be used to find all MPE, the main focus of this paper. In games with a single agent, the upwind procedure is similar to the upwind Gauss–Seidel method, Judd (1998, p. 418). Abbring *et al*. (2010, 2014) exploit directionality and other special features to provide a simple solution method that appears specific to their setting. However, to our knowledge none of these papers, nor any other work we are aware of, has defined the concept of "directionality" and shown how it can be used to find all MPE of the game.

2. Audet *et al*. (2001) developed a different enumerative algorithm that can provably find all extreme Nash equilibria of static bi-matrix games. However, their algorithm involves an enumeration of the vertices of polytopes

of all possible interdependent choices of equilibria of a set of recursively linked stage games that constitute the "building blocks" for DDGs due to our decomposition result. Thus, RLS systematically builds all MPE of the overall DDG from all possible MPE of its recursively linked stage games.

A DDG is a dynamic Markovian game where some of the state variables evolve in a manner that satisfies an intuitive notion of "directionality". Examples of DDGs include: chess where directionality results from the declining number of pieces on the board; Rubinstein's (1982) alternating offer model of bargaining over a stochastically shrinking pie, and many examples in industrial organization such as patent races (*e.g.* Judd *et al.*, 2012b) where part of the state of the game represents technological progress that improves over time. Another example is the dynamic model of platform competition of Dubé *et al.* (2010) where market shares evolve in a directional fashion. We use RLS to find all Markov perfect equilibria of a model of Bertrand pricing with leapfrogging investments by Iskhakov *et al.* (2015) where directionality arises from endogenous adoption decisions of a stochastically improving production technology.[3]

When the state space is finite we can exploit directionality and partition it into a finite number of equivalence classes we call *stages*. Similar to the "arrow of time" the evolution of the directional component of the state space is unidirectional, although it does not necessarily have to be *sequential* or *deterministic*. We can index the stages of a DDG by τ and order them from 1 to $\mathcal{T}$. The key requirement of directionality is that once the game reaches stage τ there is zero probability of returning to any earlier stage $\tau' < \tau$ under *any* feasible Markov strategy of the game.

The partition of the state space into stages implies a corresponding partition of the overall DDG into a finite number of *stage games*. Our concept of a stage game is different from the traditional notion of a single period static stage game in the literature on repeated games. In our setting, the stage games will generally be dynamic, though on a much reduced state space that makes them much simpler to solve than the overall DDG.

The state recursion algorithm we propose to find MPE of a DDG is a form of backward induction, but one that is performed over the stages of the game τ rather than over time t. We start the backward induction by computing an MPE of the last (absorbing) stage of the DDG, $\mathcal{T}$, which we refer to as the *end game*. State recursion can be viewed as a generalization of the method of backward induction that Kuhn (1953) and Selten (1965) proposed as a method to find *subgame perfect equilibria* of finite extensive form games. However, the backward induction that Kuhn and Selten analysed is performed on the *game tree* of the extensive form representation of the game. State recursion is not performed on the game tree, rather it is backward induction on a different object, the *directed acyclic graph* (DAG) that encodes the strategy-independent partial order on the state space rather than the temporal order of moves in a game tree.

When applicable, state recursion is a much more effective method for finding an MPE than traditional time-based backward induction methods. For example, in an infinite horizon DDG there is no last period for performing backward induction in time, as required for backward induction on the game tree. The usual method for finding an MPE for these problems involves a variant of the method of *successive approximations* to find a fixed point of the system of Bellman equations of the players. However, it is well known that in dynamic games the Bellman equations generally do not satisfy the requisite continuity conditions for contraction mappings. As a result,

on a simplex that constitute the subset of "extreme" Nash equilibrium points, the convex hull of which contains the set of all Nash equilibria of the game. When stage games are bi-matrix games, their algorithm can be used jointly with the RLS algorithm to provably find all MPE of the overall DDG.

3. It is worth noting, however, that popular games that exhibit bi-directional dynamics including entry–exit games in the tradition of Pakes and McGuire, do not belong to the DDG class.

there is no guarantee that successive approximations will converge to a fixed point and hence an MPE of the game.[4]

The problems with the standard time-recursion and homotopy algorithms that are usually used to find an MPE of a dynamic game have been recognized in the literature.[5] For example, even though the model of platform competition studied in Dubé *et al*. (2010) is a DDG that can be solved by state recursion, the authors used successive approximations algorithm to try to find a single MPE of the game. The authors report that "iteration algorithms do not necessarily converge" and that they "often found oscillations", with these problems "particularly pronounced in the presence of multiple equilibria" (p. 247).

Indeed, regarding the "iterative techniques for computing equilibrium" Doraszelski and Pakes (2007) note that "there is no proof that either of these algorithms converge" (pp. 1889–1914). Thus, there are *no known algorithms that are guaranteed to find even a single MPE*—at least for general games that do not have "special structure" discussed in the opening paragraph, a class that now include DDGs.

Why is it important to have algorithms that are not only guaranteed to find a single MPE, but are guaranteed to find *all of them?* The current state of the art in empirical Industrial Organization is to use an iterative technique and hope that it converges. Researchers also typically restrict attention to computing symmetric MPE only. However, Doraszelski and Pakes (2007) note that

> we generally have little reason to believe that there is a unique symmetric and anonymous equilibrium. This raises a number of issues for applied work. ... Examples include the estimation of parameters in the presence of multiple equilibria, the ability of data to identify the equilibria (or the equilibrium selection rule) that is being played, and the issue of how to evaluate the possible outcomes of alternative policies in the presence of multiple equilibria.

They also note that existing methods raise "questions that involve the interaction between computation and multiple equilibria, starting with the fact that any algorithm which computes an equilibrium implicitly selects an equilibrium. Most applications have sufficed with the selected equilibrium, perhaps after performing some rudimentary search for alternatives." As a result, "it is not clear that the equilibrium computed by it is the one we should focus on" (pp. 1916–1917).

When we have the ability to find *all* MPE we can use *economic principles* to select an equilibrium rather than relying on an computational algorithm as an inadvertent equilibrium selection rule in the cases where it happens to converge. The state recursion algorithm is not an iterative algorithm in the sense of Doraszelski and Pakes (2007), and is not subject to the problems they identified. For a dynamic directional game, if at least one equilibrium can be found for each of the component stage games, then state recursion *will* return an MPE of the full dynamic game in a *finite number of steps* $\mathcal{T}$ which equals the total number of stages in the game. State recursion finds a *single* MPE of the overall DDG, but when the game has multiple equilibria the selected MPE depends on which equilibrium is chosen in the end game and all other stages of the game by the state recursion algorithm. When there is a solution method that can find *all* MPE of each of the stage games of the DDG, we can apply the RLS algorithm to repeatedly invoke state recursion in an efficient way to compute *all* MPE of the DDG by systematically cycling through all *feasible* ESR of each of the component stage games of the DDG.

4. Note that the contraction property does hold in single-agent games that we can view as Markovian games against nature. This implies that traditional time-based backward induction reasoning will compute an approximate MPE for these problems, where the MPE is simply an optimal strategy for the single agent, a "best reply to nature".

5. We further discuss homotopy methods in the concluding section: in particular, we discuss how RLS can complement homotopy methods to combine the strengths of both approaches.

The idea of using multiple equilibria of stage games to construct a much larger set of equilibria of an overall game was used by Benoit and Krishna (1985) to show that a version of the "Folk Theorem" can hold in finitely repeated games. However, they did not propose any algorithm or constructive approach for enumerating all possible subgame perfect equilibria of a finitely repeated game. The RLS algorithm can be used to find and enumerate all such equilibria.

We illustrate the use of state recursion and RLS by finding all MPE of a dynamic model of Bertrand price competition with leapfrogging investments that was studied (using analytic methods) by Iskhakov *et al*. (2015). We show this game is a DDG and show how all MPE of each of its component stage games can be computed *exactly* by finding all roots of a pair of quadratic equations. This enables us to prove that every stage game of the simultaneous move version of the leapfrogging game (with no unobserved cost shocks to investment) must have either 1, 3, or 5 MPE, and at least one of these must be a pure strategy equilibrium.

We believe the full solution to this model is of independent interest, since even though the Bertrand equilibrium is a cornerstone concept in economics and extending this model to more realistic dynamic contexts is of significant interest, the properties of equilibria arising in even the simplest dynamic extension with stochastic production costs have never been well understood (Routledge, 2010, p. 357). The RLS algorithm enabled us to "crack" this problem and provide an analytical characterization of the set of *all* MPE in this game. In particular, we show that leapfrogging emerges as a generic feature of competition in this setting, contrary to Rust (1987) who considered a similar setup and conjectured that generally the only MPE in the game involves *investment preemption* where one of the two firms makes all investments.

We used RLS to solve two variants of the Bertrand investment game: (1) a simultaneous move version where firms make pricing and investment decisions simultaneously in each time period; and (2) an alternating move version where we introduce an additional state variable that specifies which firm has the right of acquiring state-of-the-art technology in each time period. We assume that this "right to move" state variable evolves as an independent Markov chain, allowing for a deterministically alternating move version of the game as a special case.

The directionality of the game results solely from the fact that the state-of-the-art marginal cost of production only decreases over time (stochastically or deterministically), and the existing marginal costs of the two firms never increase, but only decrease when either firm acquires the state-of-the-art technology. The right of move state variable in the alternating move version of the game is not directional. Yet, we show that the more complicated stage games in the alternating move version of the leapfrogging game still admit a solution method that finds all MPE of every stage game (though in this case we prove there can only be 1 or 3 MPE in every stage game), and thus the RLS algorithm can be applied to find all MPE in both the alternating move and simultaneous move formulations of this game.

Similar to the problems reported by Dubé *et al*. (2010), we find it is typically impossible to pinpoint even a single symmetric MPE using the traditional successive approximation approaches that Doraszelski and Pakes (2007) discussed. However, using RLS, we find that even with a relatively coarse discretization of the space of marginal costs, there are hundreds of millions of MPE and the vast majority of them are *asymmetric*. The alternating move game generally has fewer equilibria, and certain "extremal" MPE such as a zero-profit mixed strategy equilibrium or two asymmetric equilibria supporting monopoly payoffs no longer exist. The RLS algorithm reveals that if the state-of-the-art technology improves in every period with certainty, the model with alternating moves has a *unique* pure strategy MPE.

The rest of the article is organized as follows. In Section 2, we define a notion of directionality and formally define the class of DDGs, introduce the concept of stages and the new notion of *stage games*, define the state recursion algorithm, and prove that it finds an MPE of the overall DDG in a finite number of steps. In Section 3, we introduce the RLS algorithm and provide sufficient

conditions under which this algorithm finds all MPE of the DDG. In Section 4, we illustrate the state recursion and RLS algorithms by using them to find all MPE of the two variants of the duopoly Bertrand investment and pricing game described above. Section 5 concludes by summarizing our results, and discussing some extensions and limitations of the RLS algorithm.

2. FINITE STATE DIRECTIONAL DYNAMIC GAMES AND STATE RECURSION

In this section, we use the notion of *directionality* to define a class of Markovian games that we call *dynamic directional games* or DDGs. We show how directionality simplifies the problem of finding equilibria of these games using *state recursion,* a type of backward induction that differs from the standard time-based backward induction typically used to solve dynamic games. State recursion computes an MPE of the game in a finite number of steps, but it requires the user to specify an ESR that selects one equilibrium out a set of multiple equilibria at a sequence of recursively defined *stage games* of the overall directional game.

2.1. *Finite state markovian games*

Following Kuhn (1953) and Shapley (1953), consider a dynamic stochastic game $\mathcal{G}$ with n players indexed by $i \in \{1,\ldots,n\}$ and T periods indexed by $t \in \{1,\ldots,T\}$, where unless otherwise stated we assume $T=\infty$. We assume the players' payoffs in any period of the game are given by von-Neumann Morgenstern utility functions $u_i(s_t,a_t)$, where player i's payoff in any period t of the game depends both on the state of the game at time t, s_t, and the vector of actions of all players given by $a_t=(a_{1,t},\ldots,a_{n,t})$, where $a_{i,t}$ is the action chosen by player i at time t. Assume that the players maximize expected discounted utility and discount future payoffs using player-specific discount factors $(\beta_1,\ldots,\beta_n)$ where $\beta_i \in (0,1)$, $i=1,\ldots,n$.

Let $p(s'|s,a)$ denote a Markov transition probability for the next period state s' given the current period state s and vector of actions a taken by the players. If we view s as the move by "Nature", the Markovian law of motion for Nature's moves makes it natural to focus on the MPE concept of Maskin and Tirole (1988). These are subgame perfect Nash equilibria of the game $\mathcal{G}$ that are *Markovian, i.e.* they are functions only of the current state s_t and not the entire past history of the game.[6]

In this article, we follow Haller and Lagunoff (2000) and focus on games $\mathcal{G}$ that have a finite state space. They provide general conditions under which the set of MPE of such games are *generically finite.* Let S denote the set of all states the game may visit at any time period and assume that S is a finite subset of R^k for some $k \geq 1$. In every period, each player i chooses an action a_i from a set of feasible actions $A_i(s)$ for player i when the state of the game is s.[7] Assume that for each $s \in S$ and for each i we have $A_i(s) \subseteq A$ where A is a compact subset of R^m for some $m \geq 1$.

Assume that the current state $s \in S$ is known to all the players, and that their past actions are observable (though current actions are not observed in simultaneous move versions of $\mathcal{G}$). We

6. Though we do not take the space to provide a full extensive form description of the game $\mathcal{G}$ we assume that the players have *perfect recall*, and thus can condition on the entire history of states and actions at each time t to determine their choice of action. However, it is not difficult to show that if both Nature and all of the opponents are using Markovian strategies, player i can find a best reply to these strategies within the subclass of Markovian strategies. As a result, one can define a Markov Perfect equilibrium using Bellman equations without having to provide a complete extensive form description of $\mathcal{G}$. See Ritzberger (1999, 2002) and Maskin and Tirole (1988) for further discussion.

7. This formulation includes both simultaneous and alternating move games: in the latter case $A_i(s)$ is a singleton for all players but the one who has the right to move, where one of the components of the state s denotes which of the n players has the right to move.

can also allow for players to condition their decisions on non-persistent private information in the form of idiosyncratic shocks ε that can depend on the current state but not on past shocks given the current value of the serially persistent component of the state, s. However, to simplify notation we do not cover this case here. We assume that all objects in the game $\mathcal{G}$, the players' utility functions, discount factors, the constraint sets $A_i(s)$, and the law of motion $p(s'|s,a)$, are common knowledge.

Let σ denote a feasible set of Markovian *behaviour* strategies of the players in game $\mathcal{G}$, *i.e.* an n-tuple of mappings $\sigma=(\sigma_1,\ldots,\sigma_n)$ where $\sigma_i: S\to\mathcal{P}(A)$ and $\mathcal{P}(A)$ is the set of all probability distributions on the set A. Feasibility requires that $\text{supp}(\sigma_i(s))\subseteq A_i(s)$ for each $s\in S$, where $\text{supp}(\sigma_i(s))$ denotes the support of the probability distribution $\sigma_i(s)$. A pure strategy is a special case where $\sigma_i(s)$ places a unit mass on a single action $a\in A_i(s)$. Let $\Sigma(\mathcal{G})$ denote the set of all feasible Markovian strategies of the game $\mathcal{G}$.

If σ is a feasible strategy n-tuple, let σ_{-i} denote an $(n-1)$-tuple of feasible strategies for all players except player i, $\sigma_{-i}=(\sigma_1,\ldots,\sigma_{i-1},\sigma_{i+1},\ldots,\sigma_n)$, and let $(a,\sigma_{-i}(s))$ denote a strategy where player i takes action $a\in A_i(s)$ with probability one in state s, whereas the remaining players $j\neq i$ chose their actions taking independent draws from the distributions $\sigma_j(s)$.

Definition 1. *An MPE of the stochastic game $\mathcal{G}$ is a feasible strategy n-tuple σ^* and an n-tuple of value functions $V(s)=(V_1(s),\ldots,V_n(s))$ where $V_i: S\to R$, such that the following conditions are satisfied:*

(1) *the system of Bellman equations*

$$V_i(s)=\max_{a\in A_i(s)}\left[E\left\{u_i(s,(a,\sigma^*_{-i}(s))\right\}+\beta_i E\left\{\sum_{s'\in S}V_i(s')p(s'|s,(a,\sigma^*_{-i}(s)))\right\}\right], \quad (1)$$

is satisfied for every $i=1,\ldots,n$, with the expectation in equation (1) taken over the IID probability distributions given by the opponents' strategies σ^*_j, $j\neq i$, and

(2) *for $i=1,\ldots,n$, if the maximizer in the Bellman equation*

$$a^*_i(s)=argmax_{a\in A_i(s)}\left[E\left\{u_i(s,(a,\sigma^*_{-i}(s))\right\}+\beta_i E\left\{\sum_{s'\in S}V_i(s')p(s'|s,(a,\sigma^*_{-i}(s)))\right\}\right], \quad (2)$$

is a single point, σ^*_i is a probability distribution that places probability 1 on $a^*_i(s)$, and if $a^*_i(s)$ has more than one point, $\sigma^*_i(s)$ is a probability distribution with support on a subset of $a^*_i(s)$.

In Definition 1 the notion of "subgame perfectness" is reflected by the restriction implicit in equation (2) and the "Principle of optimality" of dynamic programming which require for each player's strategy σ^*_i, "that whatever the initial state and initial decision are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision" (Bellman, 1957). Thus, equation (2) implies that each player's strategy must be a best reply to their opponents' strategies at *every* point in the state space $s\in S$, but since the process is Markovian, it follows that the strategy σ^* constitutes a Nash equilibrium for all possible histories of the game $\mathcal{G}$ (see Maskin and Tirole, 2001, p. 196).

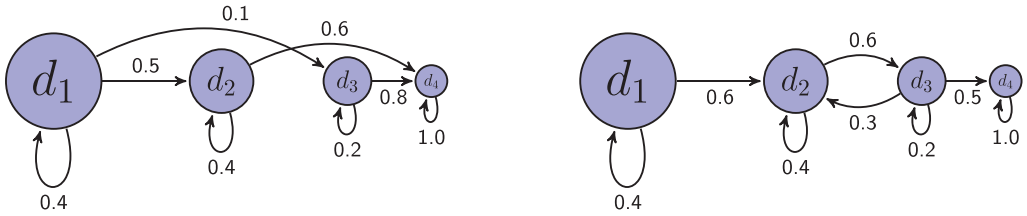

FIGURE 1
Bargaining over a stochastically shrinking pie (Example 2: left panel, Example 3: right panel)

2.2. *Directional dynamic games*

Before we formally define dynamic directional games, it is useful to provide intuitive examples of what we mean by *direction* in a Markovian game. Roughly speaking, a game $\mathcal{G}$ is directional if we can single out some dimensions of the state space S which we call "directional" components, such that the transitions between the points in these dimensions can be represented by a DAG.[8] Each vertex represents a point in the "directional" component of the state space, and the arrows (directed edges) connecting the vertices correspond to positive probabilities of transiting from one point to the other.

Figure 1 presents two directed graphs representing transitions in the state space of two examples of dynamic Markov bargaining games we discuss below. In these games, the state space is one dimensional, and is given by $S=\{d_1,d_2,d_3,d_4\}$ with $d_1>d_2>d_3>d_4$. We can interpret d_i as the size of a "pie" the players are bargaining over. The game presented in the left panel starts at d_1 and the size of the pie evolves stochastically according to the indicated transition probabilities. To qualify as a directional game, it is essential that whatever the current state d_i is, there is zero probability of returning to any state d_j such that $d_j>d_i$, *i.e.* the pie can only shrink or remain the same size, but never increase. This intuitive notion of directionality is violated in the right panel, where the state can oscillate between d_2 and d_3. Consequently, the directed graph representing the transitions among the states of this game is not acyclical, *i.e.* not a DAG.

Directionality in the stochastic evolution of the states in a game $\mathcal{G}$ can be captured by defining a *partial order* over the state space S. This partial order of the states will generally be *strategy specific* since the stochastic evolution of the states will generally depend on the strategies σ used by the players, and we use the symbol $\succ_\sigma$ to indicate this dependence. Most games that we analyse will exhibit directionality only in a subvector of the full vector of state variables. Therefore, our definition assumes there is a decomposition of S as a Cartesian product of two sets D and X, so a generic element of the state space is written as $s=(d,x)$ where we refer to d as the *directional component* of the state space, and x as the *non-directional* component. The partial order $\succ_\sigma$ is defined over the directional component D of the state space S.

In the definition below, we let $\rho(d'|d,x,\sigma)$ denote the *conditional hitting probability, i.e.* the conditional probability that a state with directional component d' is *eventually* reached given that the process starts in state $s=(d,x)$ and the players use strategy σ.[9]

8. Note that while the extensive form representation of a game, the game tree, is also an example of a DAG, it is different from the DAG over the state space. In particular, the game tree cannot have "self-loops" (transitions from a node back to itself) as in Figure 1, and its edges represent actions for each player rather than possible transitions between the points in the state space.

9. Note that $\rho(d'|d,x,\sigma)$ is different from a single-step transition probability. In the terminology of Markov chains, $\rho(d'|d,x,\sigma)$ is the probability that the *hitting time* of the set $(d'\times X)=\{(d',x')|x'\in X\}$ is finite conditional on starting in state (d,x) under strategy σ. The hitting time (or *first passage time*) is the smallest time it takes for the state to travel from state $s=(d,x)$ to some state (d',x') where $x'\in X$.

Definition 2. (Strategy-specific partial order over states). *Let σ be a feasible n-tuple of strategies for the players in the dynamic game $\mathcal{G}$. Suppose S is a finite subset of R^k that can be decomposed as a Cartesian product $S = D \times X$ where $D \subset R^N$, $X \subset R^{k-N}$ and $N \leq k$. A typical element of S is a point $s = (d,x) \in D \times X$, where we allow for the possibility that D or X is a single point (to capture the cases where S has no directional component and the case where S has no non-directional component, respectively). Then a binary relation $\succ_\sigma$ over the directional components $d \in D$ induced by the strategy profile σ is defined as*

$$d' \succ_\sigma d \text{ iff } \exists x \in X \text{ such that } \rho(d'|d,x,\sigma) > 0 \text{ and } \forall x' \in X \, \rho(d|d',x',\sigma) = 0. \tag{3}$$

Lemma 1. (Partial order over directional component of the state space). *The binary relation $\succ_\sigma$ is a partial order of the set D.*

Proof. The proofs to this and subsequent results are provided in Appendix A. ‖

The partial order of the states captures the directionality in the game implied by the strategy σ. The statement $d' \succ_\sigma d$ can be interpreted intuitively as saying that the directional component d' comes *after* the directional state d in the sense that there is a positive probability of going from d to d' but zero probability of returning to d from d'. Note that $\succ_\sigma$ will generally not be a *total order* of the directional components D because there may be pairs $(d',d) \in D \times D$ that are *non-comparable* with respect to the partial order $\succ_\sigma$. There are two ways in which a pair of points (d',d) can be non-comparable (a situation that we denote by $d' \not\succ d$): there may be no communication between d and d', *i.e.* zero probability of hitting state d' from d and vice versa, or there may be a two way transition (a *loop*) connecting d and d', *i.e.* d' can be reached with positive probability from d and vice versa.

The asymmetry and transitivity conditions guarantee that there cannot be any loops between any of the comparable pairs (d',d) of a strict partial order $\succ_\sigma$. However, loops that may exist between *non-comparable* pairs (d',d) that are not elements of the binary relation $\succ_\sigma$, also need to be ruled out.

Definition 3. (No-loop condition). *Let σ be a feasible n-tuple of strategies for the players in the dynamic game $\mathcal{G}$. We say that σ has no loops in the directional component D if the following condition is satisfied for all $d' \neq d \in D$*

$$d' \not\succ_\sigma d \Longrightarrow \forall x \in X \;\; \rho(d'|d,x,\sigma) = 0 \text{ and } \forall x' \in X \, \rho(d|d',x',\sigma) = 0. \tag{4}$$

It is not hard to show that when the No-Loop Condition is satisfied for a feasible strategy σ, the transitions among the directional components of the state vector d induced by this strategy can be represented by a DAG. Let $D(\mathcal{G},\sigma)$ denote a directed graph with nodes corresponding to elements of D and edges connecting the points d and d' if the hitting probability $\rho(d'|d,x,\sigma)$ is positive. Then if d and d' are comparable with respect to $\succ_\sigma$, there can only be an edge from d to d' or vise versa, and otherwise if d and d' are not comparable there is no edge between them due to no communication by the No-Loop Condition. Therefore, the directed graph $D(\mathcal{G},\sigma)$ does not have loops, thus it is a DAG.

Example 1. (Finite horizon). Consider a *finite horizon* Markovian game $\mathcal{G}$ which lasts for $T < \infty$ periods. We can recast this in the notation of a stationary Markovian game by writing the state space as $S = D \times X$ where $D = \{1,2,\ldots,T\}$ is the directional component of the state space and X are the other potentially non-directional components of the state space. The time index t is the directional component of the state space, *i.e.* $d = t$ and we define the partial order $\succ_\sigma$ by $d' \succ_\sigma d$

if and only if $d' > d$. Note that $\succ_\sigma$ in this example is a *total order* of D, and thus there are no pair of non-comparable states (implying that the No-Loop condition is also satisfied). Note as well that the ordering $\succ_\sigma$ holds for every strategy, and is thus independent of σ.

In this simple case, no additional steps are needed to perform the state recursion algorithm that we define below, which reduces here to ordinary backward induction in time. In the general case, we require a strategy-independent total ordering of a partition of the state space to do state recursion. This ordering has to be specifically constructed: we explain how to do this below.

Example 2. (Bargaining over a stochastically shrinking pie). Consider an extension of the Rubinstein (1982) infinite horizon alternating offer bargaining game $\mathcal{G}$ where two players make alternating offers and the size of the amount the players are bargaining over (the "size of the pie)", is given by d which can take four possible values $d \in \{d_1, d_2, d_3, d_4\}$ with $0 < d_4 < d_3 < d_2 < d_1$ as discussed above. Suppose that d evolves as a Markov chain with an exogenous (strategy independent) transition probability $p(d_j|d_i)$, $i,j \in \{1,2,3,4\}$ with values such as in the left panel of Figure 1. Thus, if the pie starts out at its largest size d_1, it has a positive probability that it will remain this size for a geometrically distributed period of time, and there is a positive probability that it will either decrease to size d_3 or d_2 but zero probability that it will shrink directly from size d_1 to its smallest size d_4. It is evident from the left panel of Figure 1 that the transition diagram for the pie is a DAG. The transitions hold for all feasible σ and thus imply a strategy-independent partial order $\succ_\sigma$ $(\forall \sigma)$ over the d variable which consists of the ordered pairs $\{(d_4,d_3),(d_4,d_2),(d_4,d_1),(d_3,d_1),(d_2,d_1)\}$. Notice that $d_2 \not\succ_\sigma d_3$ and $d_3 \not\succ_\sigma d_2$, *i.e.* the ordered pairs (d_3,d_2) and (d_2,d_3) are non-comparable under the partial order $\succ_\sigma$ since there is zero probability of going from d_2 to d_3 and vice versa.

Let $m \in \{1,2\}$ denote which of the players has the turn to make an offer, so player m proposes a division of the pie, which has size d, and the other player then either accepts or rejects the proposed division. If the proposed division is accepted, the game ends and the players consume their respective shares of the pie, otherwise the game continues to the next stage. The m variable may alternate deterministically or stochastically. In terms of our setup, the game involves a two-dimensional state space $s = (d,m)$ where the directional variable is the size of the pie d and the non-directional variable m is the index of the player who can propose a division of the pie. A version of this game was solved by Berninghaus *et al.* (2014) using a backward induction calculation in the d variable. It is an example of the state recursion algorithm we define below.

Example 3. (Bargaining over a pie that can expand or shrink in size). Consider a game similar to the one in Example 2, but slightly modify the transition probabilities for the directional state variable d as shown in the right panel of Figure 1. It is easy to verify that this transition probability induces the same partial order $\succ_\sigma$ over D as the transition probabilities in Example 2. However, in this case there is a loop connecting the non-comparable points d_2 and d_3. This cycle implies that the directed graph in the right panel of Figure 1 is not a DAG. This game will also fail to be a directional dynamic game by the definition we provide below, because the existence of the loop between d_2 and d_3 makes it impossible to devise a total order to index the induction steps in the state recursion algorithm.[10]

10. However, because the state space is finite, it is possible to reorganize the game so that the loop between d_2 and d_3 is "hidden away" in a separate dimension of the state space. With such manipulation, it would be possible to run state recursion using the directionality over the three states $(d_1, \{d_2, d_3\}, d_4)$ but as it will be evident below, the points d_2 and d_3 would not be treated independently in any of the solution algorithms.

Different strategies σ can potentially induce different partial orders of the directional component of the state space D. To be able to construct a common total order for the state recursion algorithm, it is important to ensure that strategy-specific partial orders are *consistent* with each other, *i.e.* that there is no pair of states for which d' follows from state d under strategy σ but d follows from d' under σ'.

Definition 4. (Consistent partial orders). *Let σ and σ' be any two feasible n-tuple of strategies for the players in the dynamic game $\mathcal{G}$ and let $\succ_\sigma$ and $\succ'_\sigma$ be the two corresponding induced partial orders of the directional component of the state space D. We say that $\succ_\sigma$ and $\succ_{\sigma'}$ are pairwise consistent if and only if for any $d', d \in D$ we have*

$$\begin{aligned} &\text{if } d' \succ_\sigma d \text{ then } d \not\succ_{\sigma'} d', \text{ and} \\ &\text{if } d \succ_\sigma d' \text{ then } d' \not\succ_{\sigma'} d, \end{aligned} \tag{5}$$

in other words if a transition from d to d' is possible under strategy σ, the opposite transition should not be possible under some other strategy σ', and vice versa.

It is worth noting that the definition of consistency is silent about the non-directional component of the state space, allowing for various strategies to induce any transitions between points that only differ in non-directional dimensions. Given the concept of consistent partial orders, we can formally define the concept of a DDG.

Definition 5. (DDGs). *We say that a dynamic Markovian game $\mathcal{G}$ with finite state space S and set of all feasible strategies $\Sigma(\mathcal{G})$ is a DDG if given the decomposition of the state space into directional and non-directional components $S = D \times X$, the following two conditions hold: (1) every strategy $\sigma \in \Sigma(\mathcal{G})$ has no loops in the directional component D according to Definition 3, and (2) the elements of the set of induced partial orders on D, $\{\succ_\sigma | \sigma \in \Sigma(\mathcal{G})\}$, are pairwise consistent according to Definition 4.*

In the next section, we present a more practical criterion to judge whether a particular game is DDG or not, which is based on the notion of *common directionality* of the game rather than strategy-specific directionality as in Definition 5.

2.3. *Partitioning the state space into stages and substages*

Even though the different strategy specific partial orders $\succ_\sigma$ are consistent with each other, they may nevertheless be different from each other. To define the *state recursion algorithm* for computing an MPE of the game $\mathcal{G}$, we need to introduce a concept of strategy-independent common directionality. In doing so, we invoke the notion of the coarsest common refinement (*i.e. join*) of the set of all strategy specific partial orders $\{\succ_\sigma | \sigma \in \Sigma(\mathcal{G})\}$. In this section, we prove its existence and use this partial order to define the *stages* of the overall DDG. We show how the stages of the game are totally ordered by construction, enabling backward induction over the state space. Moreover, we prove that this ordering allows for the overall game $\mathcal{G}$ to be recursively decomposed into a sequence of subgames, the equilibria to which we use to construct a Markov perfect equilibrium of the overall game. We start with the definition of a *refinement* of a partial order.

Definition 6. (Refinement of a partial order). *Let $\succ_\sigma$ and $\succ_{\sigma'}$ be two partial orders of the elements of the set D. We say that $\succ_{\sigma'}$ is a refinement of $\succ_\sigma$ if and only if for any d',*

$d \in D$ we have

$$d' \succ_\sigma d \implies d' \succ_{\sigma'} d, \tag{6}$$

or equivalently that $\succ_\sigma \subset \succ_{\sigma'}$ with inclusion operator defined as inclusion of the sets of ordered pairs that constitute the binary relations.

It is possible for two strategy-specific partial orders to be consistent, but neither to be the refinement of the other. In this case the information on the possible transitions in the state space under both strategies has to be aggregated into a common (strategy independent) notion of directionality. This is achieved with the help of refinements which by definition preserve such information.

Given a set of partial orders $\{\succ_\sigma | \sigma \in \Sigma(\mathcal{G})\}$, let $\succ_\mathcal{G}$ denote the coarsest common refinement (join) of the partial orders $\succ_\sigma$ induced by all feasible strategies $\sigma \in \Sigma(\mathcal{G})$. The following theorem guarantees the existence of the join and characterizes it as (the transitive closure of) the union of the strategy-specific partial orders $\succ_\sigma$, $\sigma \in \Sigma(\mathcal{G})$.

Theorem 1. (Strategy-independent partial order). *Let $\mathcal{G}$ be a directional dynamic game, and let $\{\succ_\sigma | \sigma \in \Sigma(\mathcal{G})\}$ be the set of pairwise consistent partial orders of D induced by all feasible Markovian strategies in the game. Then the join of this set is given by*

$$\succ_\mathcal{G} = TC(\cup_{\sigma \in \Sigma(\mathcal{G})} \succ_\sigma), \tag{7}$$

where $TC(\cdot)$ denotes the transitive closure operator, i.e. the smallest transitive binary relation that includes the binary relation in the argument.

Definition 7. (Induced DAG for a DDG). *Let $\mathcal{G}$ be a DDG with state space $S = D \times X$ where D is the directional component of the state space. Let $D(\mathcal{G})$ denote the DAG whose vertices are the elements of D and whose edges $d \to d'$ correspond (one-to-one) to $d \succ_\mathcal{G} d'$ for every pair $d, d' \in D$. Then we say that $D(\mathcal{G})$ is the DAG induced by the DDG $\mathcal{G}$.*

Consider a vertex $d \in D$ of the DAG induced by $\mathcal{G}$. We say that *d has no descendants* if there is no $d' \in D$ such that $d' \succ_\mathcal{G} d$. The *terminal nodes* of $D(\mathcal{G})$, given by $\mathcal{N}(D(\mathcal{G}))$ is a subset of vertices $d \in D$ that have no descendants. We can consider $\mathcal{N}$ to be an *operator* which returns the terminal nodes of a DAG. Now let $D_1(\mathcal{G}) = D(\mathcal{G})$ and define $D_2(\mathcal{G})$ by

$$D_2(\mathcal{G}) = D(\mathcal{G}) - \mathcal{N}(D(\mathcal{G})), \tag{8}$$

where the "$-$" sign denotes the set difference operator, *i.e.* the set of points that belong to the first argument but not to the second. If follows that $D_2(\mathcal{G})$ is also a DAG, but it is a "sub-DAG" of the original DAG $D(\mathcal{G})$ created by removing the terminal vertices of $D(\mathcal{G})$. Since a DAG has no cycles, it is not hard to see that $\mathcal{N}(D(\mathcal{G})) \neq \varnothing$ for every DAG, *i.e.* every finite DAG must have at least one terminal node. Moreover, the nodes of every DAG induced by a finite state DDG $\mathcal{G}$ will be exhausted after a finite number of iterations of the recursive operator

$$D_{j+1}(\mathcal{G}) = D_j(\mathcal{G}) - \mathcal{N}(D_j(\mathcal{G})). \tag{9}$$

Lemma 2. (DAG recursion). *Let $\mathcal{G}$ be a finite state DDG with the induced DAG $D(\mathcal{G})$. Let $D_1(\mathcal{G}) = D(\mathcal{G})$ and define the sequence $\{D_j(\mathcal{G})\} = \{D_1(\mathcal{G}), D_2(\mathcal{G}), \ldots, D_\mathcal{T}(\mathcal{G})\}$ by the recursion (9). This sequence will terminate in a finite number of steps, i.e. $\mathcal{T} < \infty$.*

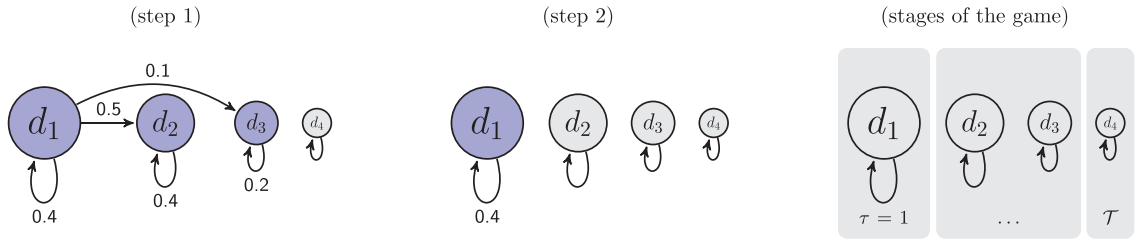

FIGURE 2
DAG recursion and stages of the DDG in Example 2 ($\mathcal{T}=3$).

All the nodes in the DAG $D_{\mathcal{T}}(\mathcal{G})$ have no descendants, and thus it represents the set of *initial nodes* of the original DAG $D(\mathcal{G})$. The corollary to Lemma 2 presented in the Appendix shows that the recursion (9) can also be used to check if an arbitrary directed graph is a DAG.

Example 4. Figure 2 provides an illustration of the DAG recursion for a game we considered in Example 2. Applying operator (9) to the DAG induced by this game (shown in left panel of Figure 1) yields in step 1 the left-most sub-DAG where node d_4 is removed. Terminal node d_4 is identified by the fact that all edges (except the loop to itself) point in and none point out. Applying the same principle in step 2 to the sub-DAG obtained in step 1, we find two new terminal nodes, namely d_2 and d_3. Removing these two nodes produces the new sub-DAG shown in the middle panel of Figure 2. Because the new sub-DAG contains only a single point d_1, the recursion terminates on the third step, inducing the partition of the directional component of the state space $\big\{\{d_1\},\{d_2,d_3\},\{d_4\}\big\}$ as shown in the right panel of Figure 2.

Given the whole sequence of DAGs $\{D_1(\mathcal{G}),D_2(\mathcal{G}),\ldots,D_{\mathcal{T}}(\mathcal{G})\}$ generated by the recursion in Lemma 2, let

$$\{D_1,\ldots,D_{\mathcal{T}}\}:\cup_{\tau=1}^{\mathcal{T}}D_\tau=D,\ D_\tau\cap D_{\tau'}=\varnothing \text{ when } \tau\neq\tau' \tag{10}$$

denote the partition of the directional component D of the state space, such that D_τ contains the points corresponding to the vertices of the DAG found in step $\mathcal{T}-\tau+1$ of the DAG recursion (9). In other words, let the indices of the subsets of D in the partition $\{D_1,\ldots,D_{\mathcal{T}}\}$ be *inverted* compared to the DAG recursion in Lemma 2. The right-most panel of Figure 2 presents this partition graphically for the game in Example 2.

Definition 8. (Stages of a DDG). *Let $\mathcal{G}$ be a finite state DDG, and let $\{D_1,\ldots,D_{\mathcal{T}}\}$ be the partition of the directional component of the state space D induced by the DAG recursion in Lemma 2. Let*

$$S_\tau=D_\tau\times X \tag{11}$$

denote the stage of the game $\mathcal{G}$, and index τ denote the index of the stage, so that S_1 denotes the initial stage of the game $\mathcal{G}$ and $S_{\mathcal{T}}$ denotes the terminal stage.

Note that if the partition elements D_τ contain more than one element of D, then there can be no transitions between them by virtue of the way the partition $\{D_1,\ldots,D_{\mathcal{T}}\}$ was constructed. Suppose that $D_\tau=\{d_{1,\tau},\ldots,d_{n_\tau,\tau}\}\subset D$ where n_τ is the number of distinct points in D_τ. We refer to points $\{d_{i,\tau}\times X\}$ as a (i,τ)-*substage* of the game $\mathcal{G}$, since formula (11) can be extended to $S_\tau=D_\tau\times X=\cup_{i=1}^{n_\tau}\{d_{i,\tau}\times X\}$.

With the definition of stages and substages of the DDG $\mathcal{G}$ at hand, the state dynamics of the DDG $\mathcal{G}$ can be described as follows. Imagine that the game starts at a point $s_1=(d_1,x_1)\in S_1\subset S$

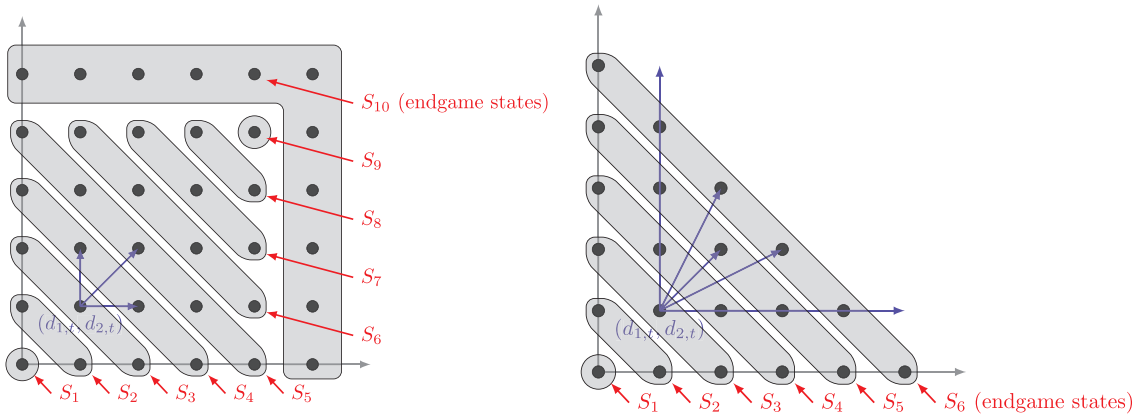

FIGURE 3

DAG recursion, stages and substages for Example 5 (left panel) and Example 6 (right panel)

Notes: Left panel depicts the patent race DDG by Judd *et al.* (2012b), $\mathcal{T}=10$. Right panel depicts the platform competition DDG by Dubé *et al.* (2010), $\mathcal{T}=6$.

at the initial stage S_1. It may remain in the substage $\{d_1 \times X\}$ for some time, moving freely between the points that only differ from s_1 in non-directional dimensions. Yet, while the game is in stage $\tau=1$, there can be no transitions to the points $(d_1', x_1) \in S_1 \subset S$ if $d_1 \neq d_1'$ due to the no-loop condition (4) which rules out any transitions between the substages of the same stage. At some time period a transition occurs to one of the subsequent stages S_τ, $\tau > 1$, namely to some point $s_\tau = (d_\tau, x_\tau) \in S_\tau \subset S$. Again, any transitions are possible within the substage $\{d_\tau \times X\}$, but the game will remain in the same substage until the state transitions to the next stage.

The DAG-recursion that constructs the stages of $\mathcal{G}$ does not rule out the possibility that a substage of some stage S_τ for $\tau < \mathcal{T}$ could be an absorbing class of states. While it is true that such states will be identified as terminal nodes of the DAG $D(\mathcal{G})$ of the DAG-recursion (9), this is relative to the strategy-independent partial order $\succ_{\mathcal{G}}$. In a particular MPE, S_τ could be an absorbing class relative to $\succ_\sigma$ for a particular MPE σ. But in many cases S_τ will all be *transient states* and only the final substage $\{d_{\mathcal{T}} \times X\} \subset S_{\mathcal{T}}$ of the final stage $S_{\mathcal{T}}$ will be an absorbing set of state points.

Example 5. (Patent race game). The left panel of Figure 3 illustrates the state space, stages and substages of the patent race model of Judd *et al.* (2012b). In their model, two competing firms invest in R&D to move the state of their knowledge forwards. Let $d_{i,t}$ denote the state of knowledge of firm i at time t. The firms start at a knowledge level $d_{i,t}=0$, and the first firm to exceed some threshold $\overline{d}$ is awarded a patent, giving it a monopoly on the subsequent rents it generates. The firms' state of knowledge advance stochastically and independently, conditional on their R&D effort, by at most one step per period. The left panel of Figure 3 illustrates the possible transitions at state $(d_{1,t}, d_{2,t}) = (1,1)$.

This game is clearly a DDG and the state $d_t = (d_{1,t}, d_{2,t})$ is the relevant directional state variable, as the state of the game can only move to one of the adjacent vertices to the northeast of d_t, or with some probability remain at the same point. This is true *regardless of the R&D investment of the firms* (since there is no "forgetting" in this model), and thus the directionality is *strategy independent.* That is, we can define the partial order $d' \succ d$ as the ordinary element-wise inequality $d' > d$ where each component of the state d' is strictly larger than the corresponding element of d. Applying the DAG-recursion operator (9) to the DAG induced by this game we obtain the $\mathcal{T}=10$ stages for a game with six points on the grid of d with $\overline{d}=6$. The end game is

trivial: one of the firms has won the patent at this point so the payoffs for these states are just the present value of monopoly rents (if the firms reach the threshold at the same time, a tie-breaking rule is assumed to split the monopoly rents). Importantly, even though the end game stage $S_{\mathcal{T}}$ contains $n_{\mathcal{T}} = 11$ substages, these points can be solved independently from one another, as the payoffs of the firms in a situation when one of them reached the monopoly power with a particular state of knowledge of the other firm are independent of the payoffs in any other possible end game state.

Example 6. (Platform competition game). The right panel of Figure 3 illustrates the state space, stages, and substages of a finite state version of the platform competition game of Dubé *et al.* (2010). This is a model where two competing video game consoles sell to a finite population of N consumers. The firms earn profits from sales of the video consoles plus royalties from third-party software that is licensed to run on these consoles. The authors assume that consumers are forward-looking, dynamic optimizers who will choose at most one of the two consoles, but may delay their purchase based on their belief of the future prices of the consoles and the selection of video games available to each of them (which is a function of the market share).

The right panel of Figure 3 illustrates a game where there are $N = 5$ consumers in the market. At state $d_t = (d_{1,t}, d_{2,t}) = (1, 1)$ a total of 2 consumers have purchased video consoles, one each of the two different consoles offered for sale. This game is clearly a DDG as the state d_t evolves directionally to the northeast quadrant of S from any point $d_t \in S$. Under the assumption that there are *IID* extreme value shocks that affect consumers' purchase decisions, there will be a positive probability of transiting to any point to the northeast of d_t regardless of the pricing strategies of the firms for their video consoles. This is illustrated by the arrows that reach to all points to the northeast of $d_t = (1, 1)$, representing the positive probability that the state can transit from d_t to any of these points.

Thus, a strategy-independent partial order $\succ$ also exists for this game induced by the element-wise inequality over S, *i.e.* $d' \succ d$ if and only if $d' > d$ element-wise. The DAG-recursion decomposition of this game results in $\mathcal{T} = 6$ stages, and once again the end game S_6 is trivial: there are no more consumers left to buy the video consoles, so the values of the two firms in these states equal the present value of their respective video game royalties. But at stage S_5 a total of four customers have bought the video consoles, so there is one remaining customer the firms are competing to sell to. Similar to the previous example, even though each stage contains several substages (in fact $n_\tau = \tau$), these points are independent and can be solved independently of each other. At each of the subgames originating in the points of S_5, there is duopoly price competition by the two firms to try to entice the last customer to buy their console. State recursion involves solving the game by backward induction starting at each subgame of the the end game states S_6 and working backward through each stage S_j, $j \leq 6$, represented by the successive diagonals in the right panel of Figure 3.

2.4. *Stage games and subgame perfection*

Recall that the DAG induced by the DDG $\mathcal{G}$ represents all possible transitions between the elements of the directional component of the state space D under any feasible strategy. Therefore, by virtue of the way the stages are constructed, once the state of the game reaches some point s at stage τ, *i.e.* $s \in S_\tau$, there is zero probability that the state will return to any point $s' \in S_{\tau'}$ at any previous stage $\tau' < \tau$ under any feasible strategy $\sigma \in \Sigma(\mathcal{G})$. This ordering allows us to define a new concept of a *stage game* that provides the basis for the backward induction solution method for the overall DDG $\mathcal{G}$ that we refer to as *state recursion.*

Definition 9. (Subgames of a DDG). *Let $\mathcal{G}$ be a finite state DDG, and let $\{S_1,\ldots,S_{\mathcal{T}}\}$ be the partition of S into stages. Define Ω_τ and $\Omega_\tau(d)$ as subsets of S by*

$$\Omega_\tau = \cup_{t=\tau}^{\mathcal{T}} S_t, \quad \Omega_\tau(d) = \{d\times X\} \cup \left(\cup_{t=\tau+1}^{\mathcal{T}} S_t\right), \tag{12}$$

and let $\mathcal{G}_\tau$ and $\mathcal{G}_\tau(d)$ denote the DDG with state spaces Ω_τ and $\Omega_\tau(d)$, respectively. Let the other elements of the game (number of players, time horizon, utility functions, discount factors, action sets, and laws of motion) be properly restricted versions of the elements of the original game $\mathcal{G}$. Then we say that $\mathcal{G}_\tau$ is the τ-subgame of the DDG $\mathcal{G}$ and that $\mathcal{G}_\tau(d)$ is the d-subgame of $\mathcal{G}$.

Lemma 3. (First-stage τ-subgame). *For a finite state DDG $\mathcal{G}$ we have $\mathcal{G}_1 = \mathcal{G}$.*

To account for the possibility of multiple equilibria in the subgames of $\mathcal{G}$, let $\mathcal{E}(\cdot)$ denote the set of all Markov perfect equilibria of the whole DDG or its subgames. In case there are more than one MPE in some of d-subgames $\mathcal{G}_\tau(d)$ in the stage τ, the equilibria in the d'-subgames at the earlier stages $\tau' < \tau$ from which a transition to d is possible ($d \succ_{\mathcal{G}} d'$) will be dependent on which MPE is played in the d-subgames of the later stages of the DDG. This implies that in case of multiplicity of equilibria in $\mathcal{G}$ (and thus its subgames), the solution computed by backward induction depends on the ESR that selects one of the equilibria at every d-subgame of $\mathcal{G}$, and thus induces (or selects) a particular MPE in the whole game.

Definition 10. (Equilibrium selection rule). *Let Γ denote a deterministic rule for selecting one of the MPE from every d-subgame $\mathcal{G}_\tau(d)$ or τ-subgame $\mathcal{G}_\tau$, i.e.*

$$e(\mathcal{G}_\tau) = \Gamma\left(\mathcal{E}(\mathcal{G}_\tau)\right), \;\; e(\mathcal{G}_\tau(d)) = \Gamma\left(\mathcal{E}(\mathcal{G}_\tau(d))\right) \;\; \forall d \in D, \tag{13}$$

where $e(\mathcal{G}_\tau(d)) \in \mathcal{E}(\mathcal{G}_\tau(d))$ and $e(\mathcal{G}_\tau) \in \mathcal{E}(\mathcal{G}_\tau)$ are particular selected MPE from the set of all MPE of $\mathcal{G}_\tau(d)$ and $\mathcal{G}_\tau$, and projections $e_\sigma(\mathcal{G}_\tau(d))$, $e_V(\mathcal{G}_\tau(d))$, where $e_\sigma(\mathcal{G}_\tau)$ and $e_V(\mathcal{G}_\tau)$ give the n-tuple of the players' strategies and the n-tuple of the value functions constituting the selected equilibrium in the corresponding subgames $\mathcal{G}_\tau(d)$ and $\mathcal{G}_\tau$.

We show below that because the substages $\{d\times X\}$ within each stage S_τ do not communicate, the MPE of the stage τ d-subgames $\mathcal{G}_\tau$ can be constructed from the simpler MPE of d-subgames $\mathcal{G}_\tau(d)$. Further, we can restrict our search for MPE of the d-subgames to *continuation strategies* which only require calculating equilibria on the state space $\{d\times X\}$ of the stage game, and then revert to an already calculated (and in case of multiplicity, selected by the ESR Γ) MPE for all subsequent subgames after stage τ. The power of the state recursion algorithm comes from its ability to decompose the problem of finding an MPE of the large and complex overall DDG $\mathcal{G}$ into the much more tractable problem of recursively finding equilibria for an appropriately defined sequence of the *generalized stage games* (which we also simply refer to as "stage games") over the smaller set of feasible continuation strategies.[11]

11. Note that our definition of stage games is different from the definition that is traditionally used in the theory of repeated games. In a repeated game, the stage games are *single period games* while the whole repeated game is a finite or infinite repetition of these stage games. In our setting, each stage game is itself generally a dynamic game, which is played for a random length of time until the state of the system transits out of the substage that defines the state space of this stage game.

Definition 11. (Continuation strategies). *Let $\mathcal{G}$ be a finite state DDG, and consider a particular stage $\tau \in \{1,\dots,\mathcal{T}\}$ of this game. Given a d-subgame $\mathcal{G}_\tau(d)$, define the d-continuation strategy $\sigma_\tau\left(s\middle|\{d\times X\}, e_\sigma(\mathcal{G}_{\tau+1})\right)$ to be any feasible Markovian strategy for points $s\in\{d\times X\}$, $d\in D_\tau$ that reverts to the MPE strategy $e_\sigma(\mathcal{G}_{\tau+1})$ in the $\tau+1$-subgame $\mathcal{G}_{\tau+1}$. That is,*

$$\sigma_\tau\left(s\middle|\{d\times X\}, e_\sigma(\mathcal{G}_{\tau+1})\right) = \begin{cases} \sigma(s) & \text{if } s\in\{d\times X\},\ d\in D_\tau \\ e_\sigma(\mathcal{G}_{\tau+1}) & \text{otherwise,} \end{cases} \tag{14}$$

where $\sigma(s)$ is any feasible Markovian strategy in s, i.e. $\sigma_i(s):\{d\times X\}\to\mathcal{P}(A)$ for $i\in\{1,\dots,n\}$, $s\in\{d\times X\}$ and $d\in D_\tau$. Similarly, define a τ-continuation strategy $\sigma_\tau\left(s\middle|S_\tau, e_\sigma(\mathcal{G}_{\tau+1})\right)$ to be any feasible Markovian strategy for points $s\in S_\tau$ that reverts to an MPE strategy $e_\sigma(\mathcal{G}_{\tau+1})$ in the $\tau+1$-subgame $\mathcal{G}_{\tau+1}$. That is,

$$\sigma_\tau\left(s\middle|S_\tau, e_\sigma(\mathcal{G}_{\tau+1})\right) = \begin{cases} \sigma(s) & \text{if } s\in S_\tau \\ e_\sigma(\mathcal{G}_{\tau+1}) & \text{otherwise,} \end{cases} \tag{15}$$

where $\sigma(s)$ is any feasible Markovian strategy in point $s\in S_\tau$.

Definition 12. (Stage game). *Let $\mathcal{G}$ be a finite state DDG, and consider a particular stage of the game $\tau\in\{1,\dots,\mathcal{T}\}$ and $d\in D_\tau$. A d-stage game, $\mathcal{SG}_\tau(d)$, is a d-subgame $\mathcal{G}_\tau(d)$ where the set of feasible strategies is restricted to continuation strategies (14). Similarly, we define $\mathcal{SG}_\tau$ to be the τ-stage game by restricting the set of all feasible Markovian strategies in the τ-subgame to the continuation strategies (15).*

Let $\mathcal{E}(\mathcal{SG}_\tau(d))$ and $\mathcal{E}(\mathcal{SG}_\tau)$ denote the set of all MPE of the d-stage game $\mathcal{SG}_\tau(d)$ and τ-stage game $\mathcal{SG}_\tau$, respectively. By establishing how the equilibria of the d-stage games relate to the equilibria of the τ-stage game, the next theorem provides the theoretical grounds for treating the d-stage game equilibria as the building blocks of the MPE of the whole game $\mathcal{G}$.

Theorem 2. (Decomposition of the stage game). *Let $\mathcal{G}$ be a finite state DDG with $\mathcal{T}$ stages. At each stage $\tau\in\{1,\dots,\mathcal{T}\}$ the following decomposition holds*

$$\mathcal{E}(\mathcal{SG}_\tau) = \cup_{i=1}^{n_\tau}\mathcal{E}(\mathcal{SG}_\tau(d_{i,\tau})),\ \ \mathcal{E}(\mathcal{SG}_\tau(d_{i,\tau}))\cap\mathcal{E}(\mathcal{SG}_\tau(d_{j,\tau})) = \varnothing,\ i\neq j, \tag{16}$$

where $d_{i,\tau}\in D_\tau$, $i\in\{1,\dots n_\tau\}$, and the union of the possible equilibria in the various d-stage games can be interpreted as defining an equilibrium (σ, V) as per Definition 1 whose domain is the union of the disjoint domains $\{d_{i,\tau}\times X\}$ of the equilibria of the d-stage games.

Thus, computed equilibria of the d-stage games can be "combined" into the equilibria of the τ-stage games. The start of the backward induction procedure is the last stage $\mathcal{T}$, where the stage games and the subgames of $\mathcal{G}$ coincide, as shown by the following lemma.

Lemma 4. ($\mathcal{T}$-stage game). *Let $\mathcal{G}$ be a finite state DDG, and consider the final stage of the game $\mathcal{T}$. It follows $\forall d\in D_\mathcal{T}\ \mathcal{SG}_\mathcal{T}(d) = \mathcal{G}_\mathcal{T}(d)$ and $\mathcal{SG}_\mathcal{T} = \mathcal{G}_\mathcal{T}$.*

By construction it follows that $\Sigma(\mathcal{SG}_\tau(d))\subset\Sigma(\mathcal{G}_\tau(d))$, *i.e.* the set of feasible Markovian strategies in a d-stage game $\mathcal{SG}_\tau(d)$ is a subset of the set of feasible Markovian strategies in the d-subgame $\mathcal{G}_\tau(d)$. Similarly the set of feasible Markovian strategies in the τ-stage game $\mathcal{G}_\tau$ is a

subset of the feasible Markovian strategies in the τ-subgame $\mathcal{G}_\tau$. In other words, considering the stage games with their restricted strategy space indeed reduces the computational burden for the state recursion algorithm. The next theorem shows that in computing the MPE of the d-subgame $\mathcal{G}_\tau(d)$ we can restrict attention to finding the MPE of the stage game $\mathcal{SG}_\tau(d)$ within the subclass of continuation strategies.

Theorem 3. (Subgame perfection). *Let $\mathcal{G}$ be a finite state DDG with $\mathcal{T}$ stages. Let $\mathcal{E}(\mathcal{SG}_\tau(d))$ and $\mathcal{E}(\mathcal{SG}_\tau)$ denote the set of all MPE of the d-stage game $\mathcal{SG}_\tau(d)$ and τ-stage game $\mathcal{SG}_\tau$, respectively. Let $\mathcal{E}(\mathcal{G}_\tau(d))$ and $\mathcal{E}(\mathcal{G}_\tau)$ denote the set of all MPE of the d-subgame $\mathcal{G}_\tau(d)$ and τ-subgame $\mathcal{G}_\tau$, respectively. Then $\forall d \in D_\mathcal{T}\, \mathcal{E}(\mathcal{SG}_\tau(d)) = \mathcal{E}(\mathcal{G}_\tau(d))$ and $\mathcal{E}(\mathcal{SG}_\tau) = \mathcal{E}(\mathcal{G}_\tau)$.*

Theorems 2 and 3 provide the foundation for the validity of the state recursion algorithm. They justify the backward induction process for computing the MPE of the DDG $\mathcal{G}$ by computing the MPE of every d-stage game, combining them to obtain the MPE of τ-stage game and thus MPE of τ-subgame, restricting our search to the much smaller class of continuation strategies whose values have already been calculated in previous stages of a backward induction procedure we call *state recursion.* We formally present the state recursion algorithm in the next section.

2.5. *State recursion algorithm*

Definition 13. (State Recursion Algorithm). *Consider a finite state DDG $\mathcal{G}$ with $\mathcal{T}$ stages. The state recursion algorithm consists of the following nested do-loop of operations*

---

input : DDG $\mathcal{G}$, partition $\{D_1, \ldots, D_\mathcal{T}\}$, equilibrium selection rule Γ
output: MPE of the DDG $\mathcal{G}$

for $\tau = \mathcal{T}, \mathcal{T}-1, \ldots, 1$ **do**
  for $d \in D_\tau$ **do**
    compute all MPE of the d-stage game $\mathcal{SG}_\tau(d)$
    using an ESR Γ, select a particular MPE $e(\mathcal{SG}_\tau(d)) = \Gamma(\mathcal{E}(\mathcal{SG}_\tau(d)))$
  construct $e(\mathcal{SG}_\tau) = \cup_{d \in D_\tau} e(\mathcal{SG}_\tau(d))$
  by Theorem 2 $e(\mathcal{SG}_\tau)$ is a MPE for the τ-stage game $\mathcal{G}_\tau$
  if $\tau = \mathcal{T}$ **then**
    by Lemma 4, $e(\mathcal{SG}_\tau) = e(\mathcal{G}_\tau)$ is a MPE of the τ-subgame
  else
    by Theorem 3, $e(\mathcal{SG}_\tau) = e(\mathcal{G}_\tau)$ is a MPE of the τ-subgame
  construct the next iteration continuation strategies $\sigma_{\tau-1}\left(s \middle| \{d \times X\}, e_\sigma(\mathcal{G}_\tau)\right)$ using $e(\mathcal{G}_\tau)$
by Lemma 3 $e(\mathcal{G}_1) = e(\mathcal{G})$ is a MPE of the whole DDG $\mathcal{G}$

The state recursion algorithm computes an MPE for the whole DDG $\mathcal{G}$ by combining the results of $N = \sum_{\tau=1}^{\mathcal{T}} n_\tau$ computations of the equilibria of the d-stage games. This only needs to be done once, so that state recursion will find an MPE of $\mathcal{G}$ using only one "pass" of a recursive, backward induction process that loops through all of the d-stage games which can be solved independently and sequentially over the stages of the game τ starting at $\tau = \mathcal{T}$ and working

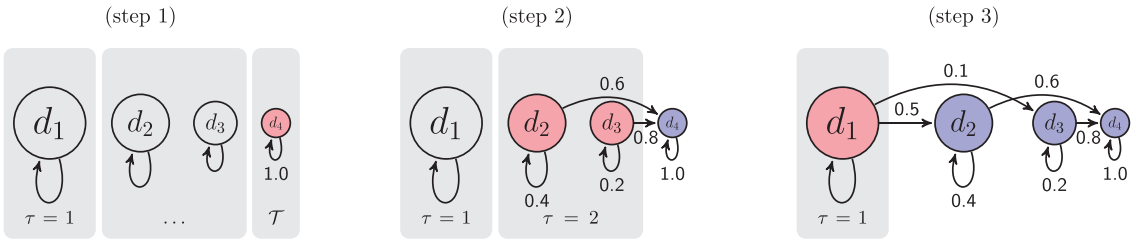

FIGURE 4
Graphical illustration of state recursion on the DAG $D(\mathcal{G})$ in Example 2

backward. Because it is always known beforehand how many iterations the algorithm has to take, it is always guaranteed to converge, as shown by the next theorem.

Theorem 4. (Convergence of State Recursion). *Let $\mathcal{G}$ be a finite state DDG. The state recursion algorithm given in Definition 13 computes an MPE of $\mathcal{G}$ in a finite number of steps.*

Example 7. Continuing with the DDG shrinking pie example (Example 2), Figure 4 illustrates state recursion on the induced DAG $D(\mathcal{G})$ that we introduced in the left panel of Figure 1, and partitioned into stages in the right panel of Figure 2. Because the game has three stages ($\mathcal{T}=3$), the state recursion algorithm requires three steps of the outer loop over τ. In the first step, we solve the end game which in this example is given by a single point d_4. Note that because there are no non-directional dimensions of the state space, d_4 should be interpreted as a point of the state space S. Thus, the terminal d-stage game constitutes the $\tau=\mathcal{T}$ stage game, which is by Lemma 4 is also a terminal subgame of the whole DDG. This subgame is essentially a repeated game in which the same state d_4 reappears in every period with probability 1 (as shown in the left panel of Figure 4). At the first step of the state recursion only the d_4-stage game has to be solved.

Given the solution of the d_4-stage game, the algorithm moves on to stage game $\tau=2$ shown in middle panel of Figure 4. This stage consists of two points d_2 and d_3, so $n_2=2$, which can be solved in any order during two iterations of the inner loop of the state recursion algorithm. In both cases, the continuation strategies are based on the equilibrium chosen in the d_4-stage game solved in step 1. After all MPE in the stage games are found, one particular equilibrium is chosen using the exogenously fixed ESR.

Once stage $\tau=2$ is solved, the algorithm moves on to stage $\tau=1$ shown in the right panel of Figure 4, where the last d-stage game, namely d_1-stage game is solved using the already known solutions in the rest of the points. By Lemma 3 the whole DDG is then solved.

3. RECURSIVE LEXICOGRAPHICAL SEARCH

The state recursion algorithm described in Section 2 finds a *single* MPE of the DDG $\mathcal{G}$ via a recursion that involves (a) finding *all* equilibria among continuation strategies at each d-stage game of the DDG $\mathcal{G}$, and then (b) selecting a single equilibrium from this set using some equilibrium selection rule Γ. The RLS algorithm presented in this section finds *all* MPE of $\mathcal{G}$ by systematically examining all feasible ESRs while at the same time recognizing the *interdependency of choices of MPE for stage games in different stages of* $\mathcal{G}$. For example, it is possible that one choice of MPE for the end game $\mathcal{G}_\mathcal{T}$ might result in a unique MPE at some earlier stage game $\mathcal{G}_\tau$, $\tau<\mathcal{T}$, whereas a different choice of MPE for the end game could result in *multiple* MPE existing at the same earlier stage game $\mathcal{G}_\tau$.

3.1. *Prerequisites*

Note that our general presentation of the RLS algorithm presumes the existence of a solution method to find *all* MPE in every d-stage game (*i.e.* equilibria within the class of continuation strategies). We show below that when this condition is satisfied RLS finds *all* MPE of the DDG $\mathcal{G}$. However, RLS also works if this algorithm can only find *some* of the equilibria of d-stage games. In the latter case, RLS is not guaranteed to find *all* MPE of $\mathcal{G}$, but it can still find, potentially, *very many* MPE of $\mathcal{G}$. It is more likely that we can find all MPE of each of the stage games of $\mathcal{G}$ than for $\mathcal{G}$ itself because the stage games generally have much smaller state spaces $\{d \times X\}$, and also because relying on Theorem 3 we restrict our search for MPE in the stage games to a smaller set of continuation strategies.

We can interpret RLS as a systematic way of directing the state recursion algorithm to "build" all possible MPE of $\mathcal{G}$ by enumerating all possible equilibrium selection rules Γ and constructing all possible MPE of every stage game of $\mathcal{G}$. Lemma 3 implies that this results in the set of all possible MPE for $\mathcal{G}$ itself. RLS is a remarkably efficient procedure for enumerating and building all possible MPE of $\mathcal{G}$. It achieves this efficiency by (1) re-using solutions from previously computed stage games of $\mathcal{G}$ wherever possible, and (2) by efficiently and rapidly disregarding large numbers of theoretically possible but *infeasible* combinations of stage game MPE of $\mathcal{G}$.

RLS is applicable to DDGs that have a *finite* number of possible MPE. As we noted in Section 2, Haller and Lagunoff (2000) proved that this is a generic propety of dynamic games with finite state spaces. The RLS algorithm will provide a complete enumeration of all of these equilibria. We also assume that each d-stage game has at least one equilibrium, implying that the whole DDG $\mathcal{G}$ also has at least one MPE.

3.2. *Equilibrium Selection Strings*

Let K denote the least upper bound on the number of possible equilibria in any d-stage game $\mathcal{G}_\tau(d)$ of the whole DDG $\mathcal{G}$. There is no need for the user to know K *a priori*, but using this notation simplifies the exposition of the algorithm. Recall that $N = \sum_{\tau=1}^{\mathcal{T}} n_\tau$ represents the total number of substages $\{d \times X\}$ of the DDG $\mathcal{G}$, which form the finest partition of the state space S according to equation (12). To construct an MPE of the whole game, the state recursion algorithm loops over all N of these substages to compute MPE of the d-stage games defined over the state spaces given by the substages.

We assume that in case of multiplicity the user fixes some ordering of the set of all equilibria at each d-stage game $\mathcal{G}_\tau(d)$, so that they can be indexed from 0 to at most $K-1$. To operationalize the use in the RLS algorithm of the equilibrium selection rules Γ defined in equation (13), we provide a numerical representation which we term *equilibrium selection strings* (ESS).

Definition 14. (ESS). *An ESS, denoted by γ, is a vector in Z_+^N (the subset of all vectors in R^N that have non-negative integer coordinates) where each coordinate of γ is an integer expressed in base K arithmetic, i.e. each coordinate (or "digit") of γ takes values in the set $\{0, 1, \ldots, K-1\}$. Further γ can be decomposed into subvectors corresponding to the stages of $\mathcal{G}$ which are ordered from right to left with the index of the stages $\tau = 1, \ldots, \mathcal{T}$, i.e.*

$$\gamma = (\gamma_{\mathcal{T}}, \gamma_{\mathcal{T}-1}, \ldots, \gamma_1), \tag{17}$$

where γ_τ denotes a sub-vector (sub-string) of γ with n_τ components where each digit, $\gamma_{i,\tau}$, $i=1,\ldots,n_\tau$ is given by

$$\gamma_\tau = (\gamma_{1,\tau},\ldots,\gamma_{n_\tau,\tau}), \tag{18}$$

with n_τ equal the number of substages of stage τ of the DDG $\mathcal{G}$.

The individual components or "digits" of the ESS $\gamma_{j,\tau}$ (in base K) point out which of the K possible MPE are selected in each of the $d_{j,\tau}$-stage games $\mathcal{SG}_\tau(d_{j,\tau})$ of every stage τ of $\mathcal{G}$. Thus, there is a one-to-one correspondence between an ESS γ and an ESR Γ when the number of MPE of the game $\mathcal{G}$ is finite ($K<\infty$). We use the notation $\Gamma=\gamma$ to signify that an ESR encoded by a particular ESS is used.

We use the subscripts notation $\gamma_{i,\tau}$ and γ_τ to denote a subvector (substring) of the ESS γ, and superscript to denote elements of a sequence of ESSs. Hence, γ^j will represent the j-th ESS in a sequence rather than the j-th component of the ESS γ. In particular, we let $\gamma^0=(0,\ldots,0)$ denote the initial ESS which is the selection rule that picks the first equilibrium in every d-stage game (which is always possible due to our assumption that $\mathcal{G}$ has at least one MPE).

It is important to note that the grouping of equilibrium strings into substrings or "sections" γ_τ corresponding to a right to left ordering of the stages of $\mathcal{G}$ as given in equation (17) is *essential* for the RLS algorithm to work correctly. However, due to the payoff-independence property for the n_τ component $d-$stage games $\mathcal{SG}_\tau(d_{i,\tau})$, $i=1,\ldots,n_\tau$ at each stage τ of $\mathcal{G}$ (Theorem 2), the ordering of the n_τ digits in each of the subvectors γ_τ (or "sections") is irrelevant and the RLS will generate the same results regardless of how the digits in each γ_τ substring are ordered.

Example 8. Consider a DDG with the induced DAG presented in the left panel of Figure 1 and the stages of the game presented in Figures 2 and 4. This game has $\mathcal{T}=3$ stages given by $S_1=\{d_1\}$, $S_2=\{d_2,d_3\}$ and $S_3=d_4$. Allow this game to deviate from the Rubinstein's bargaining model presented in Example 2 by the existence of multiple MPE and suppose that the maximum number of MPE for any of the four d-subgames is $K=3$. Then an example of an ESS would be $\gamma=(\gamma_3,\gamma_2,\gamma_1)=(\gamma_{1,3},\gamma_{1,2},\gamma_{2,2},\gamma_{1,1})=(0,2,2,1)$, indicating that the *first* MPE is selected in the only substage of stage $\tau=3$, the *third* MPE is selected in both substages of the stage $\tau=2$, and the *second* MPE is selected in the only substage of stage $\tau=1$. Different choices of MPE in stage $\tau=3$ may affect the number of MPE and the values of the MPE at stages $\tau=2$ and $\tau=1$. Different choices of MPE in stage $\tau=2$ may affect the number of MPE and the values of the MPE at stage $\tau=1$, but not at stage $\tau=3$.

The maximum number of ESS for any DGG $\mathcal{G}$ is K^N, which gives the upper bound on the number of MPE of $\mathcal{G}$. Nevertheless, not every ESS will correspond to an MPE of the game. Consider ESS $\gamma^1=(0,\ldots,0,1)$ which prescribes picking the first equilibrium in every d-stage game except $\mathcal{SG}_1(d_{n_t,1})$, where it prescribes picking the second equilibrium from the set of all MPE of this game $\mathcal{E}(\mathcal{SG}_1(d_{n_t,1}))$. Yet, it is possible $\mathcal{E}(\mathcal{SG}_1(d_{n_t,1}))$ contains only a single element, in which case we say that ESS γ^1 is *infeasible*. Similarly, among all possible ESS (which are just numbers $\{0,\ldots,K^N-1\}$ in base-K representation) there may be many other γ^j that are infeasible—*i.e.* they index non-existent MPE of the game. In general, due to interdependency between the τ-stage games, it is not a trivial task to enumerate the ESS that are guaranteed to be feasible. The RLS algorithm is designed to efficiently solve this task, namely to pick out feasible ESS from the K^N of them, spending minimum time on searching, and calling

the state recursion algorithm only when necessary to compute MPE for each feasible ESS it finds.

Definition 15. (Feasible ESS). *An equilibrium section string γ is feasible if all of its digits index an MPE that exists at each of the corresponding d-stage games of $\mathcal{G}$, $\forall d \in D$.*

Define an $N \times 1$ vector $ne(\gamma)$ to be the maximum number of MPE at each d-stage game of $\mathcal{G}$ under the ESR $\Gamma = \gamma$. We define $ne(\gamma)$ using the same format as the ESS, so that the digits of the ESS γ are in one-to-one correspondence with the elements of the vector $ne(\gamma)$ as follows:

$$ne(\gamma) = \Big(ne_{\mathcal{T}}, ne_{\mathcal{T}-1}\big(\gamma_{>\mathcal{T}-1}\big), \ldots, ne_1\big(\gamma_{>1}\big)\Big), \tag{19}$$

where $\gamma_{>\tau} = \big(\gamma_{\tau+1}, \ldots, \gamma_{\mathcal{T}}\big)$ is a $\mathcal{T} - \tau \times 1$ vector listing the equilibrium selection sub-string for stages of $\mathcal{G}$ higher than τ. In turn, $ne_\tau(\gamma_{>\tau})$ denotes the $n_\tau \times 1$ vector listing the maximum number of MPE in each of the $d_{i,\tau}$-stage games $\mathcal{SG}_\tau(d_{i,\tau})$, $i = 1, \ldots, n_\tau$ of stage τ of $\mathcal{G}$,

$$ne_\tau(\gamma_{>\tau}) = \Big(ne_{1,\tau}\big(\gamma_{>\tau}\big)\big), \ldots, ne_{n_\tau,\tau}\big(\gamma_{>\tau}\big)\Big). \tag{20}$$

The vector $ne(\gamma) \in Z_+^N$ summarizes how the number of possible MPE in any d-stage game at stage τ depends on the choices of the MPE at the $(\tau+1)$-stage games that are represented by the equilibrium selection substring $\gamma_{>\tau} = \big(\gamma_{\tau+1}, \ldots, \gamma_{\mathcal{T}}\big)$. We use the notation $ne_\tau(\gamma_{>\tau})$ to emphasize that the number of MPE at stage τ depends only on the equilibria selected at higher stages of $\mathcal{G}$. Notice that in the end game $\mathcal{T}$ there are no further stages of the game, so the maximum number of MPE in this stage, $n_{\mathcal{T}}$, does not depend on any substring of the ESS γ. Further, by the decomposition property for stage games in any stage τ of $\mathcal{G}$ (Theorem 2), the number of possible MPE at every d-stage game $\mathcal{SG}_\tau(d_{i,\tau})$, $i = 1, \ldots, n_\tau$ of stage τ depends only on the ESS $\gamma_{>\tau}$ and not on the choice of MPE in other d-stage games $\mathcal{SG}_\tau(d_{j,\tau})$, $j \neq i$ at the same stage τ. The following lemma formalizes this discussion.

Lemma 5. (Feasibility of ESS). *In a DDG $\mathcal{G}$ with $\mathcal{T}$ stages the ESS γ is feasible if and only if*

$$\gamma_{i,\tau} < ne_{i,\tau}(\gamma_{>\tau}), \;\; \tau = 1, \ldots, \mathcal{T}, \;\; i = 1, \ldots, n_\tau. \tag{21}$$

By assumption, we have $K = \max_{\tau=1,\ldots,\mathcal{T}} \max_{i=1,\ldots,n_\tau} \{ne_{i,\tau}(\gamma_{>\tau})\}$. However, in the operation of the RLS algorithm it is clear that we do not have to loop through all K digits $\{0, \ldots, K-1\}$ for every values of a candidate ESS γ (or equivalently cycle through all numbers $\{0, \ldots, K^N - 1\}$ in base-K representation) to check feasibility. We will generally have to check far fewer than K^N possible ESS for feasibility. But it should be evident that due to the one-to-one correspondence between an ESS and the integers $\{0, \ldots, K^N - 1\}$, a simple do-loop over them is a way to systematically enumerate all possible ESS, and thus all possible choices of MPE at each d-stage game of $\mathcal{G}$. However, this "brute force" enumeration is not efficient because typically there are huge gaps between the feasible ESS in this full enumeration loop resulting from the fact that many of the component stage games of $\mathcal{G}$ may have fewer than K of MPE. We devise a vastly more efficient approach that exploits the variability in the number of MPE of different stage games, and *jumps* directly to the next *feasible* ESS γ. Consequently, the RLS algorithm has a run time that is *linear* in $|\mathcal{E}(\mathcal{G})|$, the total number of MPE of the whole DDG. However, to describe this more efficient search procedure, we need to introduce some basic facts about *variable base arithmetic*.

3.3. *Variable base arithmetic*

We say an ESS γ has a *variable base* (also known in computer science as *mixed radix numeral systems*) if the different digits of ESS γ are expressed in different bases. Let the bases for the individual components of the ESS γ be given by the vector of integers $ne(\gamma)$, the number of MPE for each of the d-stage games of $\mathcal{G}$ after the state recursion algorithm was invoked with ESR $\Gamma=\gamma$. Continuing example 8, if a feasible $\gamma=(0,2,2,1)$ selects a particular MPE in the three stages of $\mathcal{G}$, suppose the corresponding number of equilibria in these stages is $ne(\gamma)=(1,3,3,3)$. Then the first component $\gamma_{1,3}=0$ is expressed in base=1 and can only have a value of 0, while the other components are expressed in base-3 and can take values from $\{0,1,2\}$.

An ESS γ is a variable base representation of an integer in very much the same way as before when all digits of γ have the same base K. Let $\iota: Z_+^N \to Z_+$ be the function that maps ESS of length N to integers. Then we have

$$\iota(\gamma)=\sum_{j=1}^{N}\gamma_{i(j),\tau(j)}\prod_{j'=1}^{j-1}ne_{i(j'),\tau(j')}(\gamma_{>\tau(j')}) \tag{22}$$

where $\gamma_{i(j),\tau(j)}$ is the j-th component of the ESS γ and $ne_{i(j),\tau(j)}(\gamma_{>\tau(j)})$ is the j component of the corresponding bases for the digits of the ESS γ. Continuing the example above, $\iota(0,2,2,1)=1+2\times3+2\times3\times3+0\times3\times3\times3=25$.

Since an ESS γ can be viewed as a variable-base representation of an integer, we can do all of the ordinary arithmetic. Addition can be done as in elementary school for numbers in base-10, namely to start on the right and add the first digits, "carrying" the remainder modulus-10 to the next digit of the number if addition causes the first digit to exceed 10. The variable base addition is done the same way, except for using a different base for determining how much to carry in each successive digit of the number.

Define the *successor function* $\mathcal{S}: Z_+^N \to Z^N$ that returns γ' derived by adding 1 to the ESS γ and carrying out the addition process as described above in variable base arithmetic. Thus, $\iota(\mathcal{S}(\gamma))=\iota(\gamma)+1$, except for the situation when γ is the largest N digit number in the given variable base system, and thus adding 1 to it requires an additional digit. In the example above if $ne(\gamma)=(1,3,3,3)$, the largest number is $(0,2,2,2)$ which corresponds to $\iota(0,2,2,2)=2+2\times3+2\times3\times3+0\times3\times3\times3=26$. Since all of the components of an ESS γ are non-negative, let the value of the successor operator when there is an "overflow" to be a vector $(-1,\ldots,-1)\in Z^N$.

Lemma 6. (Successor function and feasibility). *Let the "jump function" $\mathcal{J}: Z_+^N \to Z_+^N$ be given by*

$$\mathcal{J}(\gamma)=\begin{cases} argmin_{\gamma'}\{\iota(\gamma') \text{ such that } \iota(\gamma')>\iota(\gamma) \text{ and } \gamma' \text{ is feasible}\}, \\ (-1,\ldots,-1) \text{ if there is no feasible } \gamma' \text{ satisfying } \iota(\gamma')>\iota(\gamma). \end{cases} \tag{23}$$

If γ is a feasible ESS in DDG $\mathcal{G}$, then $\mathcal{J}(\gamma)=\mathcal{S}(\gamma)$.

Lemma 6 shows that the variable base arithmetic is a powerful tool for *jumping* from one feasible ESS γ to the "smallest" ESS *after* γ which is also feasible. In other words, we can easily jump to the next feasible ESS in the lexicographical order by adding 1 to the current feasible ESS γ while treating γ as a number written in variable base arithmetic with bases $ne(\gamma)$.

3.4. *RLS Algorithm*

Having set up the machinery and showing how it is possible to jump directly from one feasible ESS to another using the jump (successor) function $\mathcal{J}(\gamma)$ we are now ready to provide a simple description of how the RLS algorithm works.

Definition 16. (RLS Algorithm). *Consider a finite state DDG $\mathcal{G}$ with $\mathcal{T}$ stages. The RLS algorithm consists of the following operations*

---

input : DDG $\mathcal{G}$
output: Set Λ of all feasible ESS which correspond to all MPE of the DDG $\mathcal{G}$
run the DAG recursion (9) to find all $N=\sum_{\tau=1}^{\mathcal{T}} n_\tau$ substages d of the game $\mathcal{G}$
initialize ESS sequence with always feasible ESS $\gamma^0=(0,0,\ldots,0,0)$, let $\Lambda=\{\gamma^0\}$
run the **state recursion algorithm** with $\Gamma=\gamma^0$
record the number of computed MPE in every d-stage game $\mathcal{G}_\tau(d)$ in the vector $ne(\gamma^0)$
for $k=1,2,\ldots$ **do**
    compute the next ESS in the sequence $\gamma^k=\mathcal{S}(\gamma^{k-1})$ using variable bases $ne(\gamma^{k-1})$
    denote τ_0 the stage corresponding to the highest updated digit of γ^k
    If $\gamma^k=(-1,\ldots,-1)$ **return** Λ
    by Lemma 6, γ^k is a feasible ESS
    run **state recursion algorithm** with $\Gamma=\gamma^k$ *only* for stages $\tau<\tau_0$ which are dependent on the highest updated stage τ_0
    record the number of computed MPE in every d-stage game $\mathcal{G}_\tau(d)$ in the vector $ne(\gamma^k)$
    update the set of feasible ESS found by RLS by setting $\Lambda=\Lambda\cup\{\gamma^k\}$

Note that the RLS algorithm repeatedly invokes the state recursion algorithm to solve the game only *partially*, which requires a trivial adjustment of the Definition 13. This is an important feature of RLS which cuts its run time approximately in half. There is no need to resolve the τ stage games unless the changes of the ESS γ^k at the current iteration relative to the ESS γ^{k-1} from the previous iteration could have affected the MPE in earlier stages of the DDG. The arrangement of the digits of the equilibrium selection string (21) ensures that resolving the stage games corresponding to the digits γ to the left of the highest update digit is unnecessary. Another trivial extension of the state recursion algorithm has to do with saving the quantities of the d-stage game MPE in the vector $ne(\gamma^k)$.

As mentioned above, the prior knowledge of the upper bound K on the number of MPE in the d-stage games is not necessary. This information is updated over the course of running the RLS algorithm, starting with the initialization at the always feasible ESS $\gamma^0=(0,\ldots,0)$. Each time the RLS algorithm encounters a new feasible ESS γ, it updates the maximum number of MPE at state points where the solution may have changed.

Theorem 5. (Strong convergence of RLS Algorithm). *Given the DDG $\mathcal{G}$, assume there exists a solution algorithm that can find all MPE of every d-stage game $\mathcal{G}_\tau(d),\forall\tau,d$, and that the set of all MPE $\mathcal{E}(\mathcal{G}_\tau(d))$ of every d-stage game is finite. Then the RLS algorithm will return the set $\Lambda=\{\gamma^0,\ldots,\gamma^{J-1}\}$ of all feasible ESS of game $\mathcal{G}$, and thus find all MPE of DDG $\mathcal{G}$, in at most $J=|\mathcal{E}(\mathcal{G})|$ steps, where $|\mathcal{E}(\mathcal{G})|$ is the total number of MPE of the DDG $\mathcal{G}$.*

Corollary 5.1. (Weak convergence RLS Algorithm). *Given the DDG $\mathcal{G}$, assume there exists a solution algorithm that can find at least one MPE of every d-stage game $\mathcal{G}_\tau(d), \forall \tau, d$, and that the set of found MPE $\mathcal{E}(\mathcal{G}_\tau(d))$ of every d-stage game is finite. Then RLS algorithm will return the set $\Lambda = \{\gamma^0, \ldots, \gamma^{J-1}\}$ of feasible ESS of game $\mathcal{G}$ with $J \geq 1$, and thus find some MPE of DDG $\mathcal{G}$, in at most $J \leq |\mathcal{E}(\mathcal{G})|$ steps.*

Theorem 5 is the central result of the article. It essentially concludes the development of the solution method for the class of DDG games by establishing that the decomposition approach does allow RLS to construct all possible MPE of these games. By breaking the complicated problem of computing MPE of the dynamic game into a series of much smaller and tractable problems of finding MPE of the d-stage games which are characterized by small state spaces and reduced sets of relevant strategies (*i.e.* continuation strategies), the RLS algorithm together with the state recursion algorithm are capable of finding and computing all MPE of the overall DDG $\mathcal{G}$.

For the DDG where the assumption of existence of the solution algorithm for every d-stage game appears too strong, the RLS method is still useful as shown by Corollary 5.1. The same principle of decomposing the complicated problem of computing MPE of the dynamic game into a series of more tractable problems applies even if the small problems appear to be still too complicated. Even in this reduced form, RLS is capable of computing a subset of the MPE of the overall game in a finite number of steps.

4. AN APPLICATION OF STATE RECURSION AND THE RLS ALGORITHM

In this section, we present two non-trivial examples of dynamic directional games and show how we can solve these games using the state recursion and the recursive lexicographical search algorithms. We consider two versions of a dynamic model of Bertrand price competition with cost-reducing investments analysed by Iskhakov *et al*. (2015). The first is the simultaneous move version of this pricing and investment game, all dimensions of which are directional, and thus the stage games are relatively simple. Our second example is the alternating move version of the same model. We introduce a state variable indicating which firm has the right to move in any given time period, thus allowing for alternating moves. Because the right of move alternates back and forth between the two players, this state variable is non-directional. We show, however, that it is still possible to find all-stage game MPE, despite the additional complications induced by the non-directional component. Consequently, the alternating move version of the leapfrogging model can also be handled by the RLS algorithm and we can thus find all MPE of this game as well.

4.1. *Bertrand price and investment game with simultaneous moves*

We begin with the simultaneous move version of the leapfrogging model of Iskhakov *et al*. (2015), which describes the economic motivation of the model in greater detail.

4.1.1. The model. Consider a discrete-time, infinite horizon stochastic game where two firms $j \in \{1, 2\}$ are producing an identical good at a constant marginal cost c_1 and c_2, respectively. We assume that the two firms are price setters, have no fixed costs and face no capacity constraints. We also assume that demand is perfectly inelastic. Under these assumptions, the Bertrand equilibrium for the two firms is for the cost leader to serve the entire market at a price

$p(c_1,c_2)=\max[c_1,c_2]$. Let $r_1(c_1,c_2)$ denote the expected profits that firm 1 earns in a single period of the Bertrand–Nash pricing game, given by

$$r_1(c_1,c_2)=\begin{cases}0, & \text{if } c_1\geq c_2,\\ \max[c_1,c_2]-c_1, & \text{otherwise,}\end{cases} \tag{24}$$

and the profits for firm 2, $r_2(c_1,c_2)$ are defined symmetrically, *i.e.* $r_2(c_1,c_2)=r_1(c_2,c_1)$.

Both firms can make an investment to replace their existing plant with a new production facility which would bring marginal cost to the current state-of-the-art level denoted c. If firm j purchases the current state-of-the-art technology, then after one period of construction time it can produce at the new marginal cost of production c, so starting from the next period $c_j=c$.

Exogenous stochastic technological progress drives down the state-of-the-art marginal cost of production over time according to a Markov process with transition probability $\pi(c'|c)$. With probability $\pi(c|c)$ there is no improvement in the state-of-the-art technology, and with probability $1-\pi(c|c)$ the technology improves, so that next period state-of-the-art production cost c' is a draw from some discrete distribution over the interval $[0,c)$. Both firms have equal access to the new technology conditional on paying an investment cost $K(c)$.

At each time period each firm j also incurs idiosyncratic "disruption costs" (or subsidies) $\eta\epsilon_j=(\eta\epsilon_{0,j},\eta\epsilon_{1,j})$ associated with the choices of *not to invest* and *to invest*, respectively. It is common knowledge among the two firms that $\{\eta\epsilon_1\}$ and $\{\eta\epsilon_2\}$ are independent IID (across choices, players, and time periods) Type I bivariate extreme value processes with common scale parameter $\eta\geq 0$. Firm j observes its current and past idiosyncratic investment shocks $\{\eta\epsilon_j\}$, but does not observe its future shocks or its opponent's past, present, or future idiosyncratic investment cost shocks. The presence of the private shocks leads to a game of incomplete information, but because they are serially independent and thus satisfy the conditional independence condition of Rust (1987), do not have to be included into the state space of the game.

The timing of events and the corresponding information structure in the model are as follows. Each period, both firms observe the state of the industry, set their prices and *simultaneously* decide whether or not to invest in the state-of-the-art production technology. In setting the prices, the two firms also act independently and simultaneously. Production in the current period is carried out with the existing plants independent of firms' investment decisions.

Assuming that the two firms are expected discounted profit maximizers and have a common discount factor $\beta\in(0,1)$, we search for stationary Markov Perfect Equilibria of the game defined in Definition 1. In particular, MPE of the investment and pricing game is a pair of value functions and a pair of strategies $(P_j(c_1,c_2,c),p_j(c_1,c_2))$, $j\in\{1,2\}$ where $P_j(c_1,c_2,c)\in[0,1]$ is firm j's probability of investing and $p_j(c_1,c_2)=\max[c_1,c_2]$ is firm j's pricing decision. The investment function $P_j(c_1,c_2,c)$ must maximize the expected discounted value of firm j's future profit stream taking into account the investment and pricing strategies of its opponent. The value functions V_j, $j=1,2$ take the form

$$V_j(c_1,c_2,c,\epsilon_{0,j},\epsilon_{1,j})=\max\left[v_{I,j}(c_1,c_2,c)+\eta\epsilon_{0,j},\, v_{N,j}(c_1,c_2,c)+\eta\epsilon_{1,j}\right] \tag{25}$$

where, $v_{N,j}(c_1,c_2,c)$ denotes the expected value to firm j if it does not acquire the state-of-the-art technology, and $v_{I,j}(c_1,c_2,c,m)$ is the expected value to firm j if it does. These expected values are given by

$$v_{N,j}(c_1,c_2,c)=r_j(c_1,c_2)+\beta EV_j(c_1,c_2,c,0), \tag{26}$$

$$v_{I,j}(c_1,c_2,c)=r_j(c_1,c_2)-K(c)+\beta EV_j(c_1,c_2,c,1), \tag{27}$$

where $EV_j(c_1,c_2,c,i)$ denotes the conditional expectation of firm j's next period value functions $V_j(c_1,c_2,c,\epsilon_{0,j},\epsilon_{1,j})$ depending on whether the firm invests $(i=1)$ this period or not $(i=0)$.

The expected value function summarize the firms' expectations about future technological development governed by $\pi(c'|c)$, and their opponent's investment and pricing decisions as well as the effects of future idiosyncratic cost shocks $\eta\epsilon_j$. Since the two firms move simultaneously, firm j's investment decision is probabilistic from the standpoint of firm $i \neq j$ because firm j's decision depends on the cost benefits/shocks $(\epsilon_{0,j}, \epsilon_{1,j})$ that only firm j observes. But since firm i knows the probability distribution of these shocks, it can calculate its belief about the probability that firm j will invest given the mutually observed state (c_1, c_2, c). Let P_j denote such beliefs of firm i. Given the assumption of extreme value distribution of $(\epsilon_{0,j}, \epsilon_{1,j})$, P_j is given by the binary logit formula

$$P_j(c_1,c_2,c) = \frac{\exp\{v_{I,j}(c_1,c_2,c)/\eta\}}{\exp\{v_{N,j}(c_1,c_2,c)/\eta\} + \exp\{v_{I,j}(c_1,c_2,c)/\eta\}}. \tag{28}$$

Firm i's belief of firm j's probability of not investing is then $1 - P_j(c_1, c_2, c)$.

Further, the distributional assumption for cost shocks $(\epsilon_{0,j}, \epsilon_{1,j})$ also allow us to express the conditional expectation $EV_j(c_1, c_2, c)$ for each firm j by the well-known closed-form log-sum formula

$$\begin{aligned}
&\int_{\epsilon_0^j}\int_{\epsilon_1^j} V_j(c_1,c_2,c,\epsilon_{0,j},\epsilon_{1,j}) q(\epsilon_{0,j}) q(\epsilon_{1,j}) d\epsilon_{1,j} d\epsilon_{0,j} = \\
&\eta \log\left[\exp\{v_{N,j}(c_1,c_2,c)/\eta\} + \exp\{v_{I,j}(c_1,c_2,c)/\eta\}\right] = \\
&\phi(v_{N,j}(c_1,c_2,c), v_{I,j}(c_1,c_2,c)),
\end{aligned} \tag{29}$$

where we use $\phi()$ to simplify the notation. Then the system of Bellman equations for the simultaneous move version of the model takes the form

$$\begin{aligned}
v_{N,1}(c_1,c_2,c) &= r_1(c_1,c_2) + \beta \int_0^c \big[P_2(c_1,c_2,c)\phi(v_{N,1}(c_1,c,c'), v_{I,1}(c_1,c,c')) + \\
&\quad (1 - P_2(c_1,c_2,c))\phi(v_{N,1}(c_1,c_2,c'), v_{I,1}(c_1,c_2,c'))\big]\pi(dc'|c). \\
v_{I,1}(c_1,c_2,c) &= r_1(c_1,c_2) - K(c) + \beta \int_0^c \big[P_2(c_1,c_2,c)\phi(v_{N,1}(c,c,c'), v_{I,1}(c,c,c')) + \\
&\quad (1 - P_2(c_1,c_2,c))\phi(v_{N,1}(c,c_2,c'), v_{I,1}(c,c_2,c'))\big]\pi(dc'|c), \\
v_{N,2}(c_1,c_2,c) &= r_2(c_1,c_2) + \beta \int_0^c \big[P_1(c_1,c_2,c)\phi(v_{N,2}(c,c_2,c'), v_{I,2}(c,c_2,c')) + \\
&\quad (1 - P_1(c_1,c_2,c))\phi(v_{N,2}(c_1,c_2,c'), v_{I,2}(c_1,c_2,c'))\big]\pi(dc'|c). \\
v_{I,2}(c_1,c_2,c) &= r_2(c_1,c_2) - K(c) + \beta \int_0^c \big[P_1(c_1,c_2,c)\phi(v_{N,2}(c,c,c'), v_{I,2}(c,c,c')) + \\
&\quad (1 - P_1(c_1,c_2,c))\phi(v_{N,2}(c_1,c,c'), v_{I,2}(c_1,c,c'))\big]\pi(dc'|c).
\end{aligned} \tag{30}$$

4.1.2. Directionality of the simultaneous move game. The state-of-the-art marginal cost of production, c_t, is trivially a *directional* state variable since it can only improve. It also has a natural absorbing state $c_t = 0$,[12] which together with some initial level of the state-of-the-art

12. We assume without loss of generality that the largest lower bound of the state-of-the-art cost is zero.

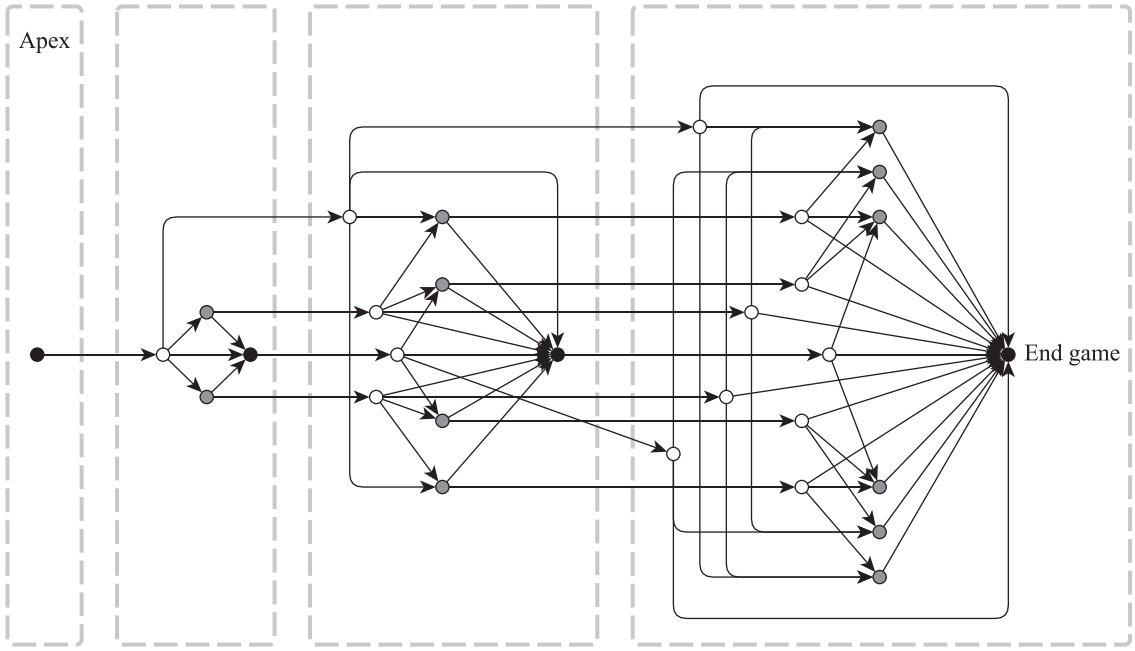

FIGURE 5

Possible transitions between state points in the dynamic Bertrand investment and pricing game

Notes: Each dot represents a vector (c_1, c_2, c). Dashed boxes enclose different layers of the state space pyramid: from the apex to the end game. White-coloured dots in each layer represent interior points ($c_1 > c, c_2 > c$), grey dots represent edges ($c_1 = c$ or $c_2 = c$), and solid black dots represent the corners ($c_1 = c_2 = c$). Only transitions from the transitive reduction are shown between layers, but a full set of transitions can be reconstructed by considering the transitive closure of the presented graph. The state variables c_1, c_2 and c are defined on a grid with $n = 4$ values. For clarity not all possible transitions are shown in the DAG, $\mathcal{D}(\mathcal{G})$.

cost c_0 allow for finite discrete approximations of the state space by discretizing the interval $[0, c_0]$. Further, it is easy to see that the remaining two state variables in the model, c_1 and c_2, are also directional because in the absence of depreciation once an investment is made by either firm, its marginal cost of production will equal the current state-of-the-art cost c. Hence, all state variables of the simultaneous move Bertrand pricing and investment game belong to the "directional" component of the state space. Using notation of Section 2, the directional variable is equal to the entire state vector, $d = (c_1, c_2, c)$, *i.e* $S = D$ and X is a singleton (*i.e.* can be treated as not existing). Because for every point of the state space $c_1, c_2 \geq c$, $S = D$ is a 3-dimensional pyramid with apex at the point (c_0, c_0, c_0) and the base given by a Cartesian product $[0, c_0] \times [0, c_0]$ on the (c_1, c_2) plane.

Figure 5 presents the induced DAG (see Definition 7) for the game with $n = 4$ points on the grid for costs. Each dot represents the state vector (c_1, c_2, c) and arrows represent possible transitions between points in the state space. Dashed boxes enclose different *layers* of the state space pyramid corresponding to different values of c: from the apex where $c = c_1 = c_2 = c_0$ to the base of the pyramid where $c = 0$ (the right-most box). White-coloured dots in each layer represent "interior points" in each layer, *i.e.* (c_1, c_2, c) where $c_1, c_2 > c$. The grey dots represent "edges", (c, c_2, c) and (c_1, c, c). The solid black dots represent the (c, c, c) "corners" where $c_1 = c_2 = c$.

The DAG in Figure 5 represents the coarsest common refinement (join) $\succ_{\mathcal{G}}$ of *strategy-specific* partial orders $\succ_\sigma$ over the state space given by the union over all feasible strategies $\sigma \in \Sigma(\mathcal{G})$ according to Theorem 1. In this case, $(c_1', c_2', c') \succ_{\mathcal{G}} (c_1, c_2, c)$ *iff* $c' < c$ or $c' = c$ and $c_1' < c_1$ or $c_2' < c_2$. Thus, only transitions between the "layers" of the state space that correspond to exogenous technological improvements (lower values of c) take place independently from the

actions of the players. All transitions within the layers are strategy specific, and the definition of $\succ_{\mathcal{G}}$ insures that it is the coarsest common refinement of all the strategy specific partial orders $\succ_{\sigma}$.

Consider, *e.g.* some interior point (c_1, c_2, c) at a time period when no technological improvement takes place. Under all strategies σ that assign a positive probability of investment to firm 1 in this point and zero probability of investment to firm 2, the game may transit to a point at the edge (c, c_2, c). Conversely, under all strategies σ that assign zero probability of investment to firm 1 and a positive probability of investment to firm 2, the game may transit to a point at the opposite edge (c_1, c, c). Finally, under all strategies σ that assign positive probabilities of investment to both firms there can be a transition to the corner (c, c, c). These transitions are indicated with arrows from white dots to grey and black dots, respectively. Under any strategy it is only possible to move from the edges to the corner (unless the technology improves) as indicated by the arrows from grey to black dots. If technology improves, so that $c' < c$, the state of the industry can move to the interior points at lower levels of the state space pyramid. These transitions are indicated with the arrows that cross the borders of the dashed boxes in Figure 5.

Figure 5 contains many points that are not connected (in one or several steps) indicating that they are not comparable under $\succ_{\mathcal{G}}$, *i.e.* under any strategy. It is not hard to verify that when any two points are not comparable, it is because they do not communicate, so the no-loop condition (as per Definition 3) is satisfied. Indeed, because from any interior point only transitions to the edges or the corner are possible, the inner points do not communicate. Similarly, because from any edge point the only possible transitions are to the corner, the points on the edges also do not communicate. Further, it is not hard to verify that there are no two strategies that result in the opposite transitions between any two points in the state space, implying that all strategy-specific partial orders $\succ_{\sigma}$ in the game are pair-wise consistent according to Definition 4. If this was not true, the graph in Figure 5 that illustrates the union of strategy-specific partial orders $\succ_{\sigma}$, would end up having two-way transitions between some points, *i.e.* loops, which would imply that the graph is not a DAG. Therefore, the Bertrand pricing and investment game satisfies Definition 5 and thus is a DDG.

Application of the DAG recursion algorithm (see Lemma 2) to the DAG in Figure 5 splits the state space into the stages by layer (indicated by dashed line in Figure 5) and the type of the points (indicated by colour in Figure 5). The end game stage ($\tau = \mathcal{T}$), is the $(0,0,0)$ corner, corresponding to the right-most black dot in Figure 5. The $\mathcal{T}$-stage game $\mathcal{G}(\mathcal{T})$ has a state space consisting of a single point $(0,0,0)$, and therefore effectively constitutes an infinitely repeated Bertrand pricing game with investments only taking place because of idiosyncratic private shocks when $\eta > 0$.

The next stage corresponding to $\tau = \mathcal{T} - 1$ consists of the $2(n-1)$ edge states of the form $(c_1, 0, 0)$ and $(0, c_2, 0)$. Thus, there are multiple values of the directional state variable in this stage, but because they do not communicate with each other, each separate point induces an infinite horizon d-subgame in which the cost vector may remain the same or change to $(0,0,0)$ at some future time period. So, the only possible "direction" of movement is to the stage $\mathcal{T}$. Because of no communication, each of these d-subgames can be solved independently of each other in the inner loop of the state recursion algorithm. By "solution" we mean finding an MPE of the d-stage game in accordance with Theorem 3, *i.e* within the class of continuation strategies that revert to the MPE in state $(0,0,0)$ (stage $\mathcal{T}$) that was already found in the previous step of the state recursion algorithm.

The next stage $\tau = \mathcal{T} - 2$ consists of the interior points in the bottom layer where $c_1, c_2 > c$, and $c = 0$. These points also do not communicate with each other, and thus form $(n-1)^2$ independent d-subgames which are solved taking into account the solutions on the edges and in the corner of the bottom layer. The stage after that, $\tau = \mathcal{T} - 3$, equals the (c, c, c) corner stage in the second to last layer of the game where $c > 0$, and so on.

Theorem 6. (Solution method for the d-stage games in simultaneous move leapfrogging game). *When $\eta = 0$ there exists an exact, non-iterative algorithm that finds all MPE of every d-stage game in the Bertrand pricing and investment game with simultaneous moves, given an equilibrium selection rule Γ. The number of possible MPE in any d-stage game is either 1, 3, or 5.*

Theorem 6 ensures that for the case of $\eta = 0$ the conditions for Theorem 5 are satisfied, which implies that the state recursion and RLS algorithms developed in the previous sections can find all MPE of the Bertrand pricing and investment game.

Figure 6 provides some intuition on how the algorithm is able to find all MPE of each stage game and why the number of MPE is either 1, 3, or 5. Each panel plots on the unit square the *best-response correspondences* of the two firms, *i.e.* mappings $P_1(c_1, c_2, c, P_2)$ and $P_2(c_1, c_2, c, P_1)$ that specify the probability of investment for each firm as a function of their belief about the probability of investment by their opponent, P_2 and P_1.

In the corner states (c, c, c), we prove that it is a dominant strategy for both firms not to invest, so the best-response correspondences take the value 0 regardless of the probability of the investment by the opponent. In these states, there is only a single pure-strategy MPE where neither firm invests. This situation is shown on the left-most panel of Figure 6, where the horizontal solid line (best response of firm 1) intersects with the vertical dashed line (best response of firm 2) only once in the point $(0, 0)$. The same situation is typical for the edge states (c_1, c, c) and (c, c_2, c).

However, in the interior states (c_1, c_2, c) where $c_1 > c$ and $c_2 > c$, it may be optimal for firm 1 to invest if P_2 is sufficiently small, but not if P_2 is sufficiently large. For some values of $P_2 \in (0, 1)$ it may so happen that firm 1 is indifferent between investing and not investing. At these values any probability of investing is equally good response for firm 1, and thus the best-response correspondence takes all values on the interval $[0, 1]$. Similarly firm 2 may be indifferent between investing and not investing for some values of P_1. It follows that the pair (P_1, P_2) then constitutes a mixed-strategy MPE of this stage game.

In the proof of Theorem 6, we characterize all such points of indifference and show that there can be at most two such points for each firm. The typical patterns of intersection of the two best-response functions corresponding to this result are shown in the three right-most panels of Figure 6. The second panel in the figure plots the best-response correspondences of the two firms when both have a single point of indifference. Notice that these graphs intersect at *three points* in this case. One intersection is at the mixed-strategy MPE, but there are two other intersections at the corners of the square. These two intersections correspond to pure-strategy *anti-coordination* MPEs where one firm invests and the other does not.

The third panel in Figure 6 shows the case when there are two values of P_2 that make firm 1 indifferent between investing and not investing. In this case, the best-response correspondence can be both hump-shaped and U-shaped. This leads to three MPE, one of which is pure and the other two are mixed. There can also be stage game MPE with 5 MPE, when the best-response correspondences are U-shaped or hump-shaped, as shown in the right-most panel of Figure 6.

The algorithm we developed that finds all MPE of every stage game is a *combinatoric algorithm*, *i.e.* it finds the MPE exactly (to the machine precision) by calculating the real roots of two quadratic equations and checking whether these roots lie on the unit interval. Further, the algorithm is *non-iterative*, *i.e.* there is no need for iterative root finding or equation solving using Newton-type method or successive approximations to compute firms' values or equilibrium probabilities of investment.

A similar result for the case of $\eta > 0$ is omitted for space considerations, and is available upon request. In this case, the analytical characterization of d-stage game MPE is more involved, and computing stage equilibria for every $d = (c_1, c_2, c)$ is more computationally costly and does requires iterative methods to compute the intersections of the firms' best-response functions and

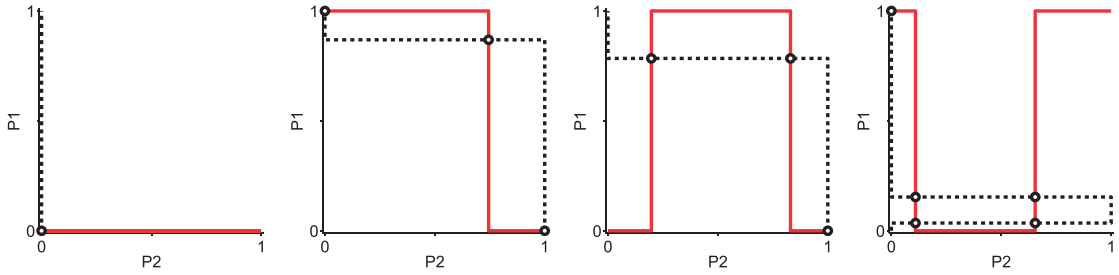

FIGURE 6

Best-response patterns and MPE of the simultaneous move stage game when $\eta = 0$

Notes: Typical patterns of the best-response correspondences of firm 1 (solid) and firm 2 (dashed) are shown on the unit square of the (P_2, P_1)-plane. Left panel shows the single pure strategy no-investment MPE in the corner (c, c, c)-stage games, and is also typical for the edge (c_1, c, c)-stage games and (c, c_2, c)-stage games $(c_1, c_2 > c)$. The right three panels show three typical MPE structures with 3 and 5 equilibria in the internal (c_1, c_2, c)-stages $(c_1, c_2 > c)$. The exact parameter values used to produce these plots are available upon request.

the implied value functions. Yet, the RLS algorithm allows us to compute all MPE of the overall leapfrogging game when $\eta > 0$.

4.1.3. Finding all MPE using RLS. Assume $\eta = 0$. Theorem 6 establishes that state recursion is guaranteed to find all d-stage game MPE given a fixed equilibrium selection rule Γ. In this section, we illustrate RLS algorithm using simultaneous move leapfrogging game with the number of points on the grid for costs is $n = 3$.

The first three rows in Figure 7 present a possible indexing of the ESS digits corresponding to d-substages within the stages of the game indexed with τ in the first row of the table. The top line contains indicators of the type of the points with "c", "e", and "i" denoting, respectively, corner, edge, and interior. Note that while there are $n = 14$ state points, there are only $\mathcal{T} = 7$ stages in this game and hence there are multiple ways we can order the state points within each stage τ and still obey the ordering of the stages of the game. The chosen order of digits of the ESS string is given in the second and third rows of the table in Figure 7.

Each column corresponds to a state point and thus to a digit in a ESS. The next three rows in Figure 7 explicitly specify the values of the state variables (c_1, c_2, c) corresponding to particular ESS digits. Starting from the right, the lowest digit represents the top layer the game with a single point $c_1 = c_2 = c = c_0 = 2$. As we explained above, the solution in this initial state depends on all subsequent points of the state space, whereas the opposite is true for the end game where $c_1 = c_2 = c = 0$ and which corresponds to the highest digit 14. The ESS digits are arranged in such a way as to obey the ordering of the stages of the game, namely stages with lower index τ are located to the right and correspond to the lower digits. The solution associated with a given digit in the ESS, $\gamma_{i,\tau}$, does not depend on the equilibrium selected at higher stages (digits to the right), but only depends on the equilibrium selected an lower states (digits to the left $\gamma_{>\tau}$).

We begin RLS with the initial ESS, γ^0 that consist of zeros at all 14 digits and solve for all MPE given the induced equilibrium selection rule, *i.e.* a rule that selects the "first" equilibrium in each d-stage game. Having computed all MPE using the state recursion algorithm, we obtain the number of MPE at each d-stage game, which we collect in the vector $ne(\gamma^0)$. The MPE at the corner and edge d-stage games have unique MPE, whereas interior substages can have either 1, 3, or 5 equilibria depending on parameters of the model. Figure 7 presents the case when $ne(\gamma^0)$ equals 3 for all interior points as indicated in the top "*ne*" line.

The next feasible ESS is found by adding 1 to ESS in $ne(\gamma^0)$ variable base arithmetic. Since $ne_{1,1} = 1$ (the two subscripts should be read from the first two rows in Figure 7), adding 1 changes

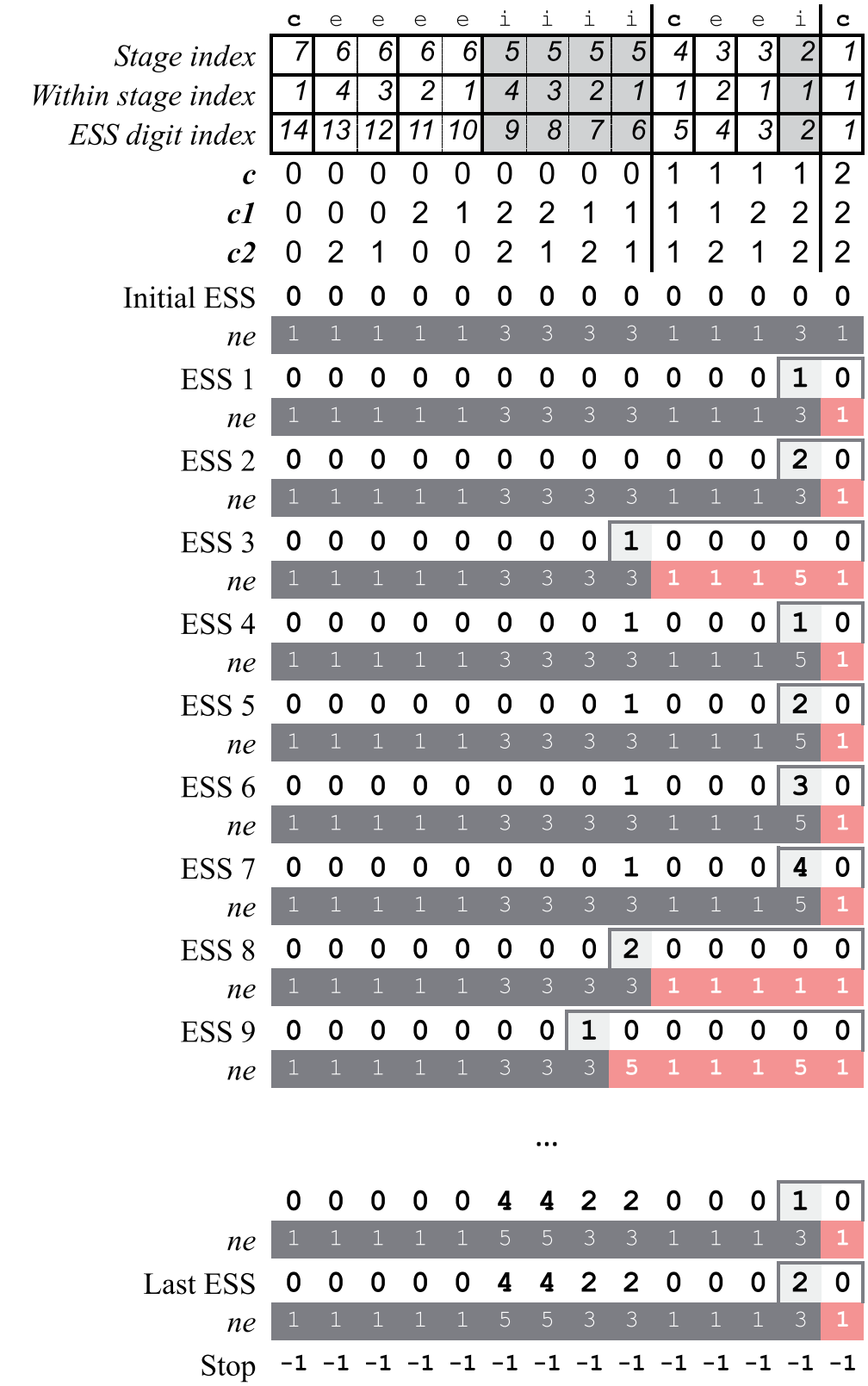

FIGURE 7

Graphic representation of RLS algorithm

Notes: Each column refers to a digit in the ESS and thus corresponds to a state point. The "corner" (where $c_1 = c_2 = c$), the "edges" (where $c_1 = c$ and $c_2 = c$), and the "interior" points (where $c_1 > c$ and $c_2 > c$) are marked with symbols "c", "e", and "i".

not the first but the second ESS digit. This implies that in principle the number of MPE in the stage game corresponding to the first ESS digit might change when the state recursion is run with the new ESS. As explained in Section 3.4 the state recursion only has to be run for the points which are affected by the change of ESS, in this case only for the point $(2,2,2)$. The updated number of stage game MPE is marked by colour in the next "*ne*" line. Again, adding 1 in $ne(\gamma^1)$ arithmetic changes the second ESS digit, which in γ^2 takes the value of 2.

Note that sometimes changing the ESS at lower levels not only affects the value functions and strategies at higher levels, it may also result in a different number of MPE. This is the case when we move from γ^2 to γ^3, where the number of MPE at the only point of stage $\tau = 2$ increases from 3 to 5. This causes no problem for RLS, as the base in the variable base arithmetic is updated accordingly. In general, the feasibility of each particular ESR string γ is simply defined through the set of inequalities on the digits given in Lemma 5. Using the simple successor function in variable base arithmetic ensures that the feasibility condition is satisfied, and we can continue the RLS loop until it is no longer possible to find an ESS that satisfies the feasibility constraint. When the process is completed we have found all MPE of the overall Bertrand price and investment DDG.

4.2. *Bertrand price and investment game with alternating moves*

We now turn to our second example, which is an alternating move version of the model outlined above. As before, we assume that firms simultaneously set their prices after having made their investment choices. However their investment choices are no longer made simultaneously. Instead, the right to move alternates between the two firms. Let $m \in \{1,2\}$ be a state variable that indicates which of the two firms is allowed to undertake an investment at a given time period. We will assume that $\{m\}$ evolves as an exogenous two-state Markov chain with transition probability $f(m'|m)$ independent of the other state variables (c_1, c_2, c), and that $f(m'|m)$ is common knowledge.

In the alternating move case, the Bellman equations for the two firms lead to a system of eight functional equations for $\left(v_{N,j}(c_1,c_2,c,m), v_{I,j}(c_1,c_2,c,m)\right)$ for $j,m \in \{1,2\}$. Note that the interpretation of the first subscript is slightly changed—N and I in the alternating move game denote if the investment is made in the current period by the firm that has the right to move. Below we write out the four Bellman equations for firm 1, but we omit the value functions for firm 2 to save space as they are defined similarly.

$$v_{N,1}(c_1,c_2,c,1) = r_1(c_1,c_2) + \beta f(1|1)\int_0^c \phi(v_{N,1}(c_1,c_2,c',1), v_{I,1}(c_1,c_2,c',1))\pi(c'|c)dc' + \beta f(2|1)\int_0^c ev_1(c_1,c_2,c')\pi(dc'|c)$$

$$v_{I,1}(c_1,c_2,c,1) = r_1(c_1,c_2) - K(c) + \beta f(1|1)\int_0^c \phi(v_{N,1}(c,c_2,c',1), v_{I,1}(c,c_2,c',1))\pi(c'|c)dc' + \beta f(2|1)\int_0^c ev_1(c,c_2,c')\pi(dc'|c)$$

$$v_{N,1}(c_1,c_2,c,2) = r_1(c_1,c_2) + \beta f(1|2)\int_0^c \phi(v_{N,1}(c_1,c_2,c',1), v_{I,1}(c_1,c_2,c',1))\pi(c'|c)dc' + \beta f(2|2)\int_0^c ev_1(c_1,c_2,c')\pi(dc'|c)$$

$$v_{I,1}(c_1,c_2,c,2) = r_1(c_1,c_2)+\beta f(1|2)\int_0^c \phi(v_{N,1}(c_1,c,c',1),v_{I,1}(c_1,c,c',1))\pi(c'|c)dc' + \beta f(2|2)\int_0^c ev_1(c_1,c,c')\pi(dc'|c). \quad (31)$$

where

$$ev_1(c_1,c_2,c)=P_2(c_1,c_2,c,2)v_{I,1}(c_1,c_2,c,2)+[1-P_2(c_1,c_2,c,2)]v_{N,1}(c_1,c_2,c,2). \quad (32)$$

Note that neither firm is allowed to invest out of turn, *i.e.* $P_2(c_1,c_2,c,1)=P_1(c_1,c_2,c,c,2)=0$.

In this example not all state variables are directional, since the right to move alternates back and forth between the two players, and thus we treat dimension $m\in\{1,2\}=X$ of the state space as non-directional, while the directional component is $d=(c_1,c_2,c)\in D$ as before. Despite this additional complexity, the partial order over D is the same as in the simultaneous move example above, and we can still solve every (c_1,c_2,c)-stage game. In particular, it can be shown that the eight functional equations (four of which are given by equation (31) above) at the each stage game $\mathcal{G}(\tau-1)$, can be solved directly given the solution at the previously calculated stage games $\mathcal{G}(\tau)$ and a deterministic equilibrium selection rule Γ. Thus, state recursion and RLS algorithms apply, allowing for a full solution of the alternating move pricing and investment game.

Theorem 7. (Solution method for the d-stage games in alternating move leapfrogging game). *When $\eta=0$ there exists an exact non-iterative algorithm that finds all MPE of every d-stage game in the Bertrand pricing and investment game with stochastically alternating moves. The number of possible MPE in any d-stage game is either 1 or 3.*

4.3. *Performance of the solution algorithms*

Theorems 6 and 7 ensure that the key assumptions required for the validity of the state recursion and RLS algorithms are satisfied, and thus these methods can be applied to both versions of the Bertrand pricing and investment game. RLS reveals that these games have a surprisingly rich set of MPE. Figure 8 displays the computed equilibrium expected profits of the two firms in both versions of the game at the apex (c_0,c_0,c_0). With only 5 points in the grid of the costs, there are around 200 million MPE in the simultaneous move version of the game (shown in the left panel of the figure), and with only 6 points in the grid of the costs, there are around 3 million MPE in the alternating move version with random alternation of the right to move.

Table 1 reports the CPU time required to compute all MPE of several specifications of the Bertrand pricing and investment game of different sizes using MacBook Pro computer with 2.7 GHz Intel Core i7 processor. Comparing the run times for the three simultaneous moves games, it is obvious that due to a sharply increasing number of times the state recursion (or partial state recursion) is invoked in the RLS loop the runtimes increase rapidly, at the same rate as the number of MPE it finds. Yet, comparing the runtimes for the largest game with simultaneous moves to that of the alternating moves, it becomes obvious that the RLS algorithm itself takes a negligible amount of time to loop through all feasible ESR strings compared to the time needed for state recursion.[13]

13. Further analysis of the Bertrand investment and pricing game, and a complete set of theoretical results from this model can be found in the companion paper Iskhakov *et al*. (2015).

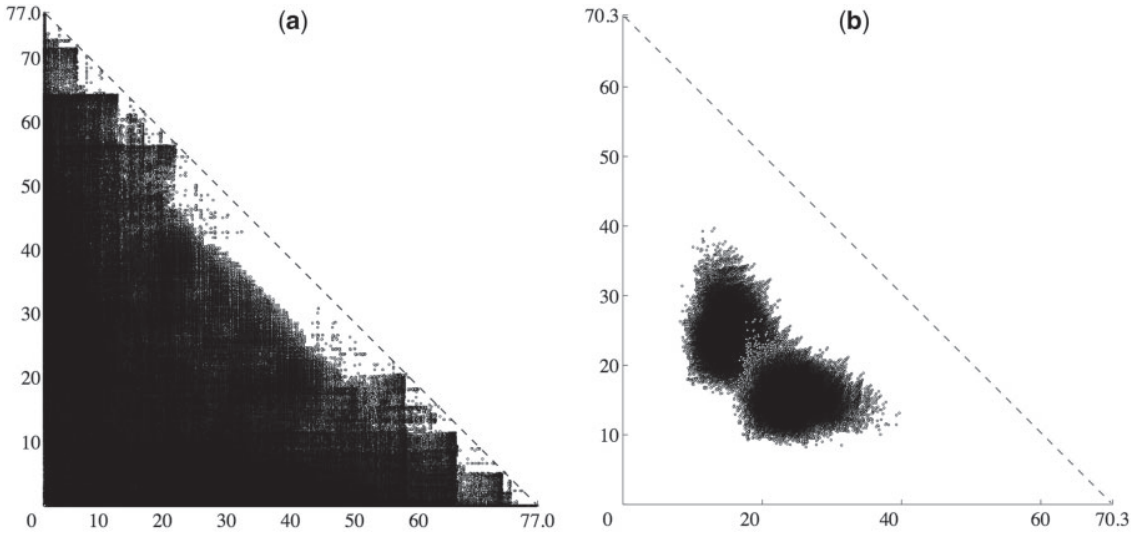

FIGURE 8

Typical sets of equilibrium outcomes in Bertrand pricing and investment game with simultaneous (a) and alternating (b) moves

Notes: Each point represents a pair of expected discounted value functions for firms 1 and 2. The vertices of the triangles are determined by the expected discounted monopoly profit under the same technological process. Panel (a) plots 164,295,079 MPE of the simultaneous move game with $n=5$. Panel (b) plots 3,138,026 MPE of the alternating move game with $n=6$ and randomly alternating right of move. Precise model parameters used in building the graphs are available upon request. See Iskhakov *et al*. (2015) for further details.

TABLE 1

Run times for full solution of the leapfrogging game

	Simultaneous moves			Alternating moves
Number of points in cost grid n	3	4	5	5
Upper bound on the number of ESS	4,782,969	3,948,865,611	$1.7445 \cdot 10^{26}$	$1.7445 \cdot 10^{26}$
Number of feasible ESS	127	46,707	192,736,405	1
RLS runtime	0.008 s	0.334 s	45 min	0.006 s

Notes: The upper bound for number of ESS is computed using $K=3$ constant base arithmetics for comparison.

5. DISCUSSION AND CONCLUSION

We introduced the concept of directionality in finite state Markovian dynamic games, defined the class of DDGs, and proposed two new solution algorithms for the games in this class. The state recursion algorithm finds a single MPE of a DDG for a specified equilibrium selection rule; the RLS algorithm finds all possible MPE of the game by efficiently searching over all feasible equilibrium selection rules. The run-time of the RLS algorithm is linear in the total number of MPE of the DDG, ensuring that negligible time is spent on enumerating all feasible equilibrium selection rules relative to the time spent to compute each of the corresponding equilibria.

The class of DDGs we defined in this article appears to be quite large and there are many examples of games of this type in the existing literature beyond the recent papers by Judd *et al*. (2012b) and Dubé, Hitsch and Chintagunta (2010) which we discussed in Section 2. The flexibility and range of application of DDGs is due partly to the fact that it is sufficient to identify just one directional component of the state space for the game to qualify as a DDG, and

also because our definition places no restrictions on the non-directional components of the state space X.

In our main theorem (Theorem 5), which lists the conditions under which RLS is guaranteed to find all MPE of DDG in a finite number of steps, we did assume the existence of a solution algorithm that can find *all* of the MPE of the component d-stage games that have reduced state spaces of the form $\{d \times X\}$ and reduced sets of strategies. State recursion and RLS may not be applicable to DDGs that have numerous or complicated non-directional components X, since this may result in stage games that are too complex, making it intractable to find all of their MPE. Yet, according to Corollary 5.1 even if there is no algorithm capable of finding all MPE of each d-stage game, RLS can still be applied to find many MPE of the DDG provided there is an algorithm that can find at least *some* of the MPE of the d-stage games.

We can also show that some dynamic games that appear to be non-directional at first glance can satisfy our definition of a DDG by appropriate redefinitions of the state space. For instance, Example 3 in Section 2 presented in the right panel of Figure 1 has a loop in the directional component of the state space, violating the no-loop condition to qualify as a DDG. However, we can add a second dimension to the state space and redefine the states so that the states connected by a loop have the same value on the newly defined directional dimension and differ only in the newly defined non-directional dimension. After this redefinition of the states, the game can be shown to be a DDG. Whether RLS is applicable is again determined by whether it is possible to find all MPE of the aggregated states (including all points that result in loops) simultaneously, as they form one of the elemental stage games of the equivalent game with the redefined state space.

The reason why the homotopy approach is not well suited for finding all MPE of finite-state DDGs is due to numerous bifurcations along the equilibrium correspondence that the homotopy algorithm tries to follow in an attempt to find all MPE of $\mathcal{G}$. Figure 9 illustrates the nature of the problem by graphing the equilibrium correspondence in the Bertrand pricing and investments game with alternating moves. This figure shows *all* MPE for *each* value of the typical homotopy parameter η, which in this example indexes the variance of idiosyncratic investment shocks as discussed in section 4. When η is sufficiently large there is a unique MPE of $\mathcal{G}$ as we can see from Figure 9. The homotopy algorithm tries to follow this initially unique equilibrium path as η is reduced in a series of steps towards 0. However, it is evident that this path does not lead to all MPE of $\mathcal{G}$ as $\eta \to 0$. Besides frequent bifurcations in the original path, we see that a multitude of new completely disjoint paths pop up elsewhere as η decreases.

However, since there are far fewer MPE in each of the stage games (at most five in the leapfrogging game example), the homotopy method may have a better chance of finding all (or at least many) of the MPE of the stage games. Therefore, the homotopy method might be far more effective when it is used together with state recursion and RLS algorithms to find MPE of the overall DDG $\mathcal{G}$.

Our Bertrand pricing and investment example leads us to conjecture that the multiplicity of MPE in general finite state dynamic games is due to the combinatoric explosion resulting from the freedom to choose different MPE at different stages of the game. RLS naturally accounts for all such combinations of equilibria selected in different stage games, and if the conjecture is correct, it can provide new insights into the bifurcations and complex structure of the equilibrium correspondence such as illustrated in Figure 9.

A final note is that RLS is likely to be subject to a curse of dimensionality that originates both from an exponential increase in the number of points in the directional component of the state space as the dimension of the problem rises, and also from a curse of dimensionality in how the total number of MPE may increase with the total number of points in the state space. Even though the run-time of the RLS is linear in the total number of MPE of $\mathcal{G}$, the number

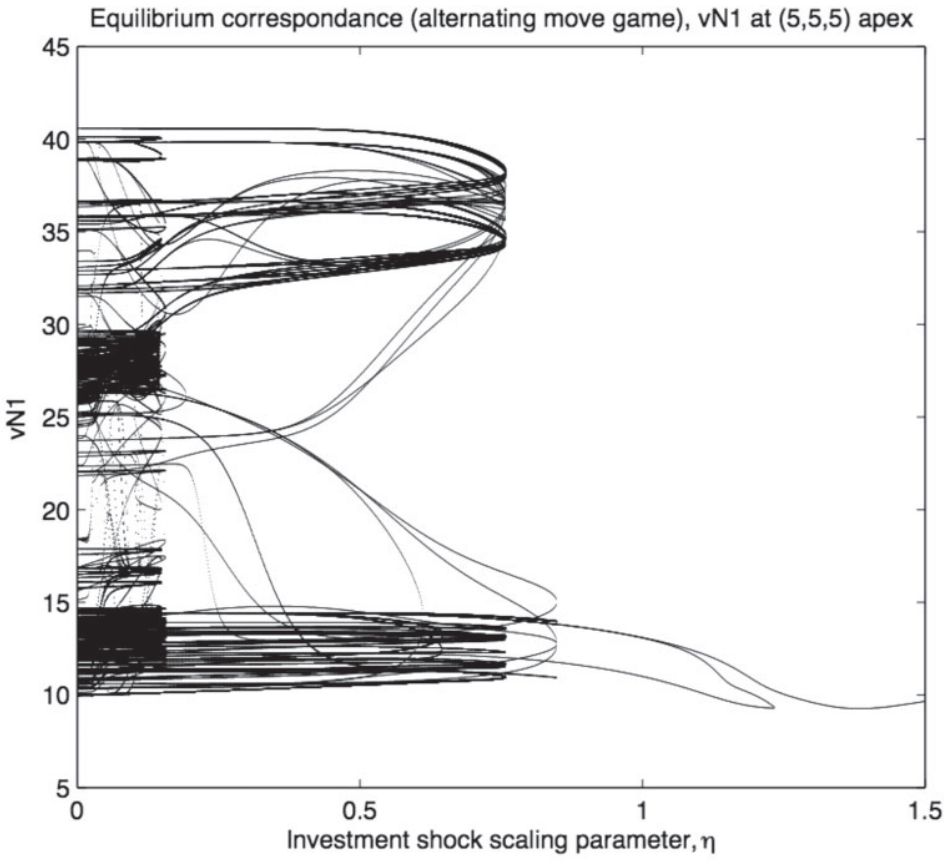

FIGURE 9

Equilibrium correspondence, alternating move game

Notes: All possible values of expected profit of firm 1 (resulting from multiple equilibria) along y-axes are plotted against values of cost shock η. Large values of η correspond to a single equilibrium, as η decreases, the correspondence is severely complicated by numerous bifurcation points and new emerging equilibria.

of MPE itself may increase exponentially as the number of points in the state space increases. This is evident in our results for the simultaneous move version of the leapfrogging game in Table 1. However, there are games where the MPE may be unique or for which the total number of MPE grow only polynomially fast as the number of points in the state space increases. In such cases, RLS may be a tractable algorithm for finding all MPE, and can run very quickly even for games with large numbers of states. In fact, we have been able to run RLS to find all MPE of alternating move games where the state space has several millions of points. Thus, whether or not the RLS algorithm is subject to a curse of dimensionality is largely dependent on whether or not the number of MPE of the game increases exponentially as the number of points in the state space increases.

APPENDIX

A. PROOFS

Lemma 1 (Partial order over directional component of the state space).

Proof. Recall that a (strict) partial order $\succ$ of the elements of a set D is a binary relation (*i.e.* a subset of $D \times D$) that is 1) irreflexive ($d \not\succ d$ for all $d \in D$), 2) asymmetric (if $d' \succ d$ then $d \not\succ d'$ for all $d', d \in D$) and 3) transitive (if $d_3 \succ d_2$ and $d_2 \succ d_1$, then $d_3 \succ d_1$ for all $d_1, d_2, d_3 \in D$). It is clear from equation (3) that $\succ_\sigma$ is irreflexive, since $d \succ_\sigma d$ would require

that $\rho(d|d,x,\sigma)$ be simultaneously equal to 0 and greater than 0. For similar reasons $\succ_\sigma$ is asymmetric, since equation (3) cannot hold simultaneously for the pairs (d,d') and (d',d). Then suppose that $d_3 \succ_\sigma d_2$ and $d_2 \succ_\sigma d_1$. This means that there is a positive probability of going from d_1 to d_2 (but zero probability of going from d_2 back to d_1) and similarly there is positive probability of going from d_2 to d_3 (but zero probability of going from d_3 back to d_2). Via a probability chain formula (the *Chapman–Kolmogorov equation*) it follows that there is a positive probability of going from d_1 to d_3. It remains to be shown that there must be a zero probability of a reverse transition from d_3 to d_1. Suppose not. Then the chain formula implies that the probability of a transition from d_2 back to d_1 via d_3 is positive, contradicting the hypothesis that $d_2 \succ_\sigma d_1$. ‖

Theorem 1 (Strategy independent partial order).

Proof. We first demonstrate that $\succ_\mathcal{G}$ is irreflexive, asymmetric and transitive, and thus a partial order of D. For any $d \in D$ it cannot be the case that $d \succ_\mathcal{G} d$ because by the definition of $\succ_\mathcal{G}$ it would have to be the case that $d \succ_\sigma d$ for some $\sigma \in \Sigma(\mathcal{G})$. However, each strategy-specific partial order $\succ_\sigma$ is irreflexive by Lemma 1, so this is a contradiction. To establish asymmetry of the partial order $\succ_\mathcal{G}$ suppose to the contrary that there is a pair of points $d,d' \in D$ such that $d' \succ_\mathcal{G} d$ and $d \succ_\mathcal{G} d'$. Then since each partial order $\succ_\sigma$ is asymmetric by Lemma 1, it must be the case that there exist two feasible strategies σ and σ' in $\Sigma(\mathcal{G})$ such that $d' \succ_\sigma d$ and $d' \succ_{\sigma'} d$. However, this violates the consistency condition (5) in the definition of a DDG, Definition 4. The transitivity of $\succ_\mathcal{G}$ follows from the fact that this binary relation is the transitive closure of the union of the transitive binary relations $\succ_\sigma$.

It follows that $\succ_\mathcal{G}$ is a partial order that contains each strategy-specific partial order $\succ_\sigma$ for $\sigma \in \Sigma(\mathcal{G})$, and hence it is a common refinement of the set of a partial orders induced by all feasible strategies of $\mathcal{G}$, $\{\succ_\sigma | \sigma \in \Sigma(\mathcal{G})\}$. To show that it is the coarsest common refinement, suppose to the contrary that there is another partial order $\succ$ that is a strict subset of $\succ_\mathcal{G}$. Let (d',d) be a ordered pair that is in the order $\succ_\mathcal{G}$ but not in $\succ$. Then there are two possibilities. Either $d' \succ_\sigma d$ for some $\sigma \in \Sigma(\mathcal{G})$, or (d',d) is a point added to $\cup_{\sigma \in \Sigma(\mathcal{G})} \succ_\sigma$ to ensure it is transitive. In the latter case, deleting this point implies that the relation $\succ$ is no longer transitive, so it cannot be a common refinement of the transitive $\{\succ_\sigma | \sigma \in \Sigma(\mathcal{G})\}$. The other possibility is that $d' \succ_\sigma d$ for some $\sigma \in \Sigma(\mathcal{G})$. However, removing this point implies that $\succ$ is no longer a refinement of $\succ_\sigma$ and thus it cannot be a common refinement of $\{\succ_\sigma | \sigma \in \Sigma(\mathcal{G})\}$. ‖

Lemma 2 (DAG recursion).

Proof. The sequence starts at the DAG $D(\mathcal{G})$ which is non-empty and has a finite number of vertices, as game $\mathcal{G}$ is a finite state DDG. Vertices are not added by the recursion (9), so it follows at each step $j < \mathcal{T}$ $D_j(\mathcal{G})$ is a DAG with finite number of vertices. Thus, as long as $D_j(\mathcal{G})$ is non-empty and is a DAG, the $\mathcal{N}$ operator never returns the empty set, reducing the number of vertices remaining in $D_j(\mathcal{G})$ as j increases. This implies that the recursion must eventually terminate at some value $\mathcal{T}$ for which we have $D_\mathcal{T}(\mathcal{G}) = \mathcal{N}(D_\mathcal{T}(\mathcal{G}))$. ‖

Corollary 2.1 (Test whether an arbitrary directed graph is a DAG). *In an arbitrary directed graph $\mathcal{D}$ with a finite number of vertices, let recursion (9) terminate either in the case when the vertices of $\mathcal{D}$ are exhausted, or when $\mathcal{N}$ operator returns the empty set. Let $\mathcal{D}_{\mathcal{T}+1}$ denote the final element of the sequence $\{\mathcal{D}_0, \mathcal{D}_1, \ldots, \mathcal{D}_{\mathcal{T}+1}\}$. Then $\mathcal{D}$ is a DAG if and only if $\mathcal{D}_{\mathcal{T}+1} = \varnothing$.*

Proof. Necessity follow from the proof of Lemma 2. To show sufficiency, imagine the contrary that $\mathcal{D}_\mathcal{T} \neq \varnothing$ and yet $\mathcal{D}$ is DAG. If $\mathcal{T}+1$ is a terminal step, it must hold that every vertex in $\mathcal{D}_{\mathcal{T}+1}$ has a descendant. Because the number of vertices in $\mathcal{D}_{\mathcal{T}+1}$ is finite, repeatedly following the link to a descendant would result in a loop in $\mathcal{D}_{\mathcal{T}+1}$, leading to a contradiction. ‖

Lemma 3 (First-stage τ-subgame).

Proof. From equation (12) we see that $\Omega_1 = S$. Therefore $\mathcal{G}_1$ has the same state space as $\mathcal{G}$ and is identical in all other respects to $\mathcal{G}$. ‖

Theorem 2 (Decomposition of the stage game).

Proof. The proof follows from the independence of the substages $d_{i,\tau} \in D_\tau$, $i = 1, \ldots, n_\tau$. The $d_{i,\tau}$-stage games $\mathcal{SG}_\tau(d_{i,\tau})$ comprising the τ-stage game are payoff-independent of each other, *i.e.* the players' payoffs in $\mathcal{SG}_\tau(d_{i,\tau})$ are unaffected by the choice of strategy in any other $d_{j,\tau}$-stage game $\mathcal{SG}_\tau(d_{j,\tau})$, $i \neq j$ in the same stage τ of $\mathcal{G}$. Therefore, putting together the the disjoint state spaces of $d_{i,\tau}$-stage games does not alter the equilibria $\mathcal{E}(\mathcal{SG}_\tau(d_{i,\tau}))$ of these games, and so the union of these equilibria constitute an equilibrium of the τ-stage game $\mathcal{SG}_\tau$. ‖

Lemma 4 ($\mathcal{T}$-stage game).

Proof. This result follows easily since at the terminal stage of the DDG $\mathcal{T}$ there is no following stage $\mathcal{T}+1$, and thus according to the Definition 11 the continuation strategies are simply given by the feasible Markovian strategies. Thus, the strategy space of the stage games $\mathcal{SG}_{\mathcal{T}}(d)$ and $\mathcal{SG}_{\mathcal{T}}$ is identical to the strategy space of the subgames $\mathcal{G}_{\mathcal{T}}(d)$ and $\mathcal{G}_{\mathcal{T}}$. The result follows from the fact that these games can only differ in strategy spaces by Definition 12. ‖

Theorem 3 (Subgame perfection).

Proof. Suppose $(\sigma, V)=e(\mathcal{SG}_\tau(d))\in\mathcal{E}(\mathcal{SG}_\tau(d))$. We want to show that it is also an element of $\mathcal{E}(\mathcal{G}_\tau(d))$. We prove the result by mathematical induction. The result holds trivially at the last stage $\mathcal{T}$ by virture of Lemma 4. This implies that $\mathcal{SG}_{\mathcal{T}}(d)=\mathcal{G}_{\mathcal{T}}(d)$ for $d\in D_{\mathcal{T}}$ which implies that $\mathcal{E}(\mathcal{SG}_{\mathcal{T}}(d))=\mathcal{E}(\mathcal{G}_{\mathcal{T}}(d))$ for $d\in D_{\mathcal{T}}$. Now suppose the result holds for all d-subgames for all stages $\tau'=\tau+1,\ldots,\mathcal{T}$. We now show that it holds for all d-subgames at stage τ as well. By definition, $e(\mathcal{SG}_\tau(d))$ is an MPE of the stage game $\mathcal{SG}_\tau(d)$ in the restricted class of continuation strategies. However, by definition, a continuation strategy is an MPE strategy in the stage τ_1 subgame $\mathcal{G}_{\tau+1}$. It follows that $e(\mathcal{SG}_\tau(d))$ is an MPE strategy on the set $(d\times X)$ for $d\in D_\tau$ and also on the stage $\tau+1$ subgame $\mathcal{G}_{\tau+1}$, so it must be an MPE for the full d-subgame $\mathcal{G}_\tau(d)$, since if it was not it would have to fail to be an MPE at some point s either for $s\in(d\times X)$ or $s\in\Omega_{\tau+1}$, where $\Omega_{\tau+1}$ is the state space for the stage $\tau+1$ subgame, given in equation (12) of Definition 9. In either case there would be a contradiction, since the property that $e(\mathcal{SG}_\tau(d))$ is a continuation strategy implies that it must be an MPE at each $s\in\Omega_{\tau+1}$, and the fact that it is also an MPE for the stage game $\mathcal{SG}_\tau(d)$ implies that it is also must be an MPE strategy for each $s\in(d\times X)$. Thus, $e(\mathcal{SG}_\tau(d))$ is an MPE strategy at each point $s\in\Omega_\tau(d)$. Since this is the state space for the d-subgame $\mathcal{G}_\tau(d)$, it follows that $e(\mathcal{SG}_\tau(d))$ must be an MPE of $\mathcal{G}_\tau(d)$.

Conversely, suppose that $e(\mathcal{G}_\tau(d))$ is an MPE strategy of the d-subgame $\mathcal{G}_\tau(d)$. We can express $e(\mathcal{G}_\tau(d))$ as a continuation strategy as follows:

$$e(\mathcal{G}_\tau(d))(s)=\begin{cases} e(\mathcal{G}_\tau(d))(s) & \text{if } s\in(d\times X) \text{ and } d\in D_\tau \\ e(\mathcal{G}_{\tau+1})(s) & \text{otherwise.}\end{cases} \tag{33}$$

This follows from the general definition of MPE in equation (1) of Definition 1, since the Bellman equation must hold at very point in the state space, and the state space for $\mathcal{G}_\tau(d)$ includes $\Omega_{\tau+1}$, so $e(\mathcal{G}_\tau(d))$ must be an MPE for $s\in\Omega_{\tau+1}$ which implies that $e(\mathcal{G}_\tau(d))=e(\mathcal{G}_{\tau+1})$ for a particular equilibrium selection from the stage $\tau+1$ subgame $\mathcal{G}_{\tau+1}$. Thus, it follows that $e(\mathcal{G}_\tau(d))$ is an MPE in the restricted class of continuation strategies for the stage game $\mathcal{SG}_\tau(d)$, and thus $e(\mathcal{G}_\tau(d))\in\mathcal{E}(\mathcal{SG}_\tau(d))$. ‖

Theorem 4 (Convergence of State Recursion).

Proof. The state recursion algorithm given in definition 13 recursively computes an MPE of the d-stage games $\mathcal{SG}_\tau(d)$ on every stage τ. By Theorem 2 the MPE of the τ-stage game $\mathcal{SG}_\tau$ can then be constructed. By Theorem 3 or Lemma 4, it is also an MPE of τ-subgame $\mathcal{G}_\tau$, so the continuation strategies for the next iteration stage $\tau-1$ can be constructed using Definition 11, and the algorithm continues to stage $\tau-1$. Once the first stage $\tau=1$ is reached, by Lemma 3 an MPE for the whole DDG $\mathcal{G}$ is found. Iterations of the algorithm are indexed by stage intex τ, thus for a finite state DDG $\mathcal{G}$ which by Lemma 2 has finite number of stages $\mathcal{T}$, state recursion terminates in a finite number of steps. ‖

Lemma 5 (Feasibility of ESS).

Proof. Necessity follows from Definition 15. We show sufficiency by induction on the digits of γ. Consider first the end game stage $\mathcal{T}$ which corresponds to the highest (left-most) digit of γ. The first sub-vector of γ satisfies element-wise inequality $\gamma_{\mathcal{T}}<ne_{\mathcal{T}}$, and assuming that the MPE of every d-stage game at stage $\mathcal{T}$ are indexed starting from zero, if follows that $\gamma_{\mathcal{T}}$ indeed points to an existing d-stage equilibrium. If γ_τ indexes the existing equilibria on stage τ, $ne_{\tau-1}(\gamma_{>\tau-1})$ is well defined as the vector containing the number of the MPE equilibria in every d-stage game on stage $\tau-1$. Again, assuming that the indices of the d-stage game equilibria start from zero, it follows that $\gamma_{\tau-1}$ also points to an existing stage game MPE. By induction, every sub-vector γ_τ indexes an existing d-stage game MPE, so by Definition 15 γ is a feasible ESS. ‖

Lemma 6 (Successor function and feasibility).

Proof. If $\mathcal{S}(\gamma)=(-1,\ldots,-1)$, then there can be no feasible $\gamma'\in Z_+^N$ satisfying $\iota(\gamma')>\iota(\gamma)$ because the successor is the result of incrementing γ by the smallest possible non-zero value, 1. It follows that $\mathcal{J}(\gamma)=(-1,\ldots,-1)$ and so $\mathcal{J}(\gamma)=\mathcal{S}(\gamma)$ in this case. Otherwise, if $\mathcal{S}(\gamma)\neq(-1,\ldots,-1)$ then we have $\iota(\mathcal{S}(\gamma))=\iota(\gamma)+1$, so if $\mathcal{S}(\gamma)$ is feasible, it must be the smallest ESS after γ, and hence $\mathcal{J}(\gamma)=\mathcal{S}(\gamma)$. But if $\mathcal{S}(\gamma)\neq(-1,\ldots,-1)$ it must be feasible by the properties of the successor operator in variable base arithmetic. The long addition process insures that we have

for each $i=1,\ldots,n_\tau$ and $\tau=1,\ldots,\mathcal{T}$, $\gamma_{i,\tau}<ne_{i,\tau}(\gamma_{>\tau})$, but by Lemma 5 it follows that $\mathcal{S}(\gamma)$ must be a feasible ESS. ‖

Lemma 7. (RLS completeness). *Let $\gamma\in\Lambda$ be a feasible ESS returned by the RLS algorithm, and let e_γ be the MPE of $\mathcal{G}$ induced by γ. Let $\mathcal{E}_\tau(\mathcal{G}|e_\gamma)$ denote the set of all MPE of $\mathcal{G}$ that revert to the MPE e_γ at stage τ, i.e. the players use e_γ to define a continuation strategy for stages $\tau+1,\ldots,\mathcal{T}$. Then if $e\in\mathcal{E}_\tau(\mathcal{G}|e_\gamma)$, there exists a $\gamma'\in\Lambda$ such that $e=e_{\gamma'}$.*

Proof. We prove the result by induction on τ starting from $\tau=1$. Consider a feasible ESS $\gamma'=(\gamma_\mathcal{T},\ldots,\gamma_2,\gamma_1)$, where $\gamma_1=(0,\ldots,0)\in Z_+^{n_1}$ is consistent with feasibility due the assumption we made about existence of at least one MPE. Then $e_{\gamma'}$ is an MPE where the players use the MPE strategy e_γ for all stages after stage $\tau=1$ but play the first equilibrium in every d-stage game in stage $\tau=1$. Let $\Lambda_1(\gamma)$ denote the set of all ESS γ' such that $\gamma'_\tau=\gamma_\tau$ for $\tau>1$. Suppose that the RLS algorithm is incomplete, *i.e.* suppose there is an $e\in\mathcal{E}_1(\mathcal{G}|e_\gamma)$ for which there is no $\gamma'\in\Lambda_1(\gamma)$ such that $e_{\gamma'}=e$. Let $e(\mathcal{SG}_1|e_\gamma)$ be the MPE on the stage games in stage $\tau=1$ of $\mathcal{G}$ induced by this e (which reverts to the continuation strategy e_γ after stage 1). Then this is a contradiction of the assumption that there is an algorithm that can compute all MPE of every stage game of $\mathcal{G}$, including all stage games at stage $\tau=1$.

The induction step is proved similarly. That is, suppose we have shown that the RLS algorithm finds all MPE of $\mathcal{G}$ in the set of continuation strategies that revert to the MPE e_γ after any stage $\tau-1,\ldots,1$. We want to show that this is also true for stage τ, *i.e.* RLS also finds all MPE of $\mathcal{G}$ in the class of continuation strategies that revert to the MPE e_γ after stage τ. Let $\Lambda_\tau(\gamma)$ denote the set of all ESSs γ' such that $\gamma'_{\tau'}=\gamma_{\tau'}$ for $\tau'>\tau$. Suppose the RLS algorithm is incomplete, *i.e.* suppose that there is an MPE $e\in\mathcal{E}_\tau(\mathcal{G}|e_\gamma)$ such that there is no $\gamma'\in\Lambda_\tau(\gamma)$ such that $e=e_{\gamma'}$. Let $e(\mathcal{SG}_\tau|e_\gamma)$ be the MPE of the stage games in stage τ induced by this e (which reverts to the continuation strategy e_γ after stage τ). Then this contradicts our assumption that RLS uses an algorithm that finds all MPE of every stage game of $\mathcal{G}$. ‖

Theorem 5 (Strong convergence of RLS algorithm).

Proof. The RLS algorithm must terminate since there is a finite number of possible ESSs, and the RLS algorithm operates by repeatedly finding the successor ESR starting from the initial feasible ESR $\gamma_0=(0,\ldots,0)$, which is always feasible by our assumption that $\mathcal{G}$ has at least one MPE. So upon termination, the RLS algorithm has returned a set Λ containing a finite number of feasible ESSs, $\Lambda=\{\gamma_0,\ldots,\gamma_{J-1}\}$. The feasibility of each ESR $\gamma_j\in\Lambda$ means that γ_j corresponds to an ESR that selects an MPE for each stage game of $\mathcal{G}$. By Theorem 3 it follows that each $\gamma_j\in\Lambda$ corresponds to an MPE of $\mathcal{G}$, e_{γ_j}. So RLS has generated a subset of the set of all MPE of $\mathcal{G}$, $\mathcal{E}(\mathcal{G})$.

Is there any MPE of $\mathcal{G}$ that the RLS algorithm could have missed? Let $e\in\mathcal{E}(\mathcal{G})$ be an MPE that RLS missed. That is, there is no $\gamma\in\Lambda$ such that $e=e_\gamma$. We now show the the existence of such an MPE leads to a contradiction. Let $e(\mathcal{SG}_\mathcal{T})$ be the MPE of the end game of $\mathcal{G}$ that is induced by the MPE e that RLS did not find. But this contradicts the assumption that RLS has access to an algorithm that can find all MPE of every stage game of the DDG $\mathcal{G}$, including the end game since the properties of the successor operator $\mathcal{S}$ discussed in the previous subsection guarantee that that RLS algorithm will evaluate all feasible ESSs for the end game of $\mathcal{G}$ and thus, there can be no MPE of any end game that would be missed by the RLS algorithm. If follows that there is some $\gamma'\in\Sigma$ such that $e(\mathcal{SG}_\mathcal{T})=e_{\gamma'}(\mathcal{SG}_\mathcal{T})$, *i.e.* the MPE of the end game $\mathcal{SG}_\mathcal{T}$ induced by the strategy e equals the MPE on the same end game induced by the ESR $\gamma'\in\Lambda$.

Now suppose there is some $e\in\mathcal{E}(\mathcal{G})$ that was missed by the RLS algorithm. We have just shown that there is an ESR $\gamma\in\Lambda$ such that $e(\mathcal{SG}_\mathcal{T})=e_{\gamma'}(\mathcal{SG}_\mathcal{T})$, *i.e.* there is at least one ESR $\gamma\in\Lambda$ whose implied MPE e_γ coincides with e on the end game states. But if this is true, then Lemma 7 implies that $\mathcal{E}_\mathcal{T}(\mathcal{G}|\gamma)$, the set of all MPE of $\mathcal{G}$ that revert to the MPE $e_{\gamma'}$ on the end game, $\mathcal{T}$, is complete, *i.e.* given any $e\in\mathcal{E}_\mathcal{T}(\mathcal{G}|\gamma)$, there is a $\gamma'\in\Lambda$ such that $e=e_{\gamma'}$. But this contradicts the hypothesis that e was missed by the RLS algorithm, *i.e.* that there is no $\gamma'\in\Lambda$ such that $e=e_{\gamma'}$. ‖

Corollary 5.1 (Weak convergence of RLS algorithm).

Proof. The proof is analogous to the proof of Theorem 5 where the condition of finding all MPE in every stage game is relaxed to yield a relaxed conclusion of finding some MPE of the whole game instead of all MPE. It remains to be shown that $J\geq 1$. This follows from the assumption that the d-stage game solution algorithm can find at least one MPE, which can then be combined according to the state recursion algorithm into at least one MPE of the whole game. ‖

Theorem 6 (Solution method for the d-stage games in simultaneous move leapfrogging game).

Proof. The proof is constructive and is based on the algorithm we developed to find all MPE of all d-stage games of the Bertrand pricing and investment game with simultaneous moves when $\eta=0$. The stages of the game defined with the DAG recursion (9) can be grouped into the *layers* of the state pyramid as shown in Figure 5 according to the values of the state-of-the-art cost c. Each layer contains three types of stages: the corner stages $\tau=\{\mathcal{T},\mathcal{T}-3,\mathcal{T}-6,\ldots\}$ composed of single point

$\{d=(c_1,c_2,c)\colon c_1=c_2=c\}$; the edge stages $\tau=\{\mathcal{T}-1,\mathcal{T}-4,\ldots\}$ composed of points $\{d=(c_1,c_2,c)\colon c_1=c\neq c_2$ or $c_2=c\neq c_1\}$; and the interior stages $\tau=\{\mathcal{T}-2,\mathcal{T}-5,\ldots\}$ composed of the points $\{d=(c_1,c_2,c)\colon c_1>c$ and $c_2>c\}$.

Consider first the bottom layer of the game where $c=0$ and no further improvement of the state-of-the-art technology is possible, $c'=0$ and $\pi(c|c)=1$. It is easy to see that at the end game $\tau=\mathcal{T}$ in the "corner state" $(c_1,c_2,c)=(0,0,0)$ there is no gain to either firm from investing, and so the only possible MPE in this state is for neither firm to invest. It is also easy to see that there is only a single no-investment MPE in the $(\mathcal{T}-1)$-stage at the "edge states" $(c_1,c_2,c)=(c_1,0,0)$ and $(c_1,c_2,c)=(0,c_2,0)$ as well again because there is no gain from investing for either firm.

Consider next the $(\mathcal{T}-2)$-stage game defined on the interior states $\{(c_1,c_2,0)\colon c_1,c_2>0\}$. Focusing on firm 1, it follows from the discussion above that

$$\begin{aligned}\phi\big(v_{N,1}(0,0,0),v_{I,1}(0,0,0)\big) &= 0,\\ \phi\big(v_{N,1}(c_1,0,0),v_{I,1}(c_1,0,0)\big) &= 0,\\ \phi\big(v_{N,1}(0,c_2,0),v_{I,1}(0,c_2,0)\big) &= c_2/(1-\beta),\end{aligned} \tag{34}$$

where $\phi(\cdot,\cdot)$ is the max operator when $\eta=0$. The Bellman equations (30) simplify to

$$\begin{aligned}v_{N,1}(c_1,c_2,0) &= r_1(c_1,c_2)+\beta\big(1-P_2(c_1,c_2,0)\big)\max\{v_{N,1}(c_1,c_2,0),v_{I,1}(c_1,c_2,0)\},\\ v_{I,1}(c_1,c_2,0) &= r_1(c_1,c_2)-K(0)+\beta\big(1-P_2(c_1,c_2,0)\big)c_2/(1-\beta).\end{aligned} \tag{35}$$

Consider first the pure-strategy MPE of this stage game. If firm 2 surely invests, $P_2(c_1,c_2,0)=1$, then the second terms in equation (35) disappear, leading to $v_{N,1}(c_1,c_2,0)>v_{I,1}(c_1,c_2,0)$. Thus, firm 1 does not invest, $P_1(c_1,c_2,0)=0$, forming the first pure-strategy "anti-coordination" equilibrium. Conversely, if firm 2 surely does not invest, $P_2(c_1,c_2,0)=0$, there are two cases to consider. Assuming in the first case that $v_{N,1}(c_1,c_2,0)<v_{I,1}(c_1,c_2,0)$ so that the max operator in equation (35) evaluates to $v_{I,1}(c_1,c_2,0)$, and thus $P_1(c_1,c_2,0)=1$, after simplifying the equations and verifying the assumption, we derive a condition

$$K(0)<\frac{\beta\big(c_2-r_1(c_1,c_2)\big)}{1-\beta}=\frac{\beta\min\{c_1,c_2\}}{1-\beta}. \tag{36}$$

In other words, for small enough investment cost, firm 1 surely invests when $P_2(c_1,c_2,0)=0$, forming another "anti-coordination" equilibrium in pure strategies. It is easy to show that the second case when $v_{N,1}(c_1,c_2,0)>v_{I,1}(c_1,c_2,0)$, and thus $P_1(c_1,c_2,0)=0$, holds when the investment cost is prohibitively high in the sense of equation (36) with the flipped inequality sign. In this case, firm 1 does not invest with probability one independently of $P_2(c_1,c_2,0)$.

Consider now the mixed-strategy MPE of this stage game, assuming that equation (36) holds. For some level of $P_2(c_1,c_2,0)$ firm 1 can be indifferent between investing and not investing, in which case any $P_1(c_1,c_2,0)\in[0,1]$ would constitute its best-response investment probability. In this case, equation (35) give rise to the following quadratic equation in $P_2=P_2(c_1,c_2,0)$

$$\frac{\beta^2c_2}{1-\beta}(1-P_2)^2+\beta\big(r_1(c_1,c_2)-K(0)-\frac{\beta c_2}{1-\beta}\big)(1-P_2)+K=0. \tag{37}$$

Conditional on that they belong to the unit interval, the roots of equation (37) combined with the roots of similar equation for firm 2 would define the mixed-strategy equilibrium of the stage game. Note however, that for $P_2=1$ the left-hand side of equation (37) is equal to $K>0$, while for $P_2=0$ it is equal to $(1-\beta)\left(K(0)-\frac{\beta\left(c_2-r_1(c_1,c_2)\right)}{1-\beta}\right)<0$, and thus there can only be a single root in the unit interval.

Similar arguments repeated for firm 2 lead to the conclusion that $(c_1,c_2,0)$-stage games can have either one no-investment MPE in pure strategies when investment cost is too high, or two pure-strategy "anti-coordination" and one mixed-strategy MPE when equation (36) holds. Both of these situations are illustrated in two left-most panels of Figure 6.

Consider next the higher layers of the leapfrogging game state space where $c>0$, again focusing on firm 1. It is helpful to re-write the Bellman equations (30) as

$$\begin{aligned}v_{N,1}(c_1,c_2,c) &= r_1(c_1,c_2)+\beta\Big[P_2(c_1,c_2,c)H_1(c_1,c,c)+\big(1-P_2(c_1,c_2,c)\big)H_1(c_1,c_2,c)\Big]\\ v_{I,1}(c_1,c_2,c) &= r_1(c_1,c_2)-K(c)+\beta\Big[P_2(c_1,c_2,c)H_1(c,c,c)+\big(1-P_2(c_1,c_2,c)\big)H_1(c,c_2,c)\Big]\end{aligned} \tag{38}$$

where the function $H_1(c_1,c_2,c)$ is given by

$$\begin{aligned}H_1(c_1,c_2,c) &= h_1(c_1,c_2,c)+\pi(c|c)\max\big\{v_{N,1}(c_1,c_2,c),v_{I,1}(c_1,c_2,c)\big\},\\ h_1(c_1,c_2,c) &= (1-\pi(c|c))\int_0^c\max\big\{v_{N,1}(c_1,c_2,c'),v_{I,1}(c_1,c_2,c')\big\}f(c'|c)dc'.\end{aligned}$$

Recall that $\pi(c|c)$ is the probability for a cost-reducing innovation not to occur, and $f(c'|c)$ is the conditional density of the new (lower) state-of-the-art marginal cost of production conditional on an innovation having occurred. Note that

function $h_1(c_1,c_2,c)$ only depends on the entities corresponding to the lower layers and thus subsequent stages already solved in the state recursion algorithm. It can therefore be treated as fully known, given an equilibrium selection rule Γ to select a particular equilibrium in case there may be multiple MPE on some of these subsequent stages.

Following the analysis of the end game $\tau=\mathcal{T}$, it is not hard to show that at the corners (c,c,c) there is only a single no investment MPE where neither firm has an incentive to invest. Specifically, the Bellman equation (38) implies that $v_{I,1}(c,c,c)=v_{N,1}(c,c,c)-K(c)$, so it can never be optimal for firm 1 to invest in this corner state if $K(c)>0$. A similar argument applies to firm 2.

The edges (c_1,c,c) and (c,c_2,c) of the higher layers where $c>0$ can be solved similarly to the bottom layer, and it is not hard to show that corresponding $\tau=\{\mathcal{T}-4,\mathcal{T}-7,\ldots\}$-stage games have single pure-strategy MPE. Consider first points (c_1,c,c), $c_1>c>0$, where because firm 2 has already attained the state-of-the-art cost c, the Bellman equations (38) simplify to

$$\begin{aligned} v_{N,1}(c_1,c,c) &= \beta\Big[h_1(c_1,c,c)+\pi(c|c)\max\big\{v_{N,1}(c_1,c,c),v_{I,1}(c_1,c,c)\big\}\Big], \\ v_{I,1}(c_1,c,c) &= -K(c)+\beta H_1(c,c,c). \end{aligned} \tag{39}$$

In other words, the value functions of firm 1 do not depend on the investment probability of firm 2. Moreover, $v_{I,1}(c_1,c,c)$ only depends on the known entities, and both $v_{N,1}(c_1,c,c)$ and $v_{I,1}(c_1,c,c)$ can be computed directly using the following conditional expression for the former

$$v_{N,1}(c_1,c,c)=\begin{cases}\beta\Big(h_1(c_1,c,c)+\pi(c|c)v_{I,1}(c_1,c,c)\Big), & \text{if } v_{N,1}(c_1,c,c)\le v_{I,1}(c_1,c,c),\\ \beta h_1(c_1,c,c)/\big(1-\beta\pi(c|c)\big), & \text{if } v_{N,1}(c_1,c,c)>v_{I,1}(c_1,c,c).\end{cases} \tag{40}$$

Depending on whether the first or the second case of equation (40) realizes, firm 1 invests or does not invest with certainty. In the same points (c_1,c,c), $c_1>c>0$, firm 2 has no gains from investment because it has already reached the state-of-the-art cost c. Similar to the corner, it can be shown that $v_{I,2}(c_1,c,c)=v_{N,2}(c_1,c,c)-K(c)<v_{N,2}(c_1,c,c)$, and therefore firm 2 does not invest at these states. Combined with the identical arguments regarding the edge (c,c_2,c), $c_2>c>0$, this shows that similar to the edges at the bottom layer, the stage games containing the edges of the higher layers can only have one MPE, namely in pure strategies where one firm does not invest for sure, and the other firm may or may not invest.

Now turn to the interior points of the higher layers (c_1,c_2,c) where $c_1>c>0$ and $c_2>c>0$. These points form the state space of the $\tau=\{\mathcal{T}-2,\mathcal{T}-5,\ldots\}$-stage games. Unlike in the case $c=0$, for $c>0$ there can be 1, 3, or 5 MPE in these stage games, and it is productive to take the more general approach of characterizing equilibria as the intersection points of the firms' best-response correspondences, instead of searching for pure- and mixed-strategy equilibria independently.

Let $\Upsilon\{\cdot\le\cdot\}$ denote an "indicator correspondence" which takes the values 0 and 1 similarly to the indicator function, and takes all values on the interval $[0,1]$ when the inequality in the curly brackets is satisfied as equality. The best response of firm 1 to the investment choice probability $P_2=P_2(c_1,c_2,c)$ of firm 2 is then given by $\Upsilon\{v_{N,1}(c_1,c_2,c,P_2)\le v_{I,1}(c_1,c_2,c,P_2)\}$, where the last argument in the value functions emphasizes the fact that they depend on the investment probability of firm 2 (here treated as a free variable). That is when the value of investing is higher than the value of not investing, firm 1 invests with probability 1, and when the value of investing is lower than the value of not investing, firm 1 invests with probability 0. Yet, when firm 1 is indifferent between investing and not investing, any investment probability constitutes the best response to the value P_2 leading to this indifference. The points of intersection of $\Upsilon\{v_{N,1}(c_1,c_2,c,P_2)\le v_{I,1}(c_1,c_2,c,P_2)\}$ and $\Upsilon\{v_{N,2}(c_1,c_2,c,P_1)\le v_{I,2}(c_1,c_2,c,P_1)\}$ on the unit square on the plane (P_1,P_2) define the values of investment probabilities of the two firms that constitute both pure- and mixed-strategy MPE of the (c_1,c_2,c)-stage game.

Consider firm 1, and note that equation (38) implies that the value of investing is linear in P_2

$$v_{I,1}(c_1,c_2,c,P_2)=\Big[r_1(c_1,c_2)-K(c)+\beta H_1(c,c_2,c)\Big]+\beta\Big[H_1(c,c,c)-H_1(c,c_2,c)\Big]P_2, \tag{41}$$

where the coefficients only depend on the entities already computed for the consecutive stages in the state recursion algorithm. To find all P_2 where $\Upsilon\{v_{N,1}(c_1,c_2,c,P_2)\le v_{I,1}(c_1,c_2,c,P_2)\}=1$ simplify equation (38) under the assumption that the inequality holds. This gives

$$v_{N,1}(c_1,c_2,c,P_2)=r_1(c_1,c_2)+\beta\Big[P_2H_1(c_1,c,c)+\big(1-P_2\big)\big[h_1(c_1,c_2,c)+\pi(c|c)v_{I,1}(c_1,c_2,c)\big]\Big], \tag{42}$$

where $v_{I,1}(c_1,c_2,c)$ is given by equation (41). The combination of equations (41) and (42) leads to the quadratic inequality in the best-response correspondence for firm 1 which takes the form $\Upsilon\{A_1+B_1P_2+C_1P_2^2\le 0\}$, with the known coefficients

TABLE 2
Intersection patterns of the best-response correspondences in the interior stage games

Number of roots in the unit interval		Number of MPE in the stage game		
Firm 1	Firm 2	Mixed	Pure	Total
0	0, 1, 2	0	1	1
0, 1, 2	0	0	1	1
1	1	1	2	3
1	2	2	1	3
2	1	2	1	3
2	2	4	1	5

Notes: All possible numbers of pure- and mixed-strategy equilibria, as well as the totals are shown for the (c_1, c_2, c)-stage games where $c_1 > c \geq 0$ and $c_2 > c \geq 0$. When $c = 0$ there can only be a single root on the unit interval for each firm, limiting the number of intersection patterns (see the explanation in the text).

A_1, B_1, C_1 given by[14]

$$\begin{aligned} A_1 &= r_1(c_1,c_2)+\beta h_1(c_1,c_2,c)+\beta\pi(c|c)\big[r_1(c_1,c_2)-K(c)+\beta H_1(c,c_2,c)\big], \\ B_1 &= \beta\big[H_1(c_1,c,c)-h_1(c_1,c_2,c)\big]-\beta\pi(c|c)\big[r_1(c_1,c_2)-K(c)+\beta H_1(c,c_2,c)\big], \\ C_1 &= -\beta^2\pi(c|c)\big[H_1(c,c,c)-H_1(c,c_2,c)\big]. \end{aligned} \tag{43}$$

The second-degree polynomial defined by the coefficients (43) can have zero, one or two real roots on the unit interval depending on the parameter values of the model, and thus the inequality in Υ may be satisfied for any or none $P_2 \in [0,1]$, or be satisfied on a single interval or two disjoint intervals within $[0,1]$. Correspondingly, best-response correspondence of firm 1 $\Upsilon\{A_1 + B_1 P_2 + C_1 P_2^2 \leq 0\}$ may be given by a flat line at 0 or 1, a step from 0 to 1 or from 1 to 0, or have a hump- or U-shape (Figure 6).

The derivation of the best-response correspondence $\Upsilon\{A_2 + B_2 P_1 + C_2 P_1^2 \leq 0\}$ of firm 2 is analogous to that of firm 1. It is not hard to show that because the function Υ is piece-wise equal to 0 or 1, the best-response correspondences drawn on the (P_1, P_2) plane can only intersect in finite number of points. Moreover, depending on whether the real roots of the two polynomials exist and on where they are located, there can be six patterns of intersection of the best-response correspondences, as listed in Table 2. The patterns from the two bottom lines are also shown in the two right-most panels of Figure 6.

Having solved for the MPE values of (P_1, P_2) it is straightforward to express the value functions analytically. The value of investing $v_{I,1}(c_1, c_2, c)$ is linear in P_2 and given by equation (41). The value of not investing $v_{N,1}(c_1, c_2, c)$ can be computed directly using a conditional expression similar to equation (40). If $v_{N,1}(c_1, c_2, c) \leq v_{I,1}(c_1, c_2, c)$, $v_{N,1}(c_1, c_2, c)$ is given by the quadratic equation in (42). If $v_{N,1}(c_1, c_2, c) > v_{I,1}(c_1, c_2, c)$ the value of not investing is given by

$$v_{N,1}(c_1,c,c) = \frac{r_1(c_1,c_2)+\beta\big[P_2(c_1,c_2,c)H_1(c_1,c,c)+(1-P_2(c_1,c_2,c))h_1(c_1,c_2,c)\big]}{1-\beta\pi(c|c)(1-P_2(c_1,c_2,c))}. \tag{44}$$

This concludes the proof: all stage games of the Bertrand pricing and investment game have been considered, and we have shown that there can be only 1, 3, or 5 MPE in each. The solution algorithm is guaranteed to find all MPE of every stage game by computing the roots of second-degree polynomials, and checking which of them are in the unit interval. Thus, the solution algorithm is exact and non-iterative. ‖

Theorem 7 (Solution method for the d-stage games in alternating move leapfrogging game).

Proof. The proof is similar to the proof of Theorem 6, except that the presence of the non-directional "right to move" state variable $m \in \{1, 2\}$ makes the proof for the alternating move version of the Bertrand investment and pricing game somewhat more involved. Recall that m enters as the fourth argument in all functions relating to this version of the game, while subscripts I and N in the value function indicate whether the investment is made or not by the firm which has the right to move.

14. It is straightforward to verify that the value function $v_{N,1}(c_1, c_2, c, P_2)$ is continuous, and that the alternative approach to express it using the regions where $v_{N,1}(c_1, c_2, c, P_2) \geq v_{I,1}(c_1, c_2, c, P_2)$, although gives different expression from equation (42), leads to the same quadratic form in the best-response function and the same coefficients A_1, B_1, and C_1.

The best-response correspondence of firm 1 is given by $\Upsilon\{v_{N,1}(c_1,c_2,c,1)\leq v_{I,1}(c_1,c_2,c,1)\}$ when $m=1$ and because it is not allowed to invest out of turn the investment probability of firm 1 $P_1(c_1,c_2,c,2)=0$ when $m=2$. As before it is helpful to re-write the Bellman equations (31) so that entities dependent only on the subsequent stages are separated. With $\eta=0$ we have for firm 1

$$\begin{aligned} v_{N,1}(c_1,c_2,c,1) &= r_1(c_1,c_2)+\beta\Big[f(1|1)H_1(c_1,c_2,c,1)+f(2|1)H_1(c_1,c_2,c,2)\Big], \qquad (45)\\ v_{N,1}(c_1,c_2,c,2) &= r_1(c_1,c_2)+\beta\Big[f(1|2)H_1(c_1,c_2,c,1)+f(2|2)H_1(c_1,c_2,c,2)\Big],\\ v_{I,1}(c_1,c_2,c,1) &= r_1(c_1,c_2)+\beta\Big[f(1|1)H_1(c,c_2,c,1)+f(2|1)H_1(c,c_2,c,2)\Big]-K(c),\\ v_{I,1}(c_1,c_2,c,2) &= r_1(c_1,c_2)+\beta\Big[f(1|2)H_1(c_1,c,c,1)+f(2|2)H_1(c_1,c,c,2)\Big], \qquad (46)\end{aligned}$$

where the auxiliary functions $H_1(c_1,c_2,c,m)$ are given by

$$\begin{aligned} H_1(c_1,c_2,c,1) &= h_1(c_1,c_2,c,1)+\pi(c|c)\max\big\{v_{N,1}(c_1,c_2,c,1),v_{I,1}(c_1,c_2,c,1)\big\},\\ H_1(c_1,c_2,c,2) &= h_1(c_1,c_2,c,2)+\pi(c|c)\Big[P_2(c_1,c_2,c,2)v_{I,1}(c_1,c_2,c,2)\\ &\qquad +\big(1-P_2(c_1,c_2,c,2)\big)v_{N,1}(c_1,c_2,c,2)\Big],\\ h_1(c_1,c_2,c,1) &= \big(1-\pi(c|c)\big)\int_0^c \max\big\{v_{N,1}(c_1,c_2,c',1),v_{I,1}(c_1,c_2,c',1)\big\}f(c'|c)dc',\\ h_1(c_1,c_2,c,2) &= \big(1-\pi(c|c)\big)\int_0^c \Big[P_2(c_1,c_2,c,2)v_{I,1}(c_1,c_2,c,2)\\ &\qquad +\big(1-P_2(c_1,c_2,c,2)\big)v_{N,1}(c_1,c_2,c,2)\Big]\pi(c'|c)dc'.\end{aligned}$$

It is not hard to see from equation (45) that similar to the simultaneous move game there is a single no-investment pure-strategy MPE at any corner state and a single pure-strategy MPE where one firm does not invest and the other may or may not invest at the edge states. The four value functions at these state points can be found as a solution of a system of four linear equations.

Now consider the interior states (c_1,c_2,c), where $c_1>c$ and $c_2>c$. First, note that the values of investment $v_{I,1}(c_1,c_2,c,1)$ and $v_{I,1}(c_1,c_2,c,2)$ only depend on the subsequent stages, and thus can be computed directly and treated as known. Then as in the proof of Theorem 6, we solve the inequality in Υ eliminating the max operators in equation (45), and thereby express the values of no investment as function of P_2, $v_{N,1}(c_1,c_2,c,1)$ and $v_{I,1}(c_1,c_2,c,2)$

$$\begin{aligned} v_{N,1}(c_1,c_2,c,1) &= r_1(c_1,c_2)+eh_1(c_1,c_2,c,1)+\beta f(1|1)\pi v_{I,1}(c_1,c_2,c) \qquad (47)\\ &\quad +\beta f(2|1)[\pi v_{I,1}(c_1,c_2,c,2)P_2+\pi v_{N,1}(c_1,c_2,c,2)(1-P_2)],\\ v_{N,1}(c_1,c_2,c,2) &= r_1(c_1,c_2)+eh_1(c_1,c_2,c,2)+\beta f(1|2)[\pi v_{I,1}(c_1,c_2,c)] \qquad (48)\\ &\quad +\beta f(2|2)[\pi v_{I,1}(c_1,c_2,c,2)P_2+\pi v_{N,1}(c_1,c_2,c,2)(1-P_2)]\end{aligned}$$

where $eh_1(c_1,c_2,c,m)=\beta f(1|m)h_1(c_1,c_2,c,1)+\beta f(2|m)h_1(c_1,c_2,c,2)$.

Then, combining equations (47) and (48) to eliminate $v_{I,1}(c_1,c_2,c,2)$, we obtain an expression of $v_{N,1}(c_1,c_2,c,1)$ as a function of P_2. Plugging it into Υ leads to a *linear* inequality $\Upsilon\{A_1+B_1P_2\leq 0\}$ with the known coefficients A_1 and B_1 given by

$$\begin{aligned} A_1 &= \beta\pi(c|c)f(2|1)\Big[r_1(c_1,c_2)+eh_1(c_1,c_2,c,2)\Big]+[1-\beta\pi(c|c)f(2|2)]\Big[r_1(c_1,c_2)+eh_1(c_1,c_2,c,1)\Big]\\ &\quad +\Big[\beta\pi(c|c)(f(1|1)+f(2|2))-\beta^2\pi(c|c)^2\det(f)-1\Big]v_{I,1}(c_1,c_2,c,1) \qquad (49)\\ B_1 &= \beta\pi(c|c)\Big[f(2|2)(r_1(c_1,c_2)+eh_1(c_1,c_2,c,1))-f(2|1)(r_1(c_1,c_2)+eh_1(c_1,c_2,c,2))\Big]\\ &\quad +[\beta^2\pi(c|c)^2\det(f)-\beta\pi(c|c)f(2|2)]v_{I,1}(c_1,c_2,c,1)+\beta\pi(c|c)f(2|1)v_{I,1}(c_1,c_2,c,2)\end{aligned}$$

where $\det(f)$ is the determinant of the transition probability matrix for the right of move Markov process m. The fact that the best responses $\Upsilon\{A_1+B_1P_2\leq 0\}$ for firm 1 and similarly $\Upsilon\{A_2+B_2P_1\leq 0\}$ for firm 2 are characterized by the

roots of *linear* functions, makes the stages of the alternating move game simpler compared to the simultaneous move version. Obviously, since these first-degree polynomials can at most have one root on the unit interval, the last three rows of Table 2 do not apply, and therefore there can be only 1 or 3 MPE of any stage game of the alternating move game.

Once we have solved for (P_1, P_2) it is straightforward to compute the values of not investing directly. Similar to the simultaneous move game, we obtain conditional expressions for $v_{N,1}(c_1, c_2, c, 1)$ and $v_{N,1}(c_1, c_2, c, 2)$, depending on whether firm 1 finds it optional to invest.

This completes the proof of Theorem 7 which is compressed for space considerations. We have shown that there can be only 1 or 3 MPE in each of the stage games and that the solution algorithm that is guaranteed to find all equilibria on each stage of the alternating move game is exact and non-iterative. ‖

Acknowledgments. We received many helpful suggestions and comments on earlier versions of this paper and extend particular thanks to Jaap Abbring, Laurent Bouton, Jeffrey Campbell, Eddie Dekel, Ulrich Doraszelski, Roger Lagunoff, David Levine, Andrew McClennan, Stephen Morris, Philip J. Reny, Klaus Ritzberger, Larry Samuelson, Robert Wilson, and other participants in presentations of this paper at numerous seminars, including the first presentation of the paper at the ESEM 2013 Meeting in Gothenburg, Sweden in August 2013. This paper is part of the IRUC research project financed by the Danish Council for Strategic Research (DSF). Financial support is gratefully acknowledged. F.I. acknowledges support from Frisch Centre project 1307, Australian Research Council projects CE110001029 and FL110100247.

Supplementary Data

Supplementary data are available at *Review of Economic Studies* online.

REFERENCES

AUDET, C., HANSEN, P., JAUMARD, B., *et al*. (2001), "Enumeration of All Extreme Equilibria of Bimatrix Games", *SIAM Journal of Scientic Computing*, **23**, 323–338.
ABBRING, J. H., CAMPBELL, J. R., TILLY, J. *et al*. (2014), "Very Simple Markov-Perfect Industry Dynamics", (Discussion Paper, Tilburg University, Center for Economic Research).
ABBRING, J. H., CAMPBELL, J. R. and YANG, N. (2010), "Simple Markov-perfect Industry Dynamics", (Discussion Paper, Federal Reserve Bank of Chicago).
BELLMAN, R. (1957), *Dynamic Programming* (Princeton NJ: Princeton University Press).
BENOIT, J.-P. and KRISHNA, V. (1985), "Finitely Repeated Games", *Econometrica*, **53**, 905–922.
BERNINGHAUS, S. K., GÜTH, W. and SCHOSSER, S. (2014), "Backward Induction or Forward Reasoning? – An Experiment of Stochastic Alternating Offer Bargaining", *International Game Theory Review*, **16**, 144005.
BESANKO, D., DORASZELSKI, U., KRYUKOV, Y. *et al*. (2010), "Learning-by-Doing, Organizational Forgetting, and Industry Dynamics", *Econometrica*, **78**, 453–508.
BORKOVSKY, R. N., DORASZELSKI, U. and KRYUKOV, Y. (2010), "A User's Guide to Solving Dynamic Stochastic Games Using the Homotopy Method", *Operations Research*, **58**, 1116–1132.
CABRAL, L. (2011), "Dynamic Price Competition with Network Effects", *Review of Economic Studies*, **78**, 83–111.
CABRAL, L., and RIORDAN, M. H. (1994), "The Learning Curve, Market Dominance, and Predatory Pricing", *Econometrica*, **62**, 1115–40.
DATTA, R. (2010), "Finding all Nash equilibria of a Finite Game Using Polynomial Algebra", *Economic Theory*, **42**, 55–96.
DUBÉ, J.-P., HITSCH, G. and CHINTAGUNTA, P. K. (2010), "Tipping and Concentration in Markets with Indirect Network Effects", *Marketing Science*, **29**, 216–249.
DORASZELSKI, U. and PAKES, A. (2007), "A framework for applied dynamic analysis in IO" in Armstrong, M. and Porter, R. H. (eds) *Handbook of Industrial Organization* Vol. 3. (North-Holland), 1887–1966.
DORASZELSKI, U. and ESCOBAR, J. (2010), "A Theory of Regular Markov Perfect Equilibria in Dynamic Stochastic Games: Genericity, Stability, and Purification", *Theoretical Economics* forthcoming.
DORASZELSKI, U. and SATTERTHWAITE, M. (2010), "Computable Markov-Perfect Industry Dynamics", *Rand Journal of Economics*, **41**, 215–243.
HALLER, H. and LAGUNOFF, R. (2000), "Genericity and Markovian Behavior in Stochastic Games", *Econometrica*, **68**, 1231–1248.
HÖRNER, J., SUGAYA, T., TAKAHASHI, S. *et al*. (2011), "Recursive Methods in Discounted Stochastic Games: An Algorithm for $\delta \to 1$ and a Folk Theorem", *Econometrica*, **79**, 1277–1318.
ISKHAKOV, F., SCHJERNING, B. and RUST, J. (2015), "The Dynamics of Bertrand Price Competition with Cost-Reducing Investments" Georgetown University, manuscript.
JUDD, K. (1998), "*Numerical Methods in Economics*" (The MIT Press).
JUDD, K., RENNER, P. and SCHMEDDERS, K. (2012a), "Finding All Pure-Strategy Equilibria in Games with Continuous Strategies", *Quantitative Economics*, **3**, 289–331.
JUDD, K., SCHMEDDERS, K. and YELTEKIN, S. (2012b), "Optimal Rules for Patent Races" *International Economic Review*, **53**, 23–51.

KUHN, H. W. (1953) "Extensive games and the problem of information", in Kuhn, H. W. and Tucker A. W. (eds) *Contributions to the Theory of Games I* (Princeton NJ: Princeton University Press), 193–216.
MASKIN, E. and TIROLE, J. (1988), "A Theory of Dynamic Oligopoly, I: Overview and Quantity Competition with Large Fixed Costs", *Econometrica*, **56**, 549–569.
MASKIN, E. and TIROLE, J. (2001), "Markov Perfect Equilibrium I. Observerable Actions", **100**, 191–219.
PAKES, A. and McGUIRE, P. (1994), "Computing Markov-perfect Nash Equilibria: Numerical implications of a dynamic differentiated product model", *RAND Journal of Economics*, **25**, 555–589.
PAKES, A. and McGUIRE, P. (2001), "Stochastic Algorithms, Symmetric Markov Perfect Equilibrium, and the Curse of Dimensionality", *Econometrica*, **69**, 1261–1281.
RIORDAN, M. H. and SALANT, D. J. (1994), "Preemptive Adoptions of an Emerging Technology", *Journal of industrial economics*, **42**, 247–261.
RITZBERGER, K. (1999), "Recall in Extensive Form Games", *International Journal of Game Theory*, **28**, 69–87.
RITZBERGER, K. (2002), *Foundations of Non-Cooperative Game Theory* (Oxford University Press).
ROUTLEDGE, R. R. (2010), "Bertrand competition with cost uncertainty", *Economics Letters* **107**, 356–359.
RUBINSTEIN, A. (1982) "Perfect Equilibrium in a Bargaining Model", *Econometrica*, **50**, 97–109.
RUST, J. (1987) "Optimal Replacement of GMC Bus Engines: An Empirical Model of Harold Zurcher", *Econometrica*, **55**, 999–1033.
SELTEN, R. (1965), "Spieltheoretische Behandlung eines Oligopolmodells mit Nachfrageträgheit", *Zeitschrift für die gesamte Staatswissenschaft*, **121**, 301–324.
SELTEN, R. (1975), "Re-examination of the Perfectness Concept for Equilibrium Points in Extensive Games", *International Journal of Game Theory*, **4**, 25–55.
SHAPLEY, L. S. (1953) "Stochastic games", *Proceedings of the National Academy of Sciences of the United States of America*, **39**, 1095.