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
Bertel Schjerning
University of Copenhagen

September, 2013

Abstract: We define a class of dynamic Markovian games that we call *directional dynamic games* (DDG) in which directionality is represented by a partial order on the state space. We show that every finite state DDG can be decomposed into a finite number of *stages* that contain simpler dynamic games we call *stage games*. We propose a fast and robust *state recursion algorithm* that can find a Markov perfect equilibrium (MPE) via backward induction on the stages of the game. When there are multiple equilibria the state recursion algorithm relies on an *equilibrium selection rule* (ESR) to pick a particular MPE in each stage game that results in a MPE for the overall DDG. We propose a *recursive lexicographic search* (RLS) algorithm that systematically and efficiently cycles through *all feasible* ESRs. When there is a finite number of MPE and an algorithm that can find all of them in every stage game of a DDG, it is possible to use the RLS algorithm to find *all* MPE of the overall game. We apply the RLS algorithm to find all MPE of a dynamic duopoly model of Bertrand price competition and leapfrogging investments by firms that can acquire an exogenously improving state of the art production technology. We show this is a DDG where the directionality comes from the unidirectional evolution of the state of the art production technology. The stage games of this DDG are *anti-coordination games* that typically have either one, three or five MPE, and we provide an algorithm that finds all of them. Using the RLS algorithm, we find that there can be hundreds of millions of MPE in a simultaneous move version of the leapfrogging game even if the state space has a relatively small number of points. However, the RLS algorithm reveals that there is a unique MPE when the firms move in an alternating fashion and the state of the art technology improves with probability 1 in every period.

Keywords: 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 classification: D92, L11, L13

[†] Preliminary: This version of this paper is not for general distribution and quotation. We received many helpful suggestions and extend particular thanks to Jaap Abbring, Laurent Bouton, Jeffrey Campbell, Eddie Dekel, Ulrich Doraszelski, Roger Lagunoff, David Levine, Stephen Morris, Klaus Ritzberger, Larry Samuelson, Robert Wilson, and other participants in presentations of this paper at the ESEM 2013 Meeting in Gothenburg, Sweden in August 2013.

[‡] **Correspondence address:** ARC Centre of Excellence in Population Ageing Research (CEPAR), University of New South Wales, Sydney 2052, Australia, phone: (+61)299319202, email: f.iskhakov@unsw.edu.au

1 Introduction

Dynamic games have had a major impact on both economic theory and applied work over the last four decades, and much of it has been inspired by the Markov perfect equilibrium (MPE) solution concept due to Maskin and Tirole (1988). While there has been considerable progress in the development of algorithms for computing or approximating an MPE of these games, including the pioneering work by Pakes and McGuire 1994, it still remains an extremely challenging computational 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).

Though there has also been recent progress on *homotopy methods* for finding multiple equilibria of both static and dynamic games (Borkovsky *et. al.* 2010 and Besanko *et. al.* 2010) as well as algebraic approaches for finding all equilibria in cases where the equilibrium conditions can be expressed as certain classes of polynomial equations (Datta, 2010 and Judd *et. al.* 2012), the homotopy methods do not generally find all equilibria, and the algebraic methods are limited to problems where the equations defining the state-specific Nash equilibria can be expressed as systems of polynomial equations that have specific forms.

This paper reports progress on a different approach for computing all MPE that is applicable to a class of dynamic Markovian games that we refer to as *dynamic directional games* or DDG's. Of course, *every* dynamic game is inherently directional in the sense that the play of the game unfolds through time. However, we show that many dynamic games exhibit a different type of directionality that is not directly linked to the passage of calendar time. Our new concept of directionality pertains to the stochastic evolution of the *state* of the game.

A DDG is a game where a subset of the state variables evolve in a manner that satisfies certain conditions including an intuitive notion of "directionality." When the state space is finite we can exploit this directionality and partition it into a finite number of elements we call "stages". Examples of DDGs include chess, Rubinstein's (1982) model of bargaining, but over a stochastically

shrinking pie, and many examples in industrial organization such as patent races where part of the state of the game represents "technological progress" that improves over time. We solve a model of Bertrand pricing with leapfrogging investments that is an example of this type.

Similar to the "arrow of time" the evolution of the directional component of the state space is unidirectional: we can index the stages by τ and order them from 1 to $\mathcal{T}$. 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 stage game is different than the traditional notion of a stage game in a repeated game, which is a single period or static game. The stage games will also generally be dynamic games, though on a much reduced state space that makes them much simpler than the overall game we are trying to solve.

We show that a MPE for the overall dynamic game can be recursively constructed from the MPE selected for each of the component stage games. We propose a state recursion algorithm that computes a MPE of the overall game in a finite number of steps. State recursion 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 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 analyzed is performed on the *game tree* which is the extensive form representation of the game. State recursion is not performed on the game tree, but rather can be viewed as a type of backward induction that is performed on a different object, a *directed acyclic graph* (DAG) that summarizes the directionality of the game in terms of the state space instead of the temporal ordering implied by the game tree.

If a dynamic game exhibits directionality in the state space, state recursion can be 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 to do backward induction on the game tree. The usual method

for finding a MPE in infinite horizon problems involves iterating on the Bellman equations of the players starting from some initial guess of the value functions. Thus, this type of time-based backward induction is equivalent to using the method of *successive approximations* to find a fixed point of the system of Bellman equations for each of the players. However, it is well known that in dynamic games the Bellman equations generally do not satisfy the requisite continuity conditions to constitute contraction mappings that would be sufficient to guarantee that the successive approximations will converge to a fixed point and hence a MPE of the DDG.[1] Thus, there is no guarantee that time-based backward induction methods of this type will even be able to find a single MPE of the game.

State recursion, however, does not suffer from this problem: conditional on the availability of the solution method for stage games *it will return a 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 will not cycle or fail to converge, or approach a candidate MPE only asymptotically as the number of iterations or steps tends to infinity, unlike what happens with time-based recursions such as successive approximations on the players' Bellman equations.

State recursion finds a *single* MPE of the overall DDG, but when the game has multiple equilibria the found MPE depends on which equilibrium is chosen in the end game and all other stages of the game by the state recursion algorithm. Assume that there is an algorithm that can find *all* MPE of each of the stage games of the DDG and that the number of MPE in each stage is finite. We introduce the *Recursive Lexicographical Search* (RLS) algorithm that repeatedly invokes state recursion in an efficient way to compute *all* MPE of the DDG by systematically cycling through all *feasible equilibrium selection rules* (ESRs) for each of the component stage games of the DDG.

The general idea of how the presence of multiple equilibria of a stage game can be used to construct a much larger set of equilibria in the overall game was used by Benoit and Krishna (1985) to show that a version of the "Folk Theorem" can hold in finitely repeated games. The

[1] Note that the contraction property does hold in single agent games which 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, his "best reply to nature". Nevertheless, we show that when there is directionality in single agent dynamic programming problems, state recursion will be far faster than time-based backward induction, and will actually converge to the exact solution of the problem in a finite number of steps.

prevailing view prior to their work was that the sort of multiplicity of equilibria implied by the Folk Theorem for infinitely repeated games cannot happen in finitely repeated games because a backward induction argument from the last period of the game was thought to generally result in a unique equilibrium of the overall repeated game. However Benoit and Krishna showed that when there are multiple equilibria in the stage games, this can be used to create a much larger set of subgame perfect equilibria in the finitely repeated game, and that Folk Theorem sorts of multiplicity can emerge even in finitely repeated games when the time horizon is sufficiently large. However, Benoit and Krishna did not propose an algorithm or a constructive approach for enumerating all possible subgame perfect equilibria of a finitely repeated game, whereas the RLS algorithm we propose can be used to find and enumerate all such equilibria.

Though we do not claim that all dynamic games will have exploitable directional structure, we show there is a sense in which the RLS algorithm can approximate the set of all MPE to a wide class of finite and infinite-horizon dynamic games, even if there is no exploitable directionality in the game other than the passage of time. Fudenberg and Levine (1983) showed that infinite horizon dynamic games that satisfy a continuity condition have the property that the limit of the set of all MPE of a sequence of T period games converges to the set of all MPE of the infinite horizon game as $T \to \infty$. If we can find all Nash equilibria of each of the (static) stage games that are encountered in the process of finding a subgame perfect equilibrium of the T-period game via standard (time-based) backward induction, then the RLS algorithm will find all MPE of the T period game. If T is sufficiently large, this set will be close to the set of all MPE of the infinite horizon game. In effect, we show how the standard backward induction procedure *performed in the right way* (not as successive approximations to the players' Bellman equations) can approximate all MPE of a fairly broad class of infinite horizon games. We view this as an analog of Benoit and Krishna's Folk Theorem approximation result for finitely repeated games.

We use the RLS algorithm to find all MPE of two variants of a dynamic duopoly model of Bertrand price competition with leapfrogging investments. These are not trivial examples, but rather substantive contributions to the literature in their own right. The RLS algorithm revealed important new insights into the nature of long run price competition between duopolists when there is stochastic evolution of a state of the art production technology — a class of models that

are not well understood since they have not been analyzed previously. In our first example, we use RLS to find all MPE of a simultaneous move version of the investment and pricing game analyzed by Iskhakov et. al. 2013, (hereafter abbreviated IRS). The IRS model is a dynamic duopoly model of Bertrand price competition with cost-reducing investments, where the duopolists can invest in an exogenously improving state of the art production technology in an attempt to gain a production cost advantage over their rival, at least temporarily. IRS assume a constant returns to scale production technology, so the state of the game can be described by the triple (c_1, c_2, c), where c_1 is the marginal cost of firm 1, c_2 is the marginal cost of firm 2, and c represents the marginal cost under the state of the art production technology. The directionality of the game results from the facts that a) the state of the art only improves over time, so c decreases stochastically or deterministically but never increases, b) the legacy marginal costs of firms 1 and 2 will never deteriorate but will only improve if one or the other invests to acquire the state of the art production technology. This implies that the state space of the game is a "quarter pyramid", $S = \{(c_1, c_2, c) | c_1 \geq c,\ c_2 \geq c,\ c \geq 0\}$. If we further assume that the state of the art marginal costs c can only be one of a finite number of possible values and evolves as an exogenous Markov chain, then we can show that this game satisfies our definition of a finite state DDG.

We show that the stage games in this problem are *anti-coordination games* that typically have either one, three, or five MPE, and we provide an algorithm that efficiently computes all of them. We then show how state recursion can be used to combine these stage game MPE to find a MPE for the overall Bertrand duopoly game. Then we provide a detailed explanation of how RLS can be applied to find *all* MPE. We show that even for problems where the state space has a relatively small number of points, there can be hundreds of millions of MPE in the overall duopoly pricing and investment game. We also show how traditional approaches such as value function approximation can fail to find even a single MPE. Further, we illustrate the danger of the common practice in modeling of restricting attention to *symmetric equilibria* of the game. RLS reveals that only a small fraction of the full set of MPE are symmetric equilibria and these are generally inefficient mixed strategy equilibria. Traditional backward induction (successive approximation of the Bellman equations) fail to find *any* of these symmetric mixed strategy MPE.

Our second example is the alternating move version of the IRS model. This example presents

the complication that not all state variables are directional, since the right to move alternates forth and back between the two firms (in deterministic or stochastic fashion). We show that this game is still a DDG since we can partition the overall state variable $s = (c_1, c_2, c, m)$ (where m denotes which of the firms has the current right to invest) into a *directional component* $d = (c_1, c_2, c)$ and a *non-directional component* m. Consequently, we can still solve the alternating move version of the leapfrogging model by state recursion and find all MPE using the RLS algorithm. We show that in the alternating move case the structure of MPE are very different compared to the simultaneous move case. Generally there are fewer equilibria and certain "extremal" equilibria such as a zero profit mixed strategy equilibrium or two asymmetric monopoly equilibria no longer exist in the alternating move version of the game. The RLS algorithm reveals that when the state of the art cost c improves in a strictly monotonic fashion (i.e. there is zero probability that it will remain in the same state for more than one time period), then the DDG has a *unique* MPE.

The rest of the paper is organized as follows. In section 2 we define a notion of directionality and the class of DDGs. We introduce the concepts of stages of a DDG, define our new concept of "stage games" and introduce the state recursion algorithm and prove that it finds a 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 will find 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 of IRS described above. Section 5 discusses some extensions of our solution method, including the possibility of relaxing some of the assumptions we make, and discussing broader areas of application of the state recursion and RLS algorithms, and then concludes.

2 Finite state directional dynamic games and state recursion

In this section we define a class of Markovian games that have the property of *directionality,* and we refer to games that have this property as *dynamic directional games* or DDGs. We use directionality to simplify the problem of finding equilibria of these games using a *state recursion algorithm* that is a generalization of the standard *backward induction algorithm* that is typically used to find equilibria of dynamic games. However, the traditional approach is to use *time* as the

index for the backward induction, whereas the state recursion algorithm uses an index derived from the directionality in the law of motion for the *states*. The state recursion algorithm finds a single MPE of the game in a finite number of steps, but it requires the user to specify an *equilibrium selection rule* (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 is 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 their stream of payoffs in the game 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 that provides the probability distribution of 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 *Markov perfect equilibrium* (MPE) concept of Maskin and Tirole (1988) where we limit attention to a subset of all subgame perfect Nash equilibria of the game $\mathcal{G}$, namely equilibria where the players use strategies that are *Markovian*, i.e. they are functions only of the current state s_t and not the entire past history of the game.[2]

In this paper we follow Haller and Lagunoff (2000) and focus on games $\mathcal{G}$ that have a finite state space, since they provide general conditions under which the set of MPE of such games are *generically finite.* To this end, let S denote the set of all states the game may visit at any time

[2]Though we do not take the space to provide a full extensive form description of the game $\mathcal{G}$ we do assume that the players have *perfect recall* and are therefore able to condition on the entire history of states in 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 player i's opponents are using Markovian strategies, player i can find a best reply to these strategies within the subclass of Markovian strategies. Given this, we can provide a fully rigorous definition of Markov perfect equilibrium using Bellman equations for the players without having to devote the space necessary to provide a complete extensive form description of $\mathcal{G}$.

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.[3] 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 can also allow for players to have and condition their decisions on private information in the form of idiosyncratic shocks, perhaps dependent on the state[4] though to keep notation simple 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 $A_i(s)$, the law of motion $p(s'|s,a)$, and the probability distributions for independently distributed private shocks to the payoffs of the players is common knowledge.

Let σ denote a feasible set of Markovian *behavior* 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. A *Markov perfect equilibrium* of the stochastic game $\mathcal{G}$ is a pair of 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:

[3] 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.

[4] In this case the *conditional independence* assumption of Rust (1987) holds, allowing the players to compute the expectations over the actions of their opponents in Bellman equations (1).

1. the system of Bellman equations

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

is satisfied for every $i = 1,\ldots,n$, with the expectation in (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 Bellman equation

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

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 that is a subset of $a^*_i(s)$. The expectation in (2) taken in the same way as in (1).

Let $\mathcal{E}(\mathcal{G})$ denote the set of all Markov-perfect equilibria of the game $\mathcal{G}$.

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.

2.2 Directional Dynamic Games

Before we formally define dynamic directional games, it is useful to provide intuitive examples of what we mean by a *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 such that the transitions between the points identical in these dimensions can be represented as a *directed acyclic graph* (DAG), where each vertex represents a point d which is a part of state vector, and the arrows (directed edges)

connecting the vertices correspond to positive probabilities of transiting from one value of d to another.[5] We will refer to the component of the state vector d as "directional" component of the state space below.

Figure 1 presents two directed graphs representing transitions in space state of two examples of dynamic Markov games we discuss below. In these games state space is one dimensional, and is given by $S = \{d_1, d_2, d_3, d_4\}$. The game presented in the left panel progresses (stochastically according to the indicated transition probabilities) from d_1 to d_4, and it is essential that independent of what state the game is in, there is alway a zero probability of returning from a point with higher index to the point with lower index. This intuitive notion of directionality is violated in the right panel, where the game may indefinitely oscillate between states 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 denote this strategy-specific partial order of S. Most games that we analyze 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.* 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.[6]

Definition 2 (Representation of state transitions)**.** 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

[5] 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 state space. In particular, the game tree can not have "loop-backs" (transitions 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.

[6] 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 $x' \in X$.

cartesian product $S = D \times X$ where $D \subset R^N$ and $X \subset R^{k-N}$ where $N \leq k$. 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 states $s \in S$ induced by the strategy profile σ is defined as

$$d' \succ_\sigma d \quad \textit{iff} \quad \exists x \in X \; \rho\{d'|d,x,\sigma\} > 0 \quad \textit{and} \quad \forall x' \in X \; \rho\{d|d',x',\sigma\} = 0. \tag{3}$$

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 of the lemma above and all subsequent results (except those that are short and intuitive and are provided in the text) a 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 $d' \neq d \in D$

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

It is not hard to show that when 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 visualized with 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 No Loop Condition. Therefore, 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 not comparable states (implying that 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 more complicated examples, a total, strategy independent order needed to do state recursion has to be specifically constructed. We explain how to do this below.

Example 2 (Directional bargaining over 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 $0 < d_4 < d_3 < d_2 < d_1$. Suppose

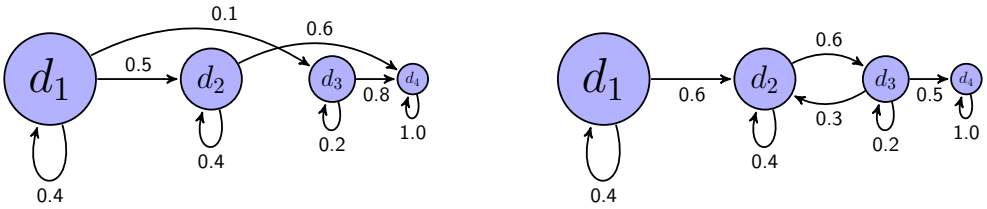

Figure 1: Bargaining over a stochastically shrinking pie (Example 2: left panel, Example 3: right panel)

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 $x \in \{1,2\}$ denote which of the players has the turn to make an offer, so player x 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 of the pie 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 $s = (d,x)$ where directional variable is the size of the pie d and the non-directional variable x is the index of the player who has the turn to move first. A version of this game was solved by Berninghaus, Güth and Schosser (2012) using a backward induction calculation in the d variable that is an example of the state recursion algorithm we define below.

Example 3 (Non-directional bargaining over shrinking pie)**.** 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 the shown 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.[7]

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 *consistent* partial orders if and only if for any $d', d \in D$ we have

$$\textit{if} \quad d' \succ_\sigma d \quad \textit{then} \quad d \not\succ_{\sigma'} d', \tag{5}$$

or equivalently that $\succ_\sigma \subset \not\succ_{\sigma'}$ with inclusion operator defined as inclusion of the sets of ordered pairs that constitute the binary relations.

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 define the concept of a *directional dynamic game* (DDG).

Definition 5 (Directional Dynamic Games)**.** We say that a dynamic Markovian game $\mathcal{G}$ with state space S is a *directional dynamic game* (DDG) if given the decomposition of the state space in directional and non-directional components $S = D \times X$, the following conditions hold:

[7] 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, joint (d_2, d_3) and 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.

1. every strategy $\sigma \in \Sigma(\mathcal{G})$ has no loops in directional component D according to Definition 3, and

2. the set of induced partial orders on D, $\{\succ_\sigma | \sigma \in \Sigma(\mathcal{G})\}$, are pairwise *consistent* according to Definition 4,

where $\Sigma(\mathcal{G})$ is the set of all feasible strategies of the dynamic Markovian game $\mathcal{G}$.

2.3 Stage games and subgame perfection

Even though the different strategy-specific partial orders $\succ_\sigma$ are consistent with each other, they may nevertheless be different from each other. In order to define the *state recursion algorithm* for computing a 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 the backward induction in state space. Moreover we prove that this ordering allows for the overall game $\mathcal{G}$ to be decomposed into a recursive 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

$$d' \succ_\sigma d \quad \Longrightarrow \quad d' \succ_{\sigma'} d, \tag{6}$$

or equivalently using the inclusion operation on partial orders, $\succ_\sigma \subset \succ_{\sigma'}$.

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. *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 $\mathcal{N}(D(\mathcal{G})) \neq \emptyset$ 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 $\mathcal{G}$ can be exhausted in finite number of applications of the recursive operator

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

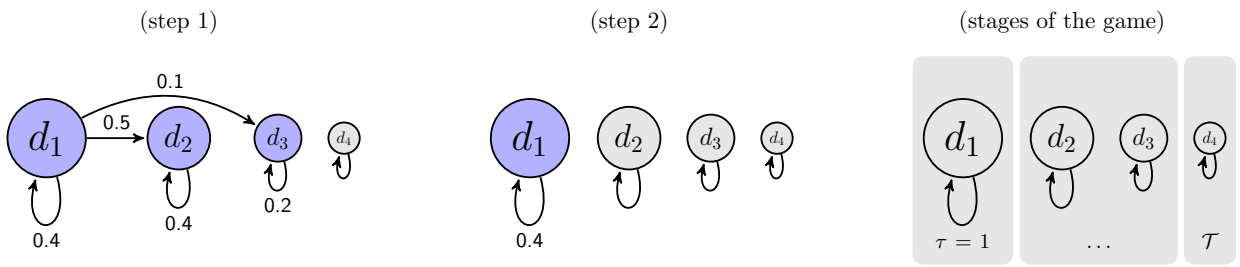

Figure 2: DAG recursion and stages of the DDG in Example 2 ($\mathcal{T} = 3$).

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. $\mathcal{T} < \infty$.*

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 for 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 when it is not immediately obvious.

Example 4. Figure 2 provides an illustration of the described 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 (9) in Lemma 2, let $\{D_1, \ldots, D_{\mathcal{T}}\}$ denote the partition of the directional component D, which is indexed with the *inverted* index τ, such that D_τ contains the points corresponding to the vertices of DAG $D_{\mathcal{T}-j}(\mathcal{G})$. (The right-most panel of Figure 2 presents this partition graphically for the game in Example 2.) We are now ready to define the stages of the game $\mathcal{G}$ using this partition.

Definition 8 (Stages of a DDG)**.** Let $\mathcal{G}$ be 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 (9) as explained above. Let

$$S_\tau = D_\tau \times X \tag{10}$$

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

The $\{S_1, \ldots, S_{\mathcal{T}}\}$ is a partition of the state space S into stages. 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 strategies. 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 will allow us to define a new concept of "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 as a subset of S by

$$\Omega_\tau = \cup_{t=\tau}^{\mathcal{T}} S_t, \tag{11}$$

and let G_τ denote the DDG with state space Ω_τ and other elements of the game (number of players, time horizon, utility functions, discount factors, action sets and laws of motion) be properly restricted for this state space versions of the element of the original game $\mathcal{G}$. Then we say that G_τ is the *stage* τ *subgame* of the DDG $\mathcal{G}$.

The state recursion algorithm, defined below, involves finding a MPE of the overall game $\mathcal{G}$ inductively, starting by finding MPEs at all points in the *endgame,* i.e. the stage $\mathcal{T}$ subgame $\mathcal{G}_{\mathcal{T}}$, and proceeding by backward induction over the stages of the game, from stage $\mathcal{T}-1$ to stage $\mathcal{T}-2$ until the initial stage 1 is reached and solved. When stage 1 is solved in this backward induction procedure, effectively the whole $\mathcal{G}$ is also solved, as follows from the following lemma.

Lemma 3. *If $\mathcal{G}$ is a finite state DDG, and $\mathcal{G}_1$ is its stage 1 subgame, then $\mathcal{G} = \mathcal{G}_1$.*

Note that if the partition elements D_τ contain more than one element of D, then there can be no transitions between the various elements in D_τ by virtue of the way the partition $\{D_1, \ldots, D_{\mathcal{T}}\}$ was constructed from the DAG recursion in Lemma 2. 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. It is useful to define an even finer grained notion of subgames of $\mathcal{G}$ that we call a *d-subgames* $G_\tau(d)$. Since there is zero probability of transitions between $d_{i,\tau}$ and $d_{j,\tau}$ for $i \neq j$, these finer subgames can be solved independently of each other in the state recursion algorithm below.

Definition 10 (d-subgames of $\mathcal{G}$)**.** Let τ be a stage of the finite state DDG $\mathcal{G}$. Consider $d \in D_\tau \subset D$. The d-subgame of $\mathcal{G}$, denoted by $\mathcal{G}_\tau(d)$, is the subgame of $\mathcal{G}$ defined in the similar way as subgame $\mathcal{G}_\tau$ on the state space $\Omega_\tau(d) \subset S$ given by

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

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 in the following way. Imagine that the game starts at a point $s_1 = (d_1, x_1) \in S_1 \subset S$ 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 $\{d_1 \times X\}$, but the game will remain in the same substage until the state of the game transitions to the next stage.

The DAG-recursion that constructs the stages of $\mathcal{G}$ rules out the possibility that a substage of some stage S_τ for $\tau < \mathcal{T}$ could be an absorbing class of states, since such states will be identified as terminal nodes of the DAG $D(\mathcal{G})$ of the DAG-recursion, (9). Thus, these will all be *transient states* and only the final stage $S_{\mathcal{T}}$ of the game will be an absorbing class of states. The final stage

too will be partitioned into substages that do not communicate with each other, so each substage of the terminal stage S_T will constitute separate absorbing sets of points.

Let $\mathcal{E}(\mathcal{G})$ denote the set of all MPE of $\mathcal{G}$. In case there are multiple MPEs in some of the 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 is possible to d ($d \succ_{\mathcal{G}} d'$) will be dependent on which of the MPEs of the d-subgames will eventually be played on the later stage. This implies that in case of multiplicity of equilibria in $\mathcal{G}$ (and thus it's subgames), the solution computed with the backward induction approach is dependent on the *equilibrium selection rule* (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. Let $e(\mathcal{G}) \in \mathcal{E}(\mathcal{G})$ denote a particular selected MPE from the set of all MPE of $\mathcal{G}$.

Definition 11 (Equilibrium selection rule)**.** Let Γ denote a *deterministic* rule for selecting one of the MPE from every d-subgame $\mathcal{G}_\tau(d)$, i.e.

$$e(\mathcal{G}_\tau(d)) = \Gamma(\mathcal{E}(\mathcal{G}_\tau(d))) \quad \forall d \in D. \tag{13}$$

By selecting an equilibrium in every d-subgame, ESR Γ also induces (or selects) an equilibrium in every subgame $\mathcal{G}_\tau$, $e(\mathcal{G}_\tau) = \Gamma(\mathcal{E}(\mathcal{G}_\tau))$. We can also interpret $e(\mathcal{G}_\tau)$ as a MPE formed from the union the equilibria at each d-subgame $\mathcal{G}_\tau(d)$.

Recall from the Definition 1 of MPE, that an equilibrium consists of two objects: the n-tuple of the players' strategies and the n-tuple of the value functions, so $e(\mathcal{G}) = (\sigma^*, V)$. Define the projections $e_\sigma(\mathcal{G}) = \sigma^*$ and $e_V(\mathcal{G}) = V$ that pick each of these objects from a given equilibrium.

The state recursion algorithm is a method for constructing a MPE for the overall DDG $\mathcal{G}$ by recursively calculating MPEs for a recursively defined sequence of "smaller games" that we refer to as *generalized stage games* (though in what follows below, for brevity we refer to them simply as "stage games"). Note that our definition of stage game is different from the definition that is traditionally used in the theory of repeated games. In a repeated game, the stage game is a *single period game* and the repeated game $\mathcal{G}$ is a finite or infinite repetition of these stage games. Each stage game is itself generally a dynamic game. This dynamic game is played for a random length

of time until the state of the system transits out of the substage $(d \times X)$ that defines the (restricted) state space of this stage game.

A MPE of each stage game involves calculating the set of all equilibria on a much smaller subset of the state space than the full state space S of the overall DDG $\mathcal{G}$. The state space for each of the stage games is $(d \times X)$ where $d \in D_\tau$ for some stage of the game $\tau \in \{1, \ldots, \mathcal{T}\}$. Further, we can restrict our search for MPE of the stage games to *continuation strategies* which only require calculating all MPE (and then selecting a particular one of them) on the state space $(d \times X)$ of the stage game, and then reverting to an already calculated and selected MPE for all subsequent stages of the game after stage τ. The power of the state recursion algorithm comes from its ability to decompose the problem of finding a MPE of the much larger and more complex overall DDG $\mathcal{G}$ into the much more tractable problem of recursively finding a MPE for an appropriately defined sequence of these stage games. This need only be done once, so that state recursion will find a 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 of each other at every stage of the backward induction over τ) and sequentially over the various stages of the game τ starting at $\tau = \mathcal{T}$ and working backward.

Definition 12 (Continuation strategies)**.** Let $\mathcal{G}$ be a finite state DDG, and consider a particular stage of this game $\tau \in \{1, \ldots, \mathcal{T}\}$. If $\mathcal{G}_\tau(d)$ is a d-subgame, define the *d-continuation strategy* $\sigma_\tau(s|(d \times X), e_\sigma(\mathcal{G}_{\tau+1}))$ to be any feasible Markovian strategy for points $s \in (d \times X)$ and $d \in D_\tau$ that reverts to a MPE strategy $e_\sigma(\mathcal{G}_{\tau+1})$ in the stage $\tau+1$ subgame $\mathcal{G}_{\tau+1}$. That is,

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

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

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

Definition 13 (Stage game)**.** Let $\mathcal{G}$ be a finite state DDG, and consider a particular stage of the game $\tau \in \{1,\ldots,\mathcal{T}\}$ 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, i.e. if $\Sigma(\mathcal{SG}_\tau(d))$ is the set of feasible, Markovian strategies of the stage game and $\Sigma(\mathcal{G}_\tau(d))$ is the set of feasible Markovian strategies of the d-subgame $\mathcal{G}_\tau(d)$, then we have

$$\sigma \in \Sigma(\mathcal{SG}_\tau(d)) \quad \textit{iff} \quad \sigma(s) = \sigma_\tau(s|(d\times X), e_\sigma(\mathcal{G}_{\tau+1})), \quad s \in (d\times X)\cup\Omega_{\tau+1}. \tag{16}$$

Similarly, we define $\mathcal{SG}_\tau$ to be the *stage game at stage* τ by restricting the set of all feasible Markovian strategies in the stage τ subgame to continuation strategies. It follows that $\Sigma(\mathcal{SG}_\tau) \subset \Sigma(\mathcal{G}_\tau)$ where we have

$$\sigma \in \Sigma(\mathcal{SG}_\tau) \quad \textit{iff} \quad \sigma(s) = \sigma_\tau(s|S_\tau, e_\sigma(\mathcal{G}_{\tau+1})). \tag{17}$$

Lemma 4. *Let $\mathcal{G}$ be a finite state DDG, and consider the final stage of the game $\mathcal{T}$. For each $d \in D_\mathcal{T}$ we have*

$$\mathcal{SG}_\mathcal{T}(d) = \mathcal{G}_\mathcal{T}(d), \quad d \in D_\mathcal{T}, \tag{18}$$

and

$$\mathcal{SG}_\mathcal{T} = \mathcal{G}_\mathcal{T}. \tag{19}$$

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 stage game $\mathcal{G}_\tau$ is a subset of the feasible Markovian strategies in the stage τ subgame $\mathcal{G}_\tau$. By restricting strategies in this way, we reduce the problem of finding MPE strategies of a stage game $\mathcal{SG}_\tau(d)$ to the much smaller, more tractable problem of computing a MPE on the reduced state space $(d\times X)$ instead of the much larger state space $\Omega_\tau(d)$ given in equation (12) of definition 10.

Theorem 2 (Subgame perfection)**.** *Let $\mathcal{E}(\mathcal{SG}_\tau(d))$ be the set of all MPE of the stage game $\mathcal{SG}_\tau(d)$ and let $\mathcal{E}(\mathcal{G}_\tau(d))$ be the set of all MPE of the d-subgame $\mathcal{G}_\tau(d)$. Then we have*

$$\mathcal{E}(\mathcal{SG}_\tau(d)) = \mathcal{E}(\mathcal{G}_\tau(d)) \tag{20}$$

i.e. there is no loss in generality from computing all MPE of every d-subgame $\mathcal{G}_\tau(d)$ by restricting the search for equilibria to finding all MPE of the corresponding stage game $\mathcal{SG}_\tau(d)$ using only continuation strategies.

Corollary 2.1. *Let $\mathcal{E}(\mathcal{SG}_\tau)$ be the set of all MPE of the stage game at stage τ and let $\mathcal{E}(\mathcal{G}_\tau)$ be the set of all MPE equilibria of the stage τ subgame $\mathcal{G}_\tau$. Then we have*

$$\mathcal{E}(\mathcal{SG}_\tau) = \mathcal{E}(\mathcal{G}_\tau). \tag{21}$$

Theorem 2 and its corollary 2.1 provide the foundation for the validity of the state recursion algorithm. They justify a backward recursion process for computing a MPE of the DDG $\mathcal{G}$ that is very similar in spirit to the use of backward induction to compute a subgame-perfect equilibrium of an extensive form game. We require one final result before providing a formal statement of the state recursion algorithm and proving the key result of this section, namely that this algorithm will compute a MPE of the DDG $\mathcal{G}$.

Theorem 3 (Decomposition of the stage game)**.** *Let $\mathcal{G}$ be a finite state DDG with $\mathcal{T}$ stages. At each stage $\tau \in \{1,\ldots,\mathcal{T}\}$, let $D_\tau = \{d_{1,\tau},\ldots,d_{n_\tau,\tau}\}$ be the set of possible values of the directional state variable d that can occur at stage τ. We have the following decomposition of the MPE of the stage game at stage τ, $\mathcal{E}(\mathcal{SG}_\tau)$, as a partition of the equilibria of its d-stage games $\mathcal{E}(\mathcal{SG}_\tau(d))$:*

$$\mathcal{E}(\mathcal{SG}_\tau) = \cup_{i=1}^{n_\tau} \mathcal{E}(\mathcal{SG}_\tau(d_{i,\tau})) \tag{22}$$

where

$$\mathcal{E}(\mathcal{SG}_\tau(d_{i,\tau})) \cap \mathcal{E}(\mathcal{SG}_\tau(d_{j,\tau})) = \emptyset, \quad i \in j \tag{23}$$

where the union of the possible equilibria in the various component d-stage games can be interpreted as defining an equilibrium (σ, V) whose domain is the union of the disjoint domains $(d_{i,\tau} \times X)$, for $i = 1,\ldots,n_\tau$. The stage games comprising stage τ are payoff-independent of each other, i.e. the players' payoffs in $\mathcal{SG}_\tau(d_{i,\tau})$ is unaffected by the choice of strategy $\sigma \in \Sigma(\mathcal{SG}_\tau(d_{j,\tau}))$ in any other stage game $\mathcal{SG}_\tau(d_{j,\tau})$, $d_{j,\tau} \neq d_{i,\tau}$, in the same stage τ of $\mathcal{G}$.

2.4 State Recursion

Definition 14 (**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:

for $\tau = \mathcal{T}, \mathcal{T}-1, \ldots, 1$ do

for $i = 1, \ldots, n_\tau$ do

- compute $\mathcal{E}(\mathcal{SG}_\tau(d_{i,\tau}))$.
- using an equilibrium selection rule Γ, select a particular MPE from $\mathcal{E}(\mathcal{SG}_\tau(d_{i,\tau}))$, $e(\mathcal{SG}_\tau(d_{i,\tau})) = \Gamma(\mathcal{E}(\mathcal{SG}_\tau(d_{i,\tau})))$.
- By Theorem 2, $e(\mathcal{SG}_\tau(d_{i,\tau}))$ is a MPE of the d-subgame $\mathcal{G}_\tau(d_{i,\tau})$,

◦ End of i do-loop. Using the decomposition property (22) of Theorem 3, the union of the MPEs for each d-stage game $\{e(\mathcal{SG}_\tau(d_{i,\tau}) | i = 1, \ldots, n_\tau\}$ is a MPE for the overall stage game at stage τ, $e(\mathcal{SG}_\tau)$.

◦ By Theorem 2 a MPE of the τ-stage game $\mathcal{SG}_\tau$ is also a MPE of the stage τ subgame, $\mathcal{G}_\tau$. That is, $e(\mathcal{SG}_\tau) = e(\mathcal{G}_\tau)$.

Theorem 4 (Convergence of State Recursion)**.** *Let $\mathcal{G}$ be a finite state DDG. The state recursion algorithm given in Definition 14 computes a MPE of $\mathcal{G}$.*

The state recursion algorithm given in definition 14 leads to a recursively defined MPE for each stage τ stage game $\mathcal{SG}_\tau$, $\tau = (1, \ldots, \mathcal{T})$. By Theorem 2, these MPE also constitute MPE of the stage τ subgames $\mathcal{G}_\tau$, $\tau = (1, \ldots, \mathcal{T})$. However by Lemma 3 we have $\mathcal{G}_1 = \mathcal{G}$, so it follows that $e(\mathcal{G}_1) = e(\mathcal{G})$, i.e. the state recursion algorithm has computed a MPE of the DDG $\mathcal{G}$ by computing MPE for a total of

$$N = \sum_{\tau=1}^{\mathcal{T}} n_\tau \tag{24}$$

d-stage games of the game $\mathcal{G}$. By Lemma 3 we have $\mathcal{G}_1 = \mathcal{G}$, so it follows that $e(\mathcal{G}_1) = e(\mathcal{G})$. Thus, it follows that the state recursion algorithm has computed a MPE of the DDG $\mathcal{G}$.

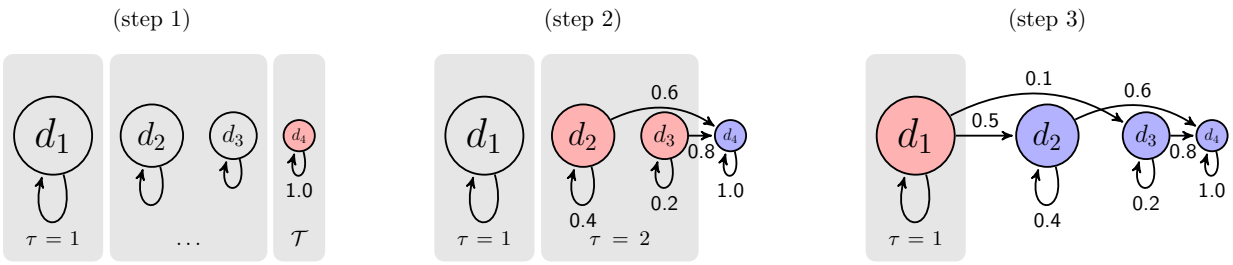

Figure 3: Graphical illustration of state recursion on the DAG $D(\mathcal{G})$ in Example 2.

Example 5. Continuing with the DDG shrinking pie example (Example 2), Figure 3 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 ($\mathcal{T} = 3$), 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 3). By assumption, solution method exists for every d-stage game, and at the first step of the state recursion algorithm it is applied to d_4-stage game.

Given the solution of the d_4-stage game, the algorithm moves on to stage game $\tau = 2$ shown in middle panel of Figure 3. 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 3, 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 *Recursive Lexicographical Search* algorithm (RLS) 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* $\mathcal{G}$. That is, a choice of a particular MPE for any stage game at any stage τ of $\mathcal{G}$ can potentially alter the set of possible MPE at all earlier stages $\tau' < \tau$. For example, it is possible that the one choice of MPE for a stage game of the end game $\tau = \mathcal{T}$ of $\mathcal{G}$ might result in a unique MPE at a stage game at some earlier stage $\tau < \mathcal{T}$, whereas a different choice of MPE of the same stage game of the end game of $\mathcal{G}$ could result in *multiple* MPE existing at the same earlier stage game at level $\tau < \mathcal{T}$ of $\mathcal{G}$.

3.1 Prerequisites

Note that our theoretical 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 have a state space $\{d \times X\}$ that is generally a small subset of of the overall state space S for $\mathcal{G}$ itself, and also because we restrict our search for MPE in the stage games to 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 and constructing all possible MPE of every stage game of $\mathcal{G}$. Theorem 2 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 a) re-using solutions from previously computed stage games of $\mathcal{G}$ wherever possible, and b) by efficiently and rapidly disregarding large numbers of potential but *infeasible* combinations of stage game MPE of $\mathcal{G}$.

RLS is applicable to DDGs that have a *finite* number of possible MPE. If we assume that the algorithm that computes all of the d-stage game equilibria can also detect if a particular stage game has an infinite number of equilibria then even though RLS will not be able to compute all MPE of $\mathcal{G}$, it will be able to establish that the game has infinite number of MPE. Otherwise, the RLS will provide a complete enumeration of all of them.

Finally, 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 (ESS)

Let K denote the least upper bound on the number of possible equilibria in any stage game of $\mathcal{G}$. We introduce K to simplify the explanation of the RLS algorithm, but we will show that is it not necessary for the user to know the value K *a priori.* Instead, the RLS algorithm will reveal the value K to the user when the algorithm terminates. Recall that N given equation (44) of section 2 represents the total number of substages of the DDG $\mathcal{G}$. The state recursion algorithm must loop over all N of these substages to find a MPE in the stage games that correspond to each of these N substages to construct a MPE of $\mathcal{G}$.

Definition 15 (Equilibrium Selection Strings)**.** An *equilibrium selection string* (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 $\mathcal{G}$ that is ordered from right to left in the same order of the stages of $\mathcal{G}$, i.e.

$$\gamma = (\gamma_T, \gamma_{T-1}, \ldots, \gamma_1), \tag{25}$$

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 also restricted to the set $\{0, 1, \ldots, K-1\}$

$$\gamma_\tau = (\gamma_{1,\tau}, \ldots, \gamma_{n_\tau,\tau}) \tag{26}$$

where n_τ equals the number of substages of stage τ of the DDG $\mathcal{G}$.

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 that consists of N zeros.

We assume that the user fixes some ordering of the set of all equilibria at each d-stage of $\mathcal{G}$, so that they can be indexed from 0 to at most $K-1$. The individual components or "digits" of the ESS $\gamma_{j,\tau}$ index (in base K) which of the K possible MPE are selected in each of the d-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 at least when the number of MPE of the game $\mathcal{G}$ is finite ($K < \infty$). The initial ESS γ^0 is the selection rule that picks the first equilibrium in every d-stage game (which is always possible due to our assumption of existence of at least one MPE in every stage game).

It is very 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 (25) is *essential* for the RLS algorithm to work correctly. However, due to the payoff-independence property for the n_τ component stage games $\mathcal{SG}_\tau(d_{i,\tau})$, $i = 1, \ldots, n_\tau$ at each stage τ of $\mathcal{G}$ (Theorem 3 of section 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 6. Consider an arbitrary DDG with the induced DAG presented in the left panel of Figure 1 and the stages of the game presented in Figure 2 and 3. 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 equilibrium string would be $\gamma = (0, 2, 2, 1)$, indicating that the *first* MPE is selected in stage $\tau = 3$ (the index for equilibria starts from 0), the *third* MPE is selected in both substages of the at

stage $\tau = 2$, and the *second* MPE is selected at stage $\tau = 1$. Due to the decomposition property (Theorem 3), the choice of an MPE for the first substage of stage $\tau = 2$ has no effect on the set of possible MPE in the second substage, but different choices of MPE in these stages may affect the number of MPE and the values of the MPE at stage $\tau = 1$.

Note that there are K^N possible equilibrium strings for the DDG $\mathcal{G}$, so this represents an upper bound on the number of possible MPE of $\mathcal{G}$. However, there will generally be far fewer MPE than this. We can enumerate all possible equilibrium strings by doing mod(K) addition, starting from the base equilibrium string γ^0. If we form the base K representations of the integers $\{0,1,\ldots,K^N-1\}$, we obtain K^N corresponding equilibrium strings $\{\gamma^0,\gamma^1,\ldots,\gamma^{K^N-1}\}$ which form the set of all *possible* equilibrium selection strings that are N digits long.

Now consider the addition operation in base K and its representation as an equilibrium string. Starting from the always feasible equilibrium string $\gamma^0 = (0,\ldots,0)$, which is the base-K representation of the integer 0, we add 1 to this to get the next possible equilibrium string, γ^1 which is the base-K representation of the integer 1, i.e $\gamma_1 = (0,0,\ldots,0,1)$. The string γ^1 may or may not be a feasible ESS because there may be only a *single* MPE at the d_{1,n_1}-stage game of $\mathcal{G}$. If there is only a single MPE in this substage, then the equilibrium string γ_1 is *infeasible* because it corresponds to choosing the first MPE (which is guaranteed to exist) at every stage game of $\mathcal{G}$ except for $\mathcal{SG}_1(d_{1,n_1})$, where the 1 in the right-most component of γ^1 indicates that the *second* MPE is to be selected for this stage game. However, there is no second MPE for this stage game, and hence we say that γ^1 is an infeasible equilibrium string. We show that the RLS algorithm can quickly determine feasibility and will immediately skip over infeasible ESSs and "jump" directly to the next feasible one, or terminate if it reaches the last ESS γ^{K^N-1}. In the latter case, the RLS algorithm will have established that $\mathcal{G}$ has a *unique MPE*, namely the MPE corresponding to the equilibrium string γ^0.

Definition 16 (Feasible Equilibrium Selection String)**.** An equilibrium string γ is *feasible* if all of its digits index a 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 stage game of $\mathcal{G}$ under the ESR implied by the equilibrium string γ. We define $ne(\gamma)$ using the same format as the

equilibrium string, so that the digits of the equilibrium string γ are in one to one correspondence with the elements of the vector $ne(\gamma)$ as follows:

$$ne(\gamma) = \Big(ne_{\mathcal{T}}, ne_{\mathcal{T}-1}\big(\gamma_{>\mathcal{T}-1}\big), \ldots, ne_1\big(\gamma_{>1}\big)\Big), \tag{27}$$

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 stage games $\mathcal{SG}_\tau(d_{i,\tau})$, $i = 1, \ldots, n_\tau$ of stage τ of $\mathcal{G}$,

$$ne_\tau(\gamma_{>\tau}) = \Big(ne_{1,\tau}\big(\gamma_{>\tau}\big)), \ldots, ne_{n_\tau,\tau}\big(\gamma_{>\tau}\big)\Big). \tag{28}$$

The vector $ne(\gamma) \in Z_+^N$ summarizes how the number of possible MPE at any stage τ of $\mathcal{G}$ depends on the choices of the MPE at the endgame and all stages after τ 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 endgame $\mathcal{T}$ 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 equilibrium string γ. Further, by the decomposition property for stage games in any stage τ of $\mathcal{G}$ (Theorem 3 of section 2), the number of possible MPE at every sub-stage game $\mathcal{SG}_\tau(d_{i,\tau})$, $i = 1, \ldots, n_\tau$ of stage τ depends only on the equilibrium strings $\gamma_{>\tau}$ and not on the choice of MPE in other substage games $\mathcal{SG}_\tau(d_{j,\tau})$, $j \neq i$ of stage τ.

Lemma 5. *The ESS γ is feasible if and only if it holds*

$$\gamma_{i,\tau} < ne_{i,\tau}(\gamma_{>\tau}), \quad \tau = 1, \ldots, \mathcal{T}, \quad i = 1, \ldots, n_\tau \tag{29}$$

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 component of a candidate equilibrium string γ to check feasibility. We will generally have to check far fewer than K^N possible equilibrium strings for feasibility. But it should be evident that due to the one-to-one correspondence between an ESS and an integer (the ESS is the base-K representation of an integer), a simple do-loop over the integers $\{0, 1, \ldots, K^N - 1\}$ is

a way to systematically enumerate all possible equilibrium strings, and thus all possible choices of MPE at each substage of $\mathcal{G}$. However this "brute force" enumeration is not efficient because typically there are huge gaps between the feasible ESSs in this full enumeration loop. We devise a vastly more efficient approach that *jumps* directly to the next *feasible* ESS γ. Consequently, the RLS algorithm has a run time that is *linear* in $|\mathcal{EG}|$, the total number of MPE of $\mathcal{G}$. 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 integers in the different components or digits of 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 component stage games of $\mathcal{G}$ after state recursion was run with ESR γ. Continuing the example above, if $\gamma = (0,2,2,1)$ indexes a particular choice of MPE in the 3 stages of $\mathcal{G}$, suppose the corresponding number of equilibria in these three 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 to 2.

An ESS γ is in one to one correspondence with an integer (i.e. it is a variable base representation of an integer) in very much the same way as γ is a representation of an integer 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

$$\iota(\gamma) = \sum_{j=1}^{N} \gamma_{i(j),\tau(j)} \prod_{j'=1}^{j-1} ne_{i(j'),\tau(j')}(\gamma_{>\tau(j')}) \tag{30}$$

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, $\iota(0,2,2,1) = 1+2\times 3+2\times 3\times 3+0\times 3\times 3\times 3 = 25$, and $\iota(0,2,2,2) = 26$, so $(0,2,2,2)$ is the largest number in this system.

Since an ESS γ can be viewed as a variable representation of an integer, we can do all of the ordinary arithmetic, including addition and subtraction. Addition can be done as we were all

taught in elementary school for numbers in base-10, namely to start on the right and add to the first digit, "carrying" the remainder mod(10) to the next digit of the number if adding a number causes the first digit to exceed 10. In variable base addition we do the same thing, except we use a different base for determining how much to carry in each successive digit of the number.

We can define the *successor function* $\mathcal{S} : Z_+^N \to Z^N$ by the γ' that results from 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 if the successor will not exist because it represents an integer that is larger than the largest integer than can be represented with N and the variable base $ne(\gamma)$. Since all of the components of a feasible ESS γ are nonnegative, we will define the result of successor operator when there is "overflow" to be a vector in Z^N all of whose components equal -1.

Now we show how variable base arithmetic can be used to define a very effective procedure for *jumping* from one feasible ESS γ to another one.

Definition 17 (Jump function)**.** Let $\mathcal{J} : Z_+^N \to Z_+^N$ be defined by

$$\mathcal{J}(\gamma) = \begin{cases} \textit{argmin}_{\gamma'}\{\iota(\gamma') | \iota(\gamma') > \iota(\gamma) \text{ and } \gamma' \text{ is feasible}\} \\ (-1, \ldots, -1) \quad \text{if there is no feasible } \gamma' \text{ satisfying } \iota(\gamma') > \iota(\gamma). \end{cases} \tag{31}$$

Thus, $\mathcal{J}(\gamma)$ is the "smallest" ESS *after* γ that is also a feasible ESS.

Lemma 6. *If γ is a feasible ESS, then $\mathcal{J}(\gamma) = \mathcal{S}(\gamma)$.*

What Lemma 6 tells us is that we can easily jump to the next feasible ESS in the lexicographical order by simply using variable base arithmetic with bases $ne(\gamma)$ and adding 1 to the ESS γ using successor function $\mathcal{S}(\gamma)$ defined above.

3.4 Recursive Lexicographical Search (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.

RLS initialization

- Set $i = 0$ and let $\gamma^0 = (0, 0, \ldots, 0, 0)$ be the always feasible N-digit ESS that corresponds to the ESR for the DDG $\mathcal{G}$ where the first MPE is selected at each d-subgame. Run the state recursion algorithm to calculate an MPE of $\mathcal{G}$ corresponding to this ESR and label all of the MPE in each stage game and record the number of possible MPE in each stage game of $\mathcal{G}$ in the $N \times 1$ vector $ne(\gamma^0)$.
- Let Λ be the set of feasible ESS found by RLS. Set $\Lambda = \{\gamma^0\}$.

Main RLS do-loop

- Compute $\gamma^{i+1} = \mathcal{J}(\gamma^i)$, the next candidate feasible ESS. Let j_0 denote the highest digit of the ESS that changed.
- *Stopping rule:* If $\gamma^{i+1} = (-1, \ldots, -1)$ then RLS stops and has computed all MPE.
- Otherwise γ^{i+1} is a feasible ESS by Lemma 6. Run *partial* state recursion for the stages τ' which are dependent on stage τ where j_0 belongs, i.e. $\tau' < \tau$, and using the ESR implied by the ESS γ^{i+1}, index the MPE of every stage game of $\mathcal{G}$, and record the total number of MPE found at each stage in the $N \times 1$ vector $ne(\gamma^{i+1})$.
- Update the set of feasible ESS found by RLS by setting $\Lambda = \Lambda \cup \{\gamma^{i+1}\}$.
- Update the loop counter by setting $i = i + 1$ and continue the main RLS do-loop.

3.5 RLS finds *all* MPE in a finite number of steps

The RLS algorithm must terminate in a finite number of steps since there at most K^N ESSs of length N. Upon termination the set Λ will contain a finite number J of feasible ESSs, $\Lambda = \{\gamma^0, \ldots, \gamma^{J-1}\}$. We now prove that $J = |\mathcal{E}(\mathcal{G})|$, i.e. the RLS algorithm has found *all* MPE of the DDG $\mathcal{G}$. We first prove a stepping stone result, namely that RLS finds all MPE that are "generated" by a given selection of MPE for the stage games in stages $\mathcal{T}$ to τ of $\mathcal{G}$.

We introduce some final notation so we can state and prove a Lemma that is a key stepping stone to the proof of the main result of this section, Theorem 5 below. If $\gamma \in \Lambda$ is a feasible ESS

returned by the RLS algorithm, let e_γ be the MPE of $\mathcal{G}$ that is implied by the ESS γ. That is, each ESS γ corresponds to an ESR Γ and so if $\Gamma(\gamma)$ is the ESR implied by the ESS γ, then we have

$$e_\gamma \equiv \Gamma(\gamma)(\mathcal{E}(\mathcal{G})). \tag{32}$$

Now if e_γ is the MPE of $\mathcal{G}$ implied by the ESS γ, Theorem 2 implies that via the use of continuation strategies, e_γ implies or induces a MPE for all of the stage games of $\mathcal{G}$, so $e_\gamma(\mathcal{SG}_\tau(d))$ denotes the MPE on the d-stage game $\mathcal{SG}_\tau(d)$ induced by the ESS γ, and $e_\gamma(\mathcal{S}_\tau(d))$ is the MPE on the d-subgame induced by γ.

Lemma 7. *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 after stage τ, i.e. the players use e_γ to define a continuation strategy for stages $\tau+1,\ldots,\mathcal{T}$. If $e \in \mathcal{E}_\tau(\mathcal{G}|e_\gamma)$, then there exists a $\gamma' \in \Lambda$ such that $e = e_{\gamma'}$.*

With Lemma 7 in hand, it is now possible to prove the main theorem of this section, i.e. that the RLS algorithm finds *all* MPE of the DDG $\mathcal{G}$.

Theorem 5. *Assume there exists an algorithm that can find all MPE of every stage game of the DDG $\mathcal{G}$, and that the number of these equilibria is finite in every stage game. Then the RLS algorithm above will find all MPE of DDG $\mathcal{G}$ in at most $|\mathcal{E}(\mathcal{G})|$ steps, which is the total number of MPE of the DDG $\mathcal{G}$.*

It is important to emphasize that the *RLS algorithm requires no prior knowledge of the maximum number of MPE K of any stage game of $\mathcal{G}$*. 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 in state points where the solution may have changed. In this way the RLS algorithm can systematically search for all MPE of $\mathcal{G}$ even though the user has no prior knowledge of how many MPE $\mathcal{G}$ or any of its stage games might have.

We conclude this section by stating the approximation result that we discussed in the introduction, that shows that the RLS algorithm is a very general method for approximating the set of all MPE of *arbitrary* infinite horizon dynamic games, including games $\mathcal{G}$ that have no exploitable directionality other than the directionality of time itself.

Theorem 6. *Consider a finite state, infinite horizon dynamic game $\mathcal{G}$ that has no exploitable directional structure, indicated by the situation where the directional component of the state space D contains only a single element. Consider approximating the infinite horizon game $\mathcal{G}$ by a finite horizon game $\mathcal{G}_T$ for some large but finite value of $T < \infty$. Let the directional component of $\mathcal{G}_T$ be the time index, i.e. $D = \{1, \ldots, T\}$ and assume that there exists and algorithm that can find all MPE of the T stage games of $\mathcal{G}_T$. Then the RLS will find all MPE of $\mathcal{G}_T$ and under the conditions of Fudenberg and Levine (1983), we have*

$$\lim_{T \to \infty} \mathcal{E}(\mathcal{G}_T) = \mathcal{E}(\mathcal{G}), \tag{33}$$

so RLS is able to approximate the set of all MPE of finite state, infinite-horizon but non-directional games $\mathcal{G}$ that satisfy the continuity conditions in Fudenberg and Levine (1983).

4 Applications of State Recursion and the RLS Algorithm

In this section we present an example of dynamic stochastic directional games and show how we solve this game using the state recursion and the recursive lexicographical search algorithms developed in the previous sections. We consider two versions of a dynamic model of Bertrand price competition with cost-reducing investments analyzed by Iskhakov et. al (2013). The first example is the simultaneous move version of this investment and pricing game, all dimensions of which are directional and thus the stage games are relatively easy to solve. Our second example is the alternating move version of the same model. Because the right of move alternates back and forth between the two duopolists, the state variable indicating whose turn it is to move in a given time period becomes non-directional. We show however, that it is still possible to find all stage game equilibria, despite the additional complications induced by the non-directional dimension. Consequently, the alternating move version of the leapfrogging model is also suitable to be handled by by the RLS algorithm and we can thus find all equilibria of this game as well.

4.1 Bertrand price and investment game with simultaneous moves

We begin with the simultaneous move formulate of the leapfrogging model of Iskhakov et al. (2013). The description of the model is abridged. Please refer to Iskhakov et al. (2013) for

economic motivation and greater detail on the model.

The model

We consider a discrete-time, infinite horizon stochastic game where two firms j, $j \in \{1,2\}$, are producing an identical good at a constant marginal cost of c_1 and c_2, respectively. We assume that the two firms are price setters, have no fixed costs and face no capacity constraints when producing the good. We also assume that demand is perfectly elastic. Under these assumptions, the Bertrand equilibrium for the two firms is for the lower cost firm to serve the entire market at a price $p(c_1,c_2)$ equal to the marginal cost of production of the higher cost rival, $p(c_1,c_2) = \max[c_1,c_2]$. We let $r_1(c_1,c_2)$ denote the expected profits that firm 1 earns in a single period equilibrium play of the Bertrand-Nash pricing game when the two firms have costs of production c_1 and c_2, respectively.

$$r_1(c_1,c_2) = \begin{cases} 0 & \text{if } c_1 \geq c_2 \\ \max[c_1,c_2] - c_1 & \text{otherwise.} \end{cases} \tag{34}$$

and the profits for firm 2, $r_2(c_1,c_2)$ are defined symmetrically, so we have $r_2(c_1,c_2) = r_1(c_2,c_1)$.

The two firms have the ability to make an investment to replace their existing plant with a new state of the art production facility. If either one of the firms purchases the current state of the art technology, then after a one period lag, the firm can produce at the new marginal cost of production, c_t. Stochastic technological progress drives down the state of the art marginal cost of production over time, such that c_t evolves according to a exogenous Markov process with transition probability $\pi(c_{t+1}|c_t)$. With probability $\pi(c_t|c_t)$ we have $c_{t+1} = c_t$ (i.e. there is no improvement in the state of the art technology at $t+1$), and with probability $1-\pi(c_t|c_t)$ the technology improves, so that $c_{t+1} < c_t$ and c_{t+1} is a draw from some discrete distribution over the interval $[0,c_t]$. Both firms have equal access to the new technology conditional on paying an investment cost $K(c_t)$.

Each firm j incurs idiosyncratic "disruption costs" (or subsidies) $\eta\varepsilon_{t,j} = (\eta\varepsilon_{0,t,j}, \eta\varepsilon_{1,t,j})$ associated with each of the choices of *not to invest* ($\eta\varepsilon_{0,t,j}$) and *to invest* ($\eta\varepsilon_{1,t,j}$) respectively. It is common knowledge among the two firms that $\{\eta\varepsilon_{t,1}\}$ and $\{\eta\varepsilon_{t,2}\}$ are independent *IID* Type I bivariate extreme value processes with common scale parameter $\eta \geq 0$. Firm j observes its current

and past idiosyncratic investment shocks $\{\eta\varepsilon_{t,j}\}$, but does does not observe its future shocks or it's opponent's past, present, or future idiosyncratic investment cost shocks.

The timing of events in the model is as follows. Each period, both firms observe the state of the the industry, set their prices and *simultaneously* decides 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 period t is performed with their existing plants independent of their investment decisions.

Assuming that the two firms are expected discounted profit maximizers and have a common discount factor $\beta \in (0,1)$, we define the stationary *Markov Perfect Equilibrium* of the duopoly investment and pricing game as 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 then investment and pricing strategies of its opponent. The value functions V_j, $j = 1,2$ take the form

$$V_j(c_1,c_2,c,\varepsilon_{0,j},\varepsilon_{1,j}) = \max\left[v_{I,j}(c_1,c_2,c)+\eta\varepsilon_{0,j},\ v_{N,j}(c_1,c_2,c)+\eta\varepsilon_{1,j}\right] \tag{35}$$

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

$$v_{N,j}(c_1,c_2,c) = r_j(c_1,c_2)+\beta EV_j(c_1,c_2,c), \tag{36}$$

$$v_{I,j}(c_1,c_2,c) = r_j(c_1,c_2)-K(c)+\beta EV_j(c_1,c_2,c), \tag{37}$$

where $EV_j(c_1,c_2,c)$ denotes the conditional expectation of firm j's next period value functions $V_j(c_1,c_2,c,\varepsilon_{0,j},\varepsilon_{1,j})$ depending on whether the firm invest this period or not. The expected value function summarize firms' expectations about future technological development governed by $\pi(c_{t+1}|c_t)$, opponent's investment and pricing decisions and the future idiosyncratic cost components $\eta\varepsilon_{t,j}$. 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 $(\varepsilon_{0,j},\varepsilon_{1,j})$ that only firm j observes. But since firm i knows the probability distribution of these shocks, it can calculate it's 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 $(\varepsilon_{0,j}, \varepsilon_{1,j})$, P_j is given by the binary logit formula

$$P_j(c_1, c_2, c) = \frac{\exp\{v_{I,j}(c_1, c_2, c)/\eta\}}{\exp\{v_{N,j}(c_1, c_2, c)/\eta\} + \exp\{v_{I,j}(c_1, c_2, c)/\eta\}}. \tag{38}$$

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 $(\varepsilon_{0,j}, \varepsilon_{1,j})$ 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

$$\begin{aligned}
&\int_{\varepsilon_0^j} \int_{\varepsilon_1^j} V_j(c_1, c_2, c, \varepsilon_{0,j}, \varepsilon_{1,j}) q(\varepsilon_{0,j}) q(\varepsilon_{1,j}) d\varepsilon_{1,j} d\varepsilon_{0,j} = \\
&\eta \log\left[\exp\{v_{N,j}(c_1, c_2, c)/\eta\} + \exp\{v_{I,j}(c_1, c_2, c)/\eta\}\right] = \\
&\phi(v_{N,j}(c_1, c_2, c), v_{I,j}(c_1, c_2, c)),
\end{aligned} \tag{39}$$

where we use $\phi()$ to denote the log-sum formula. Using this notation, we are now ready to present the system of Bellman equations for the simultaneous move version of the model, namely

$$\begin{aligned}
v_{N,1}(c_1, c_2, c) &= r_1(c_1, c_2) + \beta \int_0^c \left[P_2(c_1, c_2, c)\phi(v_{N,1}(c_1, c, c'), v_{I,1}(c_1, c, c')) + \right. \\
&\quad \left. (1 - P_2(c_1, c_2, c))\phi(v_{N,1}(c_1, c_2, c'), v_{I,1}(c_1, c_2, c'))\right] \pi(dc'|c). \\
v_{I,1}(c_1, c_2, c) &= r_1(c_1, c_2) - K(c) + \beta \int_0^c \left[P_2(c_1, c_2, c)\phi(v_{N,1}(c, c, c'), v_{I,1}(c, c, c')) + \right. \\
&\quad \left. (1 - P_2(c_1, c_2, c))\phi(v_{N,1}(c, c_2, c'), v_{I,1}(c, c_2, c'))\right] \pi(dc'|c), \\
v_{N,2}(c_1, c_2, c) &= r_2(c_1, c_2) + \beta \int_0^c \left[P_1(c_1, c_2, c)\phi(v_{N,2}(c, c_2, c'), v_{I,2}(c, c_2, c')) + \right. \\
&\quad \left. (1 - P_1(c_1, c_2, c))\phi(v_{N,2}(c_1, c_2, c'), v_{I,2}(c_1, c_2, c'))\right] \pi(dc'|c). \\
v_{I,2}(c_1, c_2, c) &= r_2(c_1, c_2) - K(c) + \beta \int_0^c \left[P_1(c_1, c_2, c)\phi(v_{N,2}(c, c, c'), v_{I,2}(c, c, c')) + \right. \\
&\quad \left. (1 - P_1(c_1, c_2, c))\phi(v_{N,2}(c_1, c, c'), v_{I,2}(c_1, c, c'))\right] \pi(dc'|c).
\end{aligned} \tag{40}$$

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 $c_t = 0$[8] making finite discrete approximations of the state space possible. Moreover, it is easy to see, that the remaining two state variables

[8] We assume without loss of generality that state of the art cost may asymptotically approach but never cross zero.

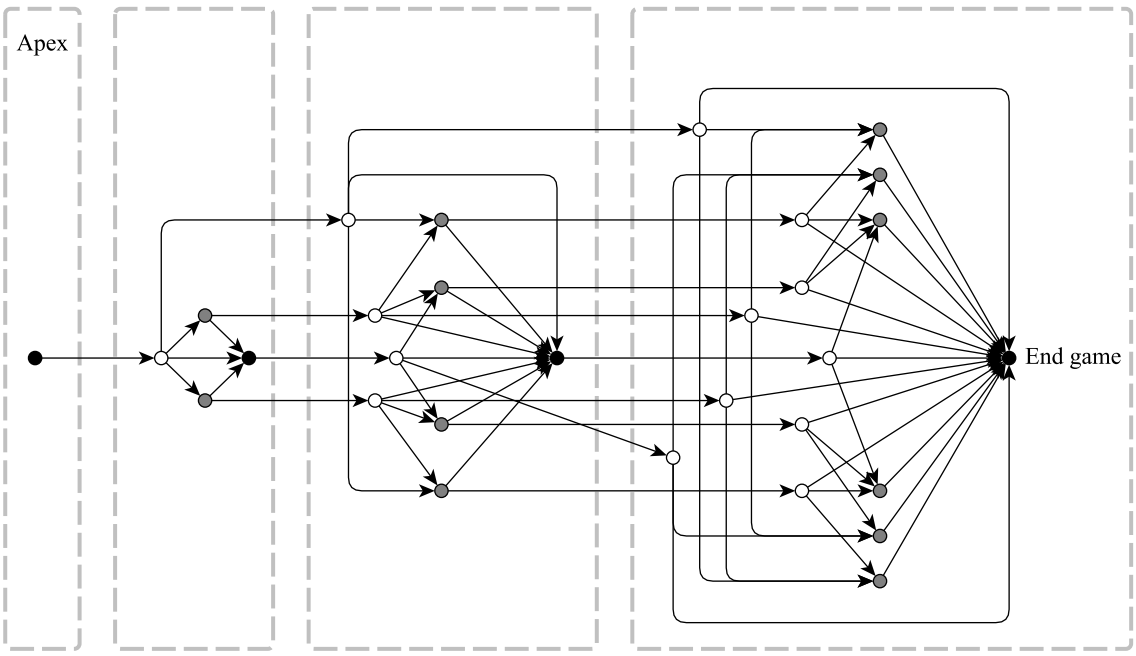

Figure 4: Possible transitions between points in the state space in dynamic Bertrand investment and pricing game with $N = 4$. Each dot represents a vector (c_1, c_2, c). Dashed boxes enclose different layers of the state space pyramid: from the apex to the end game. White colored dots in each layer 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 transitive reduction are shown between layers, full set of transitions can be reconstructed by considering the transitive closure of the presented graph.

in this model, $(c_1$ and $c_2)$ are directional as well although they are strategy dependent and thus evolve endogenously. Since there is no depreciation in the model that would ever cause costs to rise, it follows that once firms' costs attain the state of the art level at some period t, they will remain at this level or below in all future periods. Hence, all state variables of the simultaneous move Bertrand pricing and investment game belong to the "directional" component of the state space. Under every feasible strategy, every component of the cost vector (c_1, c_2, c) will only move towards the absorbing state $c = 0$.

The directional structure of Bertrand investment and pricing game is illustrated in Figure 4 for the case where c is defined on the grid with $N = 4$ values. 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 rightmost box). White colored dots in each layer represent "interior points" in each layer, i.e. (c_1, c_2, c) where $c_1 > c, c_2 > c$. 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 $c_1 = c_2 = c$.

Consider first possible transitions from an interior point, (c_1, c_2, c). In absence of technological improvement, the state variables will only move towards the "edges" (c, c_2, c) and (c_1, c, c) if either firm 1 or firm 2 invest, or to the (c, c, c) "corner" if both firms invests simultaneously. These transitions are indicated with arrows from white dots to grey and black dots respectively. From the edges it is only possible to move to the corner (unless the technology improves) as indicated by the arrows from grey to black dots. If technology improves, so that $c_{t+1} < c_t$, 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 4.

I terms the notation in section 2, the directional variable is equal to the entire state vector, $d = (c_1, c_2, c)$ since there are no non-directional variables in this game, i.e, $S = D$ and X is singleton. The endgame stage, denoted by $\tau = \mathcal{T}$, is the $(0, 0, 0)$ corner in the present model, corresponding to the right most black dot in Figure 4. In the endgame $\mathcal{G}(\mathcal{T})$ consists of a single point $(0, 0, 0)$, where the state recursion starts.

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 among each other (i.e. there is zero probability of going from one value of $d = (c_1, 0, 0)$ to another $d' = (c_1', 0, 0)$) 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 be to the stage $\mathcal{T}$. Because of no communication, each of these d-subgames is solved independently in the inner loop of the stage recursion algorithm. In accordance with Theorem 2 by solution here we mean finding an equilibrium in d-stage game among continuation strategies which revert to the optimal actions in state $(0, 0, 0)$ (stage $\mathcal{T}$) that were already found in the previous step of the stage

recursion algorithm.

State recursion algorithm proceeds to the next stage $\tau = \mathcal{T} - 2$ which is composed of the interior points in the bottom layer where $c_1 > c$, $c_2 > c$, and $c = 0$. These points also don't communicate with each other, and thus form $(N-1)^2$ d-subgames that are also solved independently 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 $c > 0$. We then continue the backward induction in state space in this way through stages of the game $\tau = \mathcal{T} - 2, \mathcal{T} - 3, ..., 2, 1$. At each stage τ all the different d-subgames of the game $\mathcal{G}(\tau)$ are solved independently given a particular equilibrium selection at lower stages of the game. Once we have solved the $\mathcal{G}(1)$ subgame at the apex (c_0, c_0, c_0), we have solved the entire game since $\mathcal{G}(1) = \mathcal{G}$

Theorem 7 (Solution method for the d-stage games in simultaneous move leapfrogging game)**.** *Given a fixed equilibrium selection rule* Γ*, solution method for every d-subgame in the Bertrand pricing and investment game with simultaneous moves exists and when* $\eta = 0$ *is guaranteed to find all d-subgame MPE for every* $d = (c_1, c_2, c)$.[9]

Finding all MPE using RLS

Theorem 7 establishes that state recursion is guaranteed to find all d-subgame MPE given a fixed equilibrium selection rule Γ. We will now show how the *Recursive Lexicographical Search* algorithm (RLS) presented in section 3 finds *all* MPE of the simultaneous move leapfrogging game, by systematically examining all feasible ESS γ corresponding to all possible ESRs Γ. We have illustrated this process in Figure 5 for the case where $n = 3$.

The first five rows in Figure 5 describe an example of how a ordering of the states may be constructed, so that the ordering of the stages of the game is satisfied. Each column corresponds to a state point and thus do a digit in a ESS. The top row in Figure 5 presents the stages of the game, the second row presents the digit number in the ESS γ for the case when $n = 3$, and the next three rows show values (c_1, c_2, c) that correspond to each digit (points of the grid for costs are indexed from 0 to 2).

[9] Our numerical solution method for this game is prone to numerical errors for a range of η close to zero.

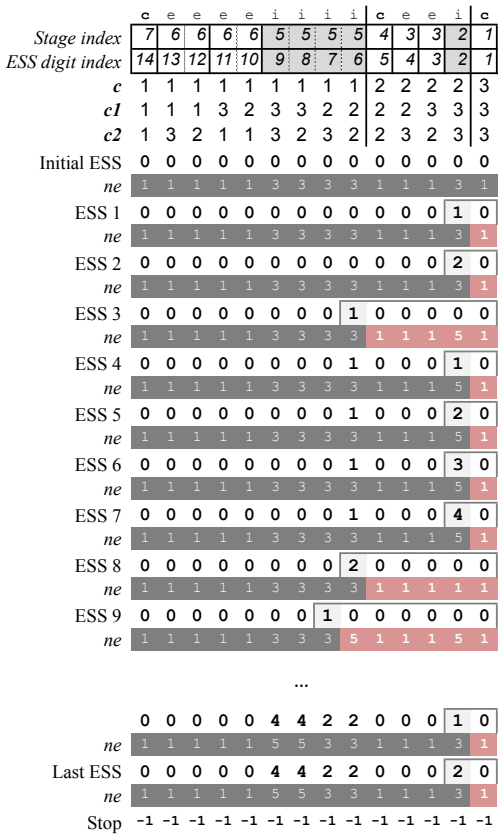

Figure 5: Graphic representation of RLS algorithm.

Note that while there are $N = 14$ state points, there are only $\mathcal{T} = 7$ stages in this game as indicated by the second row so there are multiple ways we can order the state points and still obey the ordering of the stages of the game. Starting from the right, the lowest digit represents the top layer the game with a single point $c_1 = c_2 = c = 2$. The solution in this initial state depend on all subsequent points of the state space, whereas the opposite is the case for the endgame where $c_1 = c_2 = c = 0$. In fact, the digits of the ESS are ordered to preserve the directionality such that the solution at any given point depends on points to the left, but not on points to the right.

Moving from right to left, the ESS γ then contains four points corresponding to second the highest level of the game ($c = 1$ in Figure 5), and the rest of the blocks representing other layers. The left-most block contains $n^2 = 9$ points and corresponds to lowest layer of the game ($c = 0$ in Figure 5) and includes the end game corner, where $c_1 = c_2 = c = 0$. Thus, lower layers of

the game which affect the values of the higher layers, are positioned to the left in the ESR string. Points within each layer are also organized in accordance to the dependency preserving property. The corner point (where $c_1 = c_2 = c$) is left-most in each layer, the two edges (where $c_1 = c$ and $c_2 = c$) that depend on the corner, are positioned next, and the interior points of the layer (where $c_1 < c$ and $c_2 < c$) dependent on the edges are positioned on the right of each block. Figure 5 marks these blocks within each layer of the game with symbols *c*, *e* and *i* above the ERS string.

We begin with the initial ESS, γ^0 that consist of zeros at all 14 digits and solve for all MPE given this equilibrium selection rule. This corresponds to solving the game and selecting the first equilibrium at each point in the state space. Accordingly we can solve for all MPE given the ESS γ^0 to obtain the number of MPE at each state point, which we collect in the vector vector $ne(\gamma^0)$. The feasibility of each particular ESR string γ can be naturally defined through a set of inequalities on the digits given in Lemma 5, $\gamma_{i,\tau} < ne_{i,\tau}(\gamma_{>\tau})$ to ensure that each of them is not greater than the number of d-subgame equilibria found for the corresponding point in the state space. But as mentioned above, the number of stage equilibria on each layer of the game depends on the equilibria chosen on the lower layers of the game, and thus it is important to acknowledge the recursive nature of feasibility of equilibrium selection rules.

Note that only part of the solution has changed when we one to the next equilibrium selection string. In particular, the solution has only changed in stages that follow the stage where highest digit has changed. We have shaded these states in pink in Figure 5.

As is clear from Figure 5 that if we compare to case with fixed base arithmetic where $K = 5$, the RLS algorithm jumps over huge blocks of infeasible ESS. For example the ninth ESS, γ^8, in the example Figure 5 would correspond the $15625'th$ ESS in base 5-number system, but RLS jumped here in only 9 steps without even checking for feasibility of the remaining ESS.

4.2 Bertrand price and investment game with alternating moves

We now turn to our second example, which is an alternating move version 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 $m_t \in \{1,2\}$ be a state variable that governs which

of the two firms are allowed to undertake an investment at time t, i.e. the value $m_t = 1$ indicates a state where only firm 1 is allowed to invest, and $m_t = 2$ is the state where only firm 2 can invest. We will assume that $\{m_t\}$ evolves as an exogenous two state Markov chain with transition probability $f(m_{t+1}|m_t)$ independent of the other state variables $(c_{1,t}, c_{2,t}, c_t)$.

In the alternating move case, the Bellman equations for the two firms lead to a system of eight functional equations for $(v_{N,j}(c_1,c_2,c,m), v_{I,j}(c_1,c_2,c,m))$ for $j,m \in \{1,2\}$. Below we write out the four Bellman equations for firm 1, but we omit the value functions for firm 2 to save space since they are defined similarly.

$$\begin{aligned}
v_{N,1}(c_1,c_2,c,1) &= r_1(c_1,c_2) + \beta f(1|1)\int_0^c \phi(v_{N,1}(c_1,c_2,c',1), v_{I,1}(c_1,c_2,c',1))\pi(dc'|c) + \\
&\quad \beta f(2|1)\int_0^c ev_1(c_1,c_2,c')\pi(dc'|c) \\
v_{I,1}(c_1,c_2,c,1) &= r_1(c_1,c_2) - K(c) + \beta f(1|1)\int_0^c \phi(v_{N,1}(c,c_2,c',1), v_{I,1}(c,c_2,c',1))\pi(dc'|c) + \\
&\quad \beta f(2|1)\int_0^c ev_1(c,c_2,c')\pi(dc'|c) \\
v_{N,1}(c_1,c_2,c,2) &= r_1(c_1,c_2) + \beta f(1|2)\int_0^c \phi(v_{N,1}(c_1,c_2,c',1), v_{I,1}(c_1,c_2,c',1))\pi(dc'|c) + \\
&\quad \beta f(2|2)\int_0^c ev_1(c_1,c_2,c')\pi(dc'|c) \\
v_{I,1}(c_1,c_2,c,2) &= r_1(c_1,c_2) + \beta f(1|2)\int_0^c \phi(v_{N,1}(c_1,c,c',1), v_{I,1}(c_1,c,c',1))\pi(dc'|c) + \\
&\quad \beta f(2|2)\int_0^c ev_1(c_1,c,c')\pi(dc'|c). \qquad (41)
\end{aligned}$$

where

$$ev_1(c_1,c_2,c) = P_2(c_1,c_2,c,2)v_{I,1}(c_1,c_2,c,2) + [1 - P_2(c_1,c_2,c,2)]v_{N,1}(c_1,c_2,c,2). \qquad (42)$$

Note that $P_2(c_1,c_2,c,1) = 0$, since firm 2 is not allowed to invest when it is firm 1's turn to invest, $m = 1$. A similar restriction holds for $P_1(c_1,c_2,c,c,2)$.

Directional and Non-directional states

This example presents the complication that not all state variable are directional, since the right to move alternates forth and back between their two duopolist firms and thus the two values that m_t can take are non-comparable in the sense there may be a *loop* connecting $m = 1$ and

$m = 2$, i.e. $m = 2$ can be reached with positive probability from $m = 1$ and vice versa. Despite this additional simultaneity, that seemingly complicates the partial ordering of the states, it is still quite simple to find the solution of to the stage game equilibria in this example, despite the additional complication induced by the non-directional state variables. First, the directionality of the remaining state variables, (c_1, c_2, c) is unaffected by the alternation of moves, since changing the timing the movies does change the property that nothing in this model could ever cause the cost to increase to suddenly increase. Second, for a given value of $d = (c_1, c_2, c)$, we can easily solve the game for all values of the non-directional state variable $m \in \{1, 2\}$. In particular, it can be shown that the eight functional equations (given by the four equations for firm 1 given in (41) above and the four equations that are defined similarly for firm 2) at the each stage game $\mathcal{G}(\tau - 1)$, can be solved almost analytically given the solution at the previously calculated stage games $\mathcal{G}(\tau)$ and a deterministic equilibrium selection rule, Γ, that selects the equilibrium to be played at lower level stage games.

Theorem 8 (Solution method for the d-stage games in alternating move leapfrogging game)**.** *Given a fixed equilibrium selection rule* Γ*, solution method for every* d*-subgame in the Bertrand pricing and investment game with alternating moves exists and is guaranteed to find all* d*-subgame MPE for every* $d = (c_1, c_2, c)$.

4.3 Performance of the solution algorithms

Theorems 7 and 8 ensure that the key assumption under the stage recursion and RLS algorithms is satisfied, and thus these methods can be applied for the Bertrand pricing and investments game. When we apply RLS to the simultaneous and alternating move formulation of this game, we find that these infinite horizon games turns out to have a surprisingly rich set of equilibrium outcomes.

Figure 6 displays the computed equilibrium expected profits of the two firms in the Bertrand investment and pricing game with simultaneous moves under deterministic (panel a) and stochastic (panel b) technological improvement. With only 5 points in the grid of the costs, there are around 200 million equilibria in each of the two versions of the game, although the number of distinct payoffs is much larger under stochastic technological improvement (as indicated by the size and color of the dots in Figure 6).

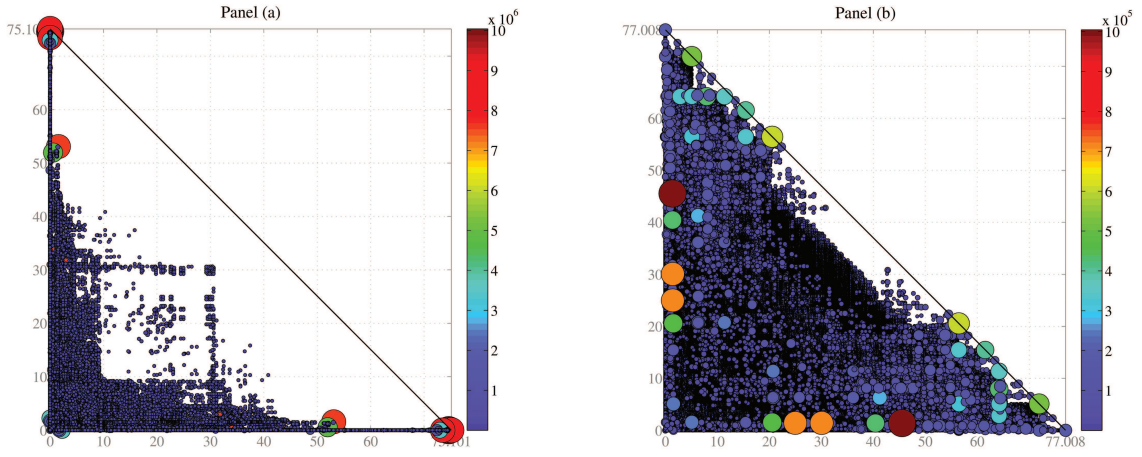

Figure 6: Equilibrium outcomes in Bertrand pricing and investment game with simultaneous moves under deterministic and stochastic technological progress.

Table 1 reports the times spent to compute the MPE equilibria in several specifications of the Bertrand pricing and investment game of different sizes. Comparing the running 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 are increasing highly non-linearly. 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 take a negligible amount of time to loop through all feasible ESR strings compared to the time needed for state recursions.

A comprehensive 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,Rust and Schjerning (2013).

5 Discussion and conclusions

In this paper we introduce a concept of directionality in finite state Markov dynamic games, define the class of directional dynamic games (DDGs) and devise two solution algorithms for the games of this class. The first is state recursion algorithm that finds a single MPE in a DDG given

Table 1: Run times for full solution of the leapfrogging game

	Simultaneous moves			Alternating moves
Number of points in cost grid	3	4	5	5
Total number ESRs	4,782,969	3,948,865,611	$1.7445 \cdot 10^{26}$	$1.7445 \cdot 10^{26}$
Number of feasible ESRs	127	46,707	192,736,405	1
Time used	0.008 sec.	0.334 sec.	45 min.	0.006 sec.

an equilibrium selection rule; the second is the recursive lexicographical search (RLS) algorithm that efficiently finds all feasible equilibrium selection rules, effectively computing all MPE of the game. The RLS algorithm is linear in the number of points in the directional part of the state space, ensuring that negligible time is spent on enumeration of all feasible equilibrium selection rules compared to the time spent on computing the corresponding equilibria.

The class of directional dynamic games we define in this paper appears to be quite large and haves many examples in the existing literature. This is mainly a consequence of the fact that it is sufficient for a Markov game to have just one directional dimension in the state space (so that $D \subset R^1$) to become a DDG. The definition of directional games is silent about the non-directional component of the state space X, although it is implied that this component is in some sense secondary.

The other side of the generality of the class of directional dynamic games is the assumption that a solution method must exist for the smallest indivisible subgames, i.e. d-subgames, having the state space $\{d \times X\} \cup \left(\cup_{t=\tau+1}^{\mathcal{T}} S_t \right)$, where X enters under the cartesian product, see (12). Therefore, it is likely that state recursion and RLS methods can not be applied for a large fraction of directional dynamic games which have numerous or complicated non-directional components — simply due to intractability of their d-subgames. Yet, there are several extensions of the DDG class where the RLS method of finding all MPE can be useful, provided that the atomic subgames are solvable.

First, the already mentioned case of finite horizon Markov games. For these games, the directional component D always contains time that can be thought of as a state variable $t = 0, .., T < \infty$.

Then whether RLS is applicable is determined by whether it is possible to find all equilibria within the set of continuation strategies in each time period t subgame given the value functions from a selected equilibrium in the subgame starting from $t+1$. This will most probably be decided by the complexity of the non-directional components X that are the states of the original game. Nevertheless, all finite horizon dynamic games belong to the class of directional games.

Second, it is possible that a non-directional Markov game with finite state space can be transformed to the DDG by rearranging the dimensions of the state space. For instance, Example 3 in Section 2 presented in the left panel of Figure 1 has a loop in the directional component of the state space, and yet after adding a second dimension and relabeling the points so that the points of the loop have the same value on the directional dimension, and differ only in the non-directional dimension, the game can be assign to the DDG class. Whether RLS is applicable is again decided by whether it is possible to find equilibria in the points in the loop simultaneously, as they form a d-subgame in the transformed example.

Third, it is not only the definition of DDGs that is silent about the non-directional component of the state space — none of the arguments in the paper impose restrictions on X apart from the assumption that the solution method for d-subgames must exist. This implies among other things that, for example, X does not necessarily have to be finite, and provided that the assumption is satisfied, state recursion can be applied to the games with finite directional component D and continuous state variables in X. If it can be shown that the number of d-subgame equilibria is finite, then RLS is applicable and all MPE of such game can be computed.

Forth, even if the key assumption on the existence of the solution method for d-subgame is not satisfied, but instead a method for finding *some* equilibria in d-subgames is available, stage recursion and RLS algorithms can be quite helpful. Here a prominent link to path-following homotopy methods can be established. Even though in general homotopy approach is not well suited for the finite state DDGs with multiple equilibria like the leapfrogging game due to numerous bifurcations along the equilibrium correspondence (see Figure 7 for equilibrium correspondence in the Bertrand pricing and investments game with alternating moves), it may be quite helpful in finding equilibria in d-subgames, and thus be used together with stage recursion and RLS. From our Bertrand pricing and investment game example we conjecture that multiplicity of MPEs in

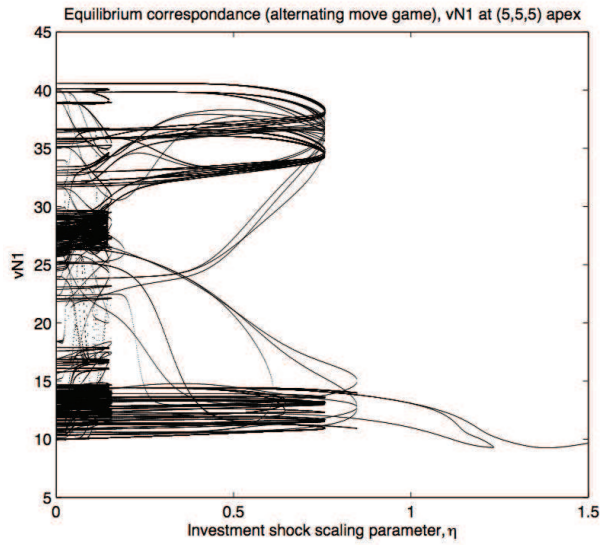

Figure 7: Equilibrium correspondence, alternating move game: expected profit of the firm 1 by different values of cost shock η.

general dynamic games to a large extent may be due to discreteness of equilibria chosen at different stages. RLS naturally accounts for all such combinations of equilibria selected in different subgames, and if the conjecture is true, to large extent accounts for the bifurcations along the equilibrium correspondence. In this case, homotopy can be very efficient in finding d-subgame equilibria that would have much less bifurcations along their equilibrium correspondences.

When the available solution method for the d-subgames is fast and reliably computes all the equilibria in the generalized d-subgames, stage recursion algorithm nested by the RLS algorithm are very efficient to produce all MPE in the directional dynamic games. Yet, this solution method is subject to the curse of dimensionality that may originate from both the number of points in the directional component of the state space k_1 and the upper bound K of the number of equilibria in d-subgames. The total number of equilibrium selection rules is equal to the quantity of numbers with k_1 digits in K-base arithmetics, and thus equal to K^{k_1}. Even though RLS algorithm spends negligible amount of time on the search of feasible ESR strings, in the case when there are in fact many MPE, the solver for each d-subgame has to be run every time, making the total solution runtime increase exponentially.

References

[1] Audet, C., Hansen, P., Jaumard, B., and Savard, G. (2001) "Enumeration of All Extreme Equilibria of Bimatrix Games" *SIAM Journal of Scientic Computing* **23-1** 323–338.

[2] Bellman, R. (1957) *Dynamic Programming* Princeton University Press.

[3] Benoit, Jean-Pierre and Vijay Krishna (1985) "Finitely Repeated Games" *Econometrica* **53-4** 905–922.

[4] Berninghaus, S., W. Güth and S. Schosser (2012) "Backward Induction or Forward Reasoning? An Experiment of Stochastic Alternating Offer Bargaining" Jena Economic Research Papers 2012-041.

[5] Besanko, D. and U. Doraszelski and Y. Kryukov and M. Satterthwaite (2010) "Learning-by-Doing, Organizational Forgetting, and Industry Dynamics" *Econometrica* **78-2** 453 – 508.

[6] Borkovsky, R. N., U. Doraszelski, and Y. Kryukov (2010) "A User's Guide to Solving Dynamic Stochastic Games Using the Homotopy Method" *Operations Research* **58-4** 1116–1132.

[7] Daskalakis, Costis and Paul Goldberg and Christos Papadimitriou (2013) "Computing a Nash Equilibrium is PPAD-Complete" forthcoming, *SIAM Journal of Computing*

[8] Datta, R. (2010) "Finding all Nash equilibria of a finite game using polynomial algebra" *Economic Theory* **42** 55–96.

[9] Doraszelski, Ulrich and Juan Escobar (2010) "A Theory of Regular Markov Perfect Equilibria in Dynamic Stochastic Games: Genericity, Stability, and Purification" forthcoming, *Theoretical Economics.*

[10] Doraszelski, Ulrich and Mark Satterthwaite (2010) "Computable Markov-Perfect Industry Dynamics" *Rand Journal of Economics* **41-2** 215–243.

[11] Fudenberg, D. and D. Levine (1983) "Subgame-Perfect Equilibria of Finite- and Infinite-Horizon Games" *Journal of Economic Theory* **31-2** 251–268.

[12] Haller, H. and R. Lagunoff (2000) "Genericity and Markovian Behavior in Stochastic Games" *Econometrica* **68-5** 1231–1248.

[13] Hörner, Johanes, Takao Sugaya, Sartoru Takahashi, and Nicolas Vielle (2011) "Recursive Methods in Discounted Stochastic Games: An Algorithm for $\delta \to 1$ and a Folk Theorem" *Econometrica* **79-4** 1277–1318.

[14] Iskhakov, Fedor, Bertel Schjerning and John Rust (2013) "The Dynamics of Bertrand Price Competition with Cost-Reducing Investments" manuscript, Georgetown University.

[15] Judd, Kenneth, Renner, Philipp and Schmedders, Karl (2012) "Finding all pure-strategy equilibria in games with continuous strategies" *Quantitative Economics* **3-2** 289–331.

[16] Kuhn, H.W. (1953) "Extensive games and the problem of information" in H.W. Kuhn and A.W. Tucker (eds) *Contributions to the Theory of Games I* Princeton University Press, Princeton NJ 193–216.

[17] Maskin, E. and J. Tirole (1988) "A Theory of Dynamic Oligopoly, I: Overview and Quantity Competition with Large Fixed Costs" *Econometrica* **56-3** 549-569.

[18] Maskin, E. and J. Tirole (2001) "Markov Perfect Equilibrium I. Observerable Actions" **100** 191–219.

[19] Nakaya, Y. and Oishi, Y. (1998) "Finding All Solutions of Nonlinear Systems of Equations Using Linear Programming with Guaranteed Accuracy" *Journal of Universal Computer Science,* **4-2** 171–177.

[20] Pakes, A. and P. McGuire (1994) "Computing Markov-perfect Nash equilibria: Numerical implications of a dynamic differentiated product model" *RAND Journal of Economics* **25** 555–589.

[21] Pakes, A. and P. McGuire (2001) "Stochastic algorithms, symmetric Markov perfect equilibrium, and the curse of dimensionality" *Econometrica* **69** 1261–1281.

[22] Roth, A. and M. F. Balcan and A. Kalai and Y. Mansour (2010) "On the Equilibria of Alternating Move Games" *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms.*

[23] Rubinstein, A. (1982) "Perfect Equilibrium in a Bargaining Model" *Econometrica* **50-1** 97–109.

[24] Rust, J. (1987) "Optimal Replacement of GMC Bus Engines: An Empirical Model of Harold Zurcher" *Econometrica* **55-5** 999–1033.

[25] Shapley, Lloyd S. "Stochastic games." *Proceedings of the National Academy of Sciences of the United States of America* **39.10 (1953)**: 1095.

[26] Selten, R. (1965) "Spieltheoretische Behandlung eines Oligopolmodells mit Nachfrageträgheit" *Zeitschrift für die gesamte Staatswissenschaft* **121** 301–324.

[27] Selten, R. (1975) "Re-examination of the perfectness concept for equilibrium points in extensive games" *International Journal of Game Theory* **4** 25–55.

[28] Smirnov, Vladimir and Andrew Wait (2013) "Innovation in a generalized timing game" manuscript, University of Sydney.

[29] Yamamura, Kiyotaka and Tsuyoshi Fujioka (2003) "Finding all solutions of nonlinear equations using the dual simplex method" *Journal of Computational and Applied Mathematics* **152-1** 587–595.

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 (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 (3) can not 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. Supposing the contrary. 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 (Join of pairwise consistent partial orders of D).

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, the $\mathcal{N}$ operator never returns the empty set, reducing the number of vertices remaining in $D_j(\mathcal{G})$ as j increases. It follows 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 ($\mathcal{D}$ is a DAG if DAG recursion terminates with no descendants after final step). 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 $\mathcal{D}_{\mathcal{T}+1} = \emptyset$.

Proof. Necessity follow from the proof of Lemma 2. To show sufficiency, imagine the contrary that $\mathcal{D}_{\mathcal{T}} \neq \emptyset$ 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 (Stage 1 subgame).

Proof. From equation (11) 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 $\mathcal{G}$. □

Lemma 4

Proof. This result follows easily since at the terminal stage of the DDG $\mathcal{T}$ the continuation game for each stage game $\mathcal{SG}_{\mathcal{T}}(d)$ is empty for $d \in D_{\mathcal{T}}$, so the set of feasible Markovian continuation strategies for each stage game at stage $\mathcal{T}$, $\Sigma(\mathcal{SG}_{\mathcal{T}}(d))$, coincide with the set of feasible Markovian strategies for the end game, $\Sigma(\mathcal{G}_{\mathcal{T}}(d))$. This implies that $\mathcal{SG}_{\mathcal{T}}(d) = \mathcal{G}_{\mathcal{T}}$, $d \in D_{\mathcal{T}}$, establishing equation (18). The second equation (19) follows from the fact that $\mathcal{T}$ is the terminal stage, so the set of all continuation strategies for $\mathcal{SG}_{\mathcal{T}}(d)$ is the same as the set of all feasible strategies for $\mathcal{G}_{\mathcal{T}}(d)$. □

Theorem 2 (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 a MPE of the stage game $\mathcal{SG}_{\tau}(d)$ in the restricted class of continuation strategies. However, by definition, a continuation strategy is a MPE strategy in the stage τ_1 subgame $\mathcal{G}_{\tau+1}$. It follows that $e(\mathcal{SG}_{\tau}(d))$ is a 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 a MPE for the full d-subgame $\mathcal{G}_{\tau}(d)$, since if it wasn't it would have to fail to be a 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 (11) 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 a MPE at each $s \in \Omega_{\tau+1}$, and the fact that it is also a MPE for the stage game $\mathcal{SG}_\tau(d)$ implies that it is also must be a MPE strategy for each $s \in (d \times X)$. Thus, $e(\mathcal{SG}_\tau(d))$ is a MPE strategy at each point $s \in \Omega_\tau(d)$. Sincce this is the state space for the d-subgame $\mathcal{G}_\tau(d)$, it follows that $e(\mathcal{SG}_\tau(d))$ must be a MPE of $\mathcal{G}_\tau(d)$.

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

$$e(\mathcal{G}_\tau(d))(s) = \begin{cases} e(\mathcal{G}_\tau(d))(s) & \text{if } s \in (d \times X) \text{ and } d \in D_\tau \\ e(\mathcal{G}_{\tau+1})(s) & \text{otherwise.} \end{cases} \tag{43}$$

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 a 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 a 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 14 leads to a recursively defined MPE for each stage τ stage game $\mathcal{SG}_\tau$, $\tau = (1, \ldots, \mathcal{T})$. By Theorem 2, these MPE also constitute MPE of the stage τ subgames $\mathcal{G}_\tau$, $\tau = (1, \ldots, \mathcal{T})$. However by Lemma 3 we have $\mathcal{G}_1 = \mathcal{G}$, so it follows that $e(\mathcal{G}_1) = e(\mathcal{G})$, i.e. the state recursion algorithm has computed a MPE of the DDG $\mathcal{G}$ by computing MPE for a total of

$$N = \sum_{\tau=1}^{\mathcal{T}} n_\tau \tag{44}$$

d-stage games of the game $\mathcal{G}$. By Lemma 3 we have $\mathcal{G}_1 = \mathcal{G}$, so it follows that $e(\mathcal{G}_1) = e(\mathcal{G})$. Thus, it follows that the state recursion algorithm has computed a MPE of the DDG $\mathcal{G}$. □

Lemma 6 (Feasibility of $\mathcal{S}(\gamma)$)

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. □

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

Proof. The proof is by mathematical induction. The base of the induction is the bottom layer of the leapfrogging game where $c = 0$. In this case the system of Bellman equations (40) is greatly simplified because no further technological improvement is possible. It is simplified even further in the corner and the edges of the game where investments by one or both firms don't change their future production cost. This makes it possible to solve Bellman equations analytically or with simple numerical procedures, as described in Appendix B. The numerical method described there is guaranteed to find all equilibria among continuation strategies in d-stage games where $d = (c_1, c_2, 0)$. By Theorem 2 these equilibria constitute MPE equilibria in the corresponding d-subgames. The induction step is the following. Assume all layers below the layer given by the value of c are solved. Then we show that d-subgames in the layer c given by $d = (c_1, c_2, c)$ can also be solved. This follows from the solution method described in Appendix B which is are again applied in a sequence prescribed by the ordering of states within the layer. When $\eta = 0$, the solution algorithm is guaranteed to find all equilibria in every d-stage as fixed points in the second order best response function in a particular state (c_1, c_2, c) and taking into account the solutions on the lower layers. Again, applying Theorem 2 we conclude that found equilibria constitute the MPE equilibria in every d-subgame on the layer c. □

Theorem 8 (Solution method for the d-stage games in the alternating move leapfrogging game).

Proof. The proof is analogous to the proof of Theorem 8, with solution method for the d-stage games are presented in Appendix C. □

B Solving the Simultaneous Move Pricing and Investment Game

B.1 $(\mathcal{T})$-End Game, $(c_1, c_2, c) = (0,0,0)$ Corner

The simplest "end game" end corresponds to the state $(0,0,0)$, i.e. when the zero cost absorbing state has been reached and both firms have adopted this state-of-the-art production technology. In the absence of random *IID* shocks $(\varepsilon_0^i, \varepsilon_1^i)$ corresponding to investing or not investing, respectively, neither of the firms would have any further incentive to invest since we assume there is no depreciation in their capital stock, and they have both already achieved the lowest possible state-of-the-art production technology. When there are idiosyncratic shocks affecting investment decisions, there may be some short term reason (e.g. a temporary investment tax credit) that would induce one or both of the firms to invest, but such investments would be purely idiosyncratic unpredictable events with no real strategic consequence to their opponent, since the opponent has already achieved the minimum cost of production and thus, there is no further possibility of leap frogging its opponent. In this zero-cost, zero-price, zero-profit, absorbing state the equations for the value functions $(v_{N,j}, v_{I,j})$ can be solved "almost" analytically.

$$\begin{aligned} v_{N,j}(0,0,0) &= r_j(0,0) + \beta P_{\sim j}(0,0,0)\phi(v_{N,j}(0,0,0), v_{I,j}(0,0,0)) \\ &+ \beta[1 - P_{\sim j}(0,0,0)]\phi(v_{N,j}(0,0,0), v_{I,j}(0,0,0)) \\ &= r^i(0,0) + \beta\phi(v_{I,j}(0,0,0), v_{N,j}(0,0,0)) \end{aligned} \tag{45}$$

where $P_{\sim j}(0,0,0)$ is a shorthand for firm i's opponent's probability of investing,

$$P_{\sim j}(0,0,0) = \frac{\exp\{v_{I,\sim j}(0,0,0)/\eta\}}{\exp\{v_{N,\sim j}(0,0,0)/\eta\} + \exp\{v_{I,\sim j}(0,0,0)/\eta\}}. \tag{46}$$

Due to the fact that $(0,0,0)$ is an absorbing state, it can be easily shown that the value of investing, $v_{I,j}(0,0,0)$, is given by

$$v_{I,j}(0,0,0) = v_{N,j}(0,0,0) - K(0), \tag{47}$$

which implies via equation (46) that

$$P_{\sim j}(0,0,0) = \frac{\exp\{-K(0)/\eta\}}{1+\exp\{-K(0)/\eta\}}. \tag{48}$$

Thus, as $\eta \to 0$, we have $P_{\sim i}(0,0,0) \to 0$ and $v_{N,j}(0,0,0) = r_i(0,0)/(1-\beta)$, and in the limiting case where the two firms are producing perfect substitutes, then $r_i(0,0) = 0$ and $v_{N,j}(0,0,0) = 0$. For positive values of η we have

$$v_{N,j}(0,0,0) = r_i(0,0) + \beta\phi(v_{N,j}(0,0,0), v_{N,j}(0,0,0) - K(0)). \tag{49}$$

This is a single linear equation with one unknown, and we can easily express the solution as

$$v_{N,j}(0,0,0) = \frac{r_i(0,0) + \beta\phi(0, -K(0))}{1-\beta}$$

Note that symmetry property for $r^i(0,0)$ implies that symmetry also holds in the $(0,0,0)$ end game: $v_{N,1}(0,0,0) = v_{I,2}(0,0,0)$ and $v_{I,1}(0,0,0) = v_{I,2}(0,0,0)$.

B.2 $(\mathcal{T}-1)$-Stage Game, $(c_1,0,0)$ and $(0,c_2,0)$ Edges

The next simplest end game state is $(c_1,0,0)$. This is where firm 1 has not yet invested to attain the state-of-the-art zero cost plant, and instead has an older plant with a positive marginal cost of production c. However firm 2 has invested and has attained the lowest possible marginal cost of production 0. In the absence of stochastic shocks, in the limiting Bertrand case, it is clear that firm 1 would not have any incentive to invest since the investment would not allow it to leap frog its opponent, but only to match its opponent's marginal cost of production. But doing this would unleash Bertrand price competition and zero profits for both firms. Therefore for any positive cost of investment $K(0)$ firm 1 would choose not to invest, leaving firm 2 to have a permanent low cost leader position in the market and charge a price of $p = c_1$.

In the case with stochastic shocks, just as in the $(0,0,0)$ endgame analyzed above, there may be transitory shocks that would induce firm 1 to invest and thereby match the 0 marginal cost of production of its opponent. However this investment is driven only by stochastic *IID* shocks and not by any strategic considerations, given that once the firm invests, it will generally not be in much better situation than if it had not invested (that is, even though $r_1(0,0) > r_1(c,0)$, both of

these will be close to zero and will approach zero as $\eta \downarrow 0$). In the general case where $\eta \geq 0$ we have

$$\begin{aligned} v_{N,1}(c_1,0,0) &= r_1(c_1,0) + \beta\phi(v_{N,1}(c_1,0,0), v_{I,1}(c_1,0,0)) \quad (50) \\ v_{I,1}(c_1,0,0) &= r_1(c_1,0) - K(0) + \beta\phi(v_{N,1}(0,0,0), v_{I,1}(0,0,0)). \quad (51) \end{aligned}$$

Note that the solution for $v_{I,1}(c_1,0,0)$ in equation (51) is determined from the solutions of $v_{N,1}$ and $v_{I,1}$ to the $(0,0,0)$ endgame in equations (45) and (47) above. Substituting the resulting solution for $v_{I,1}(c_1,0,0)$ into the first equation in (51) results in a nonlinear equation with a single unique solution $v_{N,1}(c_1,0,0)$ that can be computed by Newton's method.

The probability that firm 1 will invest, $P_1(c_1,0,0)$ is given by

$$P_1(c_1,0,0) = \frac{\exp\{v_{I,1}(c_1,0,0)]/\eta\}}{\exp\{v_{I,1}(c_1,0,0)]/\eta\} + \exp\{v_{N,1}(c_1,0,0)]/\eta\}} \quad (52)$$

We now turn firm 2. In the $(c_1,0,0)$ end game, firm 2 has no further incentive to invest since it has achieved the lowest possible cost of production. However in the presence of random cost shocks (i.e. in the case where $\eta > 0$), firm 2 will invest if there are idiosyncratic shocks that constitute unpredictable short term benefits from investing that outweigh the cost of investment $K(0)$. But since this investment confers no long term strategic advantage in this case, the equations for firm 2's values of not investing and investing, respectively, differ only by the cost of investment $K(0)$. That is,

$$v_{I,2}(c_1,0,0) = v_{N,2}(c_1,0,0) - K(0). \quad (53)$$

The probability that firm 2 invests in this case, $P_2(c_1,0,0)$ is given by

$$P_2(c_1,0,0) = \frac{\exp\{-K(0)/\eta\}}{1+\exp\{-K(0)/\eta\}} \quad (54)$$

since firm 2 has achieved the lowest possible cost of production and its decisions about investment are governed by the same idiosyncratic temporary shocks, and result in the same formula for the probability of investment as we derived above in equation (48) for the $(0,0,0)$ endgame.

The equation for $v_{N,2}(c_1,0,0)$ is more complicated however, due to the chance that firm 1 might invest, $P_1(c_1,0,0)$. We have

$$\begin{aligned} v_{N,2}(c_1,0,0) = r_2(c_1,0) &+ \beta P_1(c_1,0,0)\phi(v_{N,2}(0,0,0), v_{I,2}(0,0,0)) \\ &+ \beta[1-P_1(c_1,0,0)]\phi(v_{N,2}(c_1,0,0), v_{N,2}(c_1,0,0) - K(0)). \quad (55) \end{aligned}$$

Using the solution for $v_{N,2}(c_1,0,0)$ and $v_{I,2}(c_1,0,0))$ in equation (51) above, these solutions can be substituted into equation (52) to obtain the probability that firm 1 invests, and then this probability can be substituted into equation (55) to obtain a unique solution for $v_{N,2}(c_1,0,0)$

$$v_{N,2}(c_1,0,0) = \frac{r_2(c_1,0) + \beta P_1(c_1,0,0)\phi(v_{N,2}(0,0,0), v_{I,2}(0,0,0)) + \beta[1-P_1(c_1,0,0)]\phi(0,-K(0))}{1-\beta[1-P_1(c_1,0,0)]}$$

Finally the value of investing $v_{I,2}(c_1,0,0)$ is given by equation (53).

The value functions in the $(0,c_2,0)$ end game can be derived in a complete analogous way. In this part of the game, firm 1 is now the low cost leader and firm 2 is the high cost follower. We first derive the value functions for the cost follower $v_{I,2}(0,c_2,0)$ and $v_{N,2}(0,c_2,0)$. Using the implied investment probabilities , $P_2(0,c_2,0)$, we can derive the value functions for firm 1, $v_{I,1}(0,c_2,0)$ and $v_{N,1}(0,c_2,0)$.

It is not hard to see that the symmetry condition holds in the $(c,0,0)$ and $(0,c,0)$ end game: $v_{N,j}(c,0,0) = w_{\sim j}(0,c,0)$, and $v_j(c,0,0) = v_{\sim j}(0,c,0)$, $j = 1,2$.

B.3 $(\mathcal{T}-2)$-Stage Game, $(c_1,c_2,0)$ Interior Points

The final case to consider is the end game where both firms have positive marginal costs of production, c_1 and c_2, respectively. We begin by showing how to solve the equations for the values to firm 1 of not investing and investing, respectively, which reduce to

$$\begin{aligned} v_{N,1}(c_1,c_2,0) &= r_1(c_1,c_2) + \beta P_2(c_1,c_2,0)\phi(v_{N,1}(c_1,0,0), v_{I,1}(c_1,0,0)) \\ &\quad + \beta[1-P_2(c_1,c_2,0)]\phi(v_{N,1}(c_1,c_2,0), v_{I,1}(c_1,c_2,0)) \\ v_{I,1}(c_1,c_2,0) &= r_1(c_1,c_2) - K(0) + \beta P_2(c_1,c_2,0)\phi(v_{N,1}(0,0,0), v_{I,1}(0,0,0)) \\ &\quad + \beta[1-P_2(c_1,c_2,0)]\phi(v_{N,1}(0,c_2,0), v_{I,1}(0,c_2,0)). \end{aligned} \tag{56}$$

Given the equation for $v_{I,1}(c_1,c_2,0)$ in equation (56) depends on known quantities on the right hand side (the values for $v_{N,1}$ and $v_{I,1}$ inside the ϕ functions can be computed in the $(0,0,0)$ and $(0,c_2,0)$ end games already covered above), we can treat $v_{I,1}(c_1,c_2,0)$ as a linear function of P_2 which is not yet "known" because it depends on $(v_{N,2}(c_1,c_2,0), v_{I,2}(c_1,c_2,0))$ via the identity:

$$P_2(c_1,c_2,0) = \frac{\exp\{v_{N,2}(c_1,c_2,0)/\eta\}}{\exp\{v_{N,2}(c_1,c_2,0)/\eta\} + \exp\{v_{I,2}(c_1,c_2,0)/\eta\}}. \tag{57}$$

We write $v_{I,1}(c_1, c_2, 0, P_2)$ to remind the reader that it can be viewed as an implicit function of P_2: this is the value of $v_{I,1}$ that satisfies equation (56) for an arbitrary value of $P_2 \in [0,1]$. Substituting this into the equation for $v_{N,1}$, the top equation in (56), there will be a unique solution $v_{N,1}(c_1, c_2, 0, P_2)$ for any $P_2 \in [0,1]$ since we have already solved for the values $(v_{N,1}(c_1,0,0), v_{I,1}(c_1,0,0))$ in the $(c,0,0)$ end game (see equation (51) above). Using these values, we can write firm 1's probability of investing $P_1(c_1, c_2, 0)$ as

$$P_1(c_1, c_2, 0, P_2) = \frac{\exp\{v_{I,1}(c_1, c_2, 0, P_2)/\eta\}}{\exp\{v_{N,1}(c_1, c_2, 0, P_2)/\eta\} + \exp\{v_{I,1}(c_1, c_2, 0, P_2)/\eta\}}. \tag{58}$$

Now, the values for firm 2 $(v_{N,2}(c_1, c_2, 0), v_{I,2}(c_1, c_2, 0))$ that determine firm 2's probability of investing in equation (57) can also be written as functions of P_1 for any $P_1 \in [0,1]$. This implies that we can write firm 2's probability of investing as a function of its perceptions of firm 1's probability of investing, or as $P_2(c_1, c_2, 0, P_1)$. Substituting this formula for P_2 into equation (58) we obtain the following fixed point equation for firm 1's probability of investing

$$P_1 = \frac{\exp\{v_{I,1}(c_1, c_2, 0, P_2(c_1, c_2, 0, P_1))/\eta\}}{\exp\{v_{N,1}(c_1, c_2, 0, P_2(c_1, c_2, 0, P_1))/\eta\} + \exp\{v_{I,1}(c_1, c_2, 0, P_2(c_1, c_2, 0, P_1))/\eta\}}. \tag{59}$$

By Brouwer's fixed point theorem, at least one solution to the fixed point equation (59) exists. Further, when $\eta > 0$, the objects entering this equation (i.e. the value functions $v_{N,1}(c_1, c_2, 0, P_2)$, $v_{I,1}(c_1, c_2, 0, P_2)$, $v_{N,2}(c_1, c_2, 0, P_1)$, and $v_{I,2}(c_1, c_2, 0, P_1)$ and the logit choice probability function P_2) are all C^∞ functions of P_2 and P_1, standard topological index theorems be applied to show that for almost all values of the underlying parameters, there will be an odd number of separated equilibria.

Figure 1 plots the equilibria computed by plotting the second order best response function in equation (59) against the 45 degree line. We see that firm 1 is the low-cost leader with a substantially lower marginal cost of production than firm 2. In the mixed strategy equilibrium, firm 1 invests with probability 0.484, whereas the firm 2, the high cost follower, invests with probability 0.82. Thus, the high cost follower has a significantly higher chance of leap frogging its rival to attain the position of low cost leadership. This leadership is permanent (unless the firms happen to simultaneously invest) since by assumption, the production technology has reached the

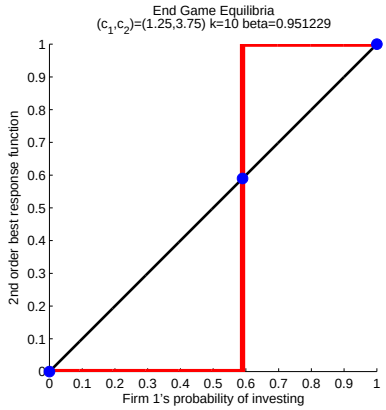

Figure 8: End game equilibria

zero marginal cost absorbing state and there can be no further future improvements in production cost.

To safe space we will not elaborate further on how fixed points are found in the $(c_1, c_2, 0)$ endgame. The solution strategy is complete analogous to how we solve for the equilibria at higher levels of the game, and we will therefore return to this below. As we will show below, the best response functions can be characterized by solution to a second order polynomial, allowing us to express the equilibrium solution almost analytically.

B.4 Solving the $(\mathcal{T}-3), (\mathcal{T}-4), ..., 1$-Stage Games

With the end game solutions in hand, we are now ready to proceed to discuss the solution of the full game. To solve the full game, i.e. the functional equations in (40), it is helpful to rewrite them in the following way,

$$v_{N,1}(c_1, c_2, c) = r_1(c_1, c_2) + \beta[P_2(c_1, c_2, c)H_1(c_1, c, c) + (1 - P_2(c_1, c_2, c))H_1(c_1, c_2, c)] \quad (60)$$

$$v_{I,1}(c_1, c_2, c) = r_1(c_1, c_2) - K(c) + \beta[P_2(c_1, c_2, c)H_1(c, c, c) + (1 - P_2(c_1, c_2, c))H_1(c, c_2, c)] \quad (61)$$

where the function H_1 is given by

$$H_1(c_1, c_2, c) = (1 - \pi(c|c)) \int_0^c \phi(v_{N,1}(c_1, c_2, c'), v_{I,1}(c_1, c_2, c')) f(c'|c) dc' + \pi(c|c)\phi(v_{N,1}(c_1, c_2, c), v_{I,1}(c_1, c_2, c)), \quad (62)$$

where $\pi(c|c)$ is the probability that a cost-reducing innovation will not 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. We assume that the support of $f(c'|c)$ is in the interval $[0,c)$, as indicated also by the the interval of integration in equation (62).

For completeness, we present the corresponding equation for firm 2 below.

$$v_{N,2}(c_1,c_2,c) = r_2(c_2,c_1)+\beta[P_1(c_1,c_2,c)H_2(c,c_2,c)+(1-P_1(c_1,c_2,c))H_2(c_1,c_2,c)] \quad (63)$$

$$v_{I,2}(c_1,c_2,c) = r_2(c_2,c_1)-K(c)+\beta[P_1(c_1,c_2,c)H_2(c,c,c)+(1-P_1(c_1,c_2,c))H_2(c_1,c,c)] \quad (64)$$

where the function H_2 is given by

$$H_2(c_1,c_2,c) = (1-\pi(c|c) = \int_0^c \phi(w_2(c_1,c_2,c'),v_{I,2}(c_1,c_2,c'))\,f(c'|c)dc' + \pi(c|c)\phi(v_{N,2}(c_1,c_2,c),v_{I,2}(c_1,c_2,c)), \quad (65)$$

We will assume initially *deterministic* equilibrium selection rules, i.e. a function that picks out one of the set of equilibria in each possible state of the game, (c_1,c_2,c). Given an equilibrium selection rule, we can assume that the integral term in equation (62) is "known" at this stage of the game. This is because we can structure a recursive algorithm for solving the game by starting with the end game solution and recursively solving the equilibria and value functions for positive values c' that are less than the current value c that we are computing. Then for each $c' < c$, the value functions $v_{N,j}(c_1,c_2,c')$ and $v_{I,j}(c_1,c_2,c')$ will be "known" for all (c_1,c_2) in the rectangle $R(c') = \{(c_1,c_2)|c' \le c_1 \le \overline{c}, c' \le c_2 \le \overline{c}'\}$. Since however, there may be multiple equilibria at lover levels of the game, the integral obviously depend on which of the equilirbia that are played when $c' < c$. This is how the equilibrium selection rule at "lower cost nodes" of the game tree (i.e. at states (c_1,c_2,c') with $c' < c$) affect the set of possible equilibria at each node (c_1,c_2,c).

Since integral term in equation (62) is "known" at this stage, we can use a similar to the strategy to the one we used to solve the value functions $(v_{N,j},v_{I,j})$ $j = 1,2$ in the end game. As before, we first solve for the equilibrium at the (c,c,c) corners, then at the (c_1,c,c) and (c,c_2,c) edges, and finally at the interior nodes of the game.

B.5 Corners and Edges

In the (c,c,c) corner, where all firms have invested and have the state-of-the-art production technology in place, there is no further incentive for either firm to invest. If we set the arguments (c_1,c_2,c) to (c,c,c) in equation (60) for w_1, and similarly in equation (61) for v_1, we deduce that

$$v_{I,1}(c,c,c) = v_{N,1}(c,c,c) - K(c) \tag{66}$$

$$v_{I,2}(c,c,c) = v_{N,2}(c,c,c) - K(c) \tag{67}$$

We can then substitute equation (66) into equation (60) and solve a simple linear equation in $v_{N,1}(c,c,c)$. With $v_{N,1}(c,c,c)$ at hand we can easily compute $v_{I,1}(c,c,c)$. By analogous operations we obtain the value functions for firm 2, $v_{N,2}(c,c,c)$ and $v_{I,2}(c,c,c)$.

We now move to the (c_1,c,c) edge in the state space lattice. If we set the arguments (c_1,c_2,c) to (c_1,c,c) in equation (61) it is easy to see that $v_{I,1}(c_1,c,c)$ is uniquely determined from the solutions at corner, $v_{N,1}(c,c,c)$ and $v_{I,1}(c,c,c)$. Substituting the resulting solution for $v_{I,1}(c_1,c,c)$ into equation in (60) results in a single nonlinear equation with a unique solution $v_{N,1}(c_1,c,c)$ that can be computed by Newton's method. Once $v_{N,1}(c_1,c,c)$ is known we can easily compute $P_1(c_1,c,c)$.

With these solutions at hand, it is straight forward to compute the (c_1,c,c) value functions of firm 2. Since firm 2 that has already invested in the state-of-the-art production technology, investing again will not change it's marginal cost of production. Again, it is easy to see that the equations for firm 2's values of not investing and investing, respectively, differ only by the cost of investment, $K(c)$

$$v_{I2}(c_1,c,c) = v_{N,2}(c_1,c,c) - K(c) \tag{68}$$

The solution for $P_1(c_1,c,c)$ and $v_{I,2}(c_1,c,c)$ given by equation (68), can be substituted into equation (63) to obtain a unique solution for $v_{N,2}(c_1,c,c)$. Using the solution for $v_{N,2}c_1,c,c)$ we can then compute $v_2(c_1,c,c)$ from by equation (68) and derive the implied investment probabilities, $P_2(c_1,c,c)$.

As in the end game, the value functions in the (c,c_2,c)game can be derived in a completely analogous to the (c_1,c,c)game by switching firm indices. Since, firm 1 is now the low cost leader and firm 2 is the high cost follower, we first derive value functions for firm 2.

The value of investing, $v_{I,2}(c,c_2,c)$, is again uniquely determined from the solutions at corner. The value of not investing, $v_{N,2}(c,c_2,c)$, is solved using newtons method. Using the implied investment probabilities, $P_2(c,c_2,c)$, we then derive the value functions for firm 1 using that $v_{I,1}(c,c_2,c) = v_{N,1}(c,c_2,c) - K(c)$.

B.6 Equilibrium Solutions at Interior (c_1,c_2,c) Nodes

In the $(c_1,c_2,0)$ end game mentioned above, either of the two firms will have incentive to invest in equilibrium if $K(c)$ is below a critical threshold, and thus may thus leapfrog their opponent to become the low cost leader forever. But at this higher level of the game, (c_1,c_2,c), where the state of the art has not yet reached it's absorbing state, the coordination between the two firms is dynamic and much richer: If one firm leapfrogs its opponent, the game does not end, but rather the firms must anticipate additional leapfrogging and cost reducing investments in the future.

In the (c_1,c_2,c) interior nodes, both firms have not yet invested in the current state of the art production technology, and there is therefore still room for strategic investment for each of the two firms. The best responses for each firm, therefore depends crucially on the investment probability of the opponent. Thus, the main complication of solving the game at the interior nodes of the game, is that the value functions,.that are solutions to the functional equations in (40), must be solved simultaneously with determining the equilibrium decision rules.

Following the procedure we used to solve for equilibria in the $(c_1,c_2,0)$ end game, the the set of all equilibria for the investment "stage game" at state (c_1,c_2,c) can be computed by finding all fixed points to the following "second order best response function" for firm 1:

$$P_1 = \frac{\exp\{v_{I,1}(c_1,c_2,c,P_2(c_1,c_2,c,P_1))/\eta\}}{\exp\{v_{N,1}(c_1,c_2,c,P_2(c_1,c_2,c,P_1))/\eta\} + \exp\{v_{I,1}(c_1,c_2,c,P_2(c_1,c_2,c,P_1))/\eta\}}. \quad (69)$$

Depending on the rule we choose to select among the possible equilibria in each state (c_1,c_2,c) (and similarly the selection rule for equilibria at all feasible points in the state space (c_1,c_2,c') with $c' < c$) we can construct a wide variety of equilibria for the overall game. The restriction is that any equilibrium selection rule must be such that the functional equations for equilibrium (see equations (60) and (61) above) are satisfied. The following steps are used to solve for the set of all equilibria at each state point (c_1,c_2,c) in the full Bertrand/investment game.

1. For each $P_1 \in [0,1]$ we compute the value functions $(v_{N,2}(c_2,c_1,c,P_1), v_{I,2}(c_2,c_1,c,P_1)$ representing *firm 2's* values of not investing and investing in state (c_1,c_2,c), respectively, by solving the system (63) and (64) for each $P_1 \in [0,1]$.

2. Compute firm 2's "best response", i.e. its probability of investing, $P_2(c_1,c_2,c,P_1)$, in response to its perception of firm 1's probability of investing, P_1, via the equation

$$P_2(c_1,c_2,c,P_1) = \frac{\exp\{v_{I,2}(c_1,c_2,c,P_1)/\eta\}}{\exp\{v_{N,2}(c_1,c_2,c,P_1)/\eta\} + \exp\{v_{I,2}(c_1,c_2,c,P_1)/\eta\}}. \tag{70}$$

   using the value functions for firm 2 computed in step 1 above.

3. Using firm 2's best response probability, P_2, calculate the value functions $v_{N,1}(c_1,c_2,c,P_2)$ and $v_{I,1}(c_1,c_2,c,P_2)$ representing *firm 1's* values of not investing and investing in state (c_1,c_2,c), respectively, by solving the system (60) and (61).

4. Using the values for firm 1, compute firm 1's probability of investing, *the second order best response function* for firm 1, and search for all fixed points in equation (69).

B.7 Polynomial representation of the best response functions

Once the game is solved at lover cost states, the solution strategy is almost analogous to the $(c_1,c_2,0)$ end game. Holding fixed the opponents investment probability, $P_2 = P_2(c_1,c_2,c)$, it is clear from equation (61) that none of the terms on the right hand side of (61) depend on $v_{I,1}(c_1,c_2,c)$. Moreover, since $H_1(c,c,c)$ and $H_1(c,c_2,c)$ are both previously computed from the edges of the game, the value of investing can be expressed as a linear function of P_2 with "known" coefficients.

The value of not investing, $v_{N,1}(c_1,c_2,c,P_2)$, is non-linear in P_2, since $H^1(c_1,c_2,c)$ that appears on the right hand side of (60) is also a function of $v_{I,1}(c_1,c_2,c)$. It is useful is rewrite equation (60) to explicitly emphasize the dependence on $v_{N,1}(c_1,c_2,c)$

$$v_{N,1}(c_1,c_2,c) = A_1(P_2) + B(P_2)\phi(v_{N,1}(c_1,c_2,c), v_{I,1}(c_1,c_2,c))$$

where

$$\begin{aligned} A_1(P_2) &= r_1(c_1,c_2)+\beta(1-\pi(c|c))\int_0^c \phi(v_{N,2}(c_1,c_2,c'),v_{I,2}(c_1,c_2,c'))\,f(c'|c)dc \\ &\quad +\beta(H_1(c_1,c,c)-p(c)\int_0^c \phi(v_{N,2}(c_1,c_2,c'),v_{I,2}(c_1,c_2,c'))\,f(c'|c)dc)P_2 \\ B(P_2) &= \beta(1-p(c))(1-P^2) \end{aligned}$$

It is clear $v_{N,1}(c_1,c_2,c)$ is nonlinear equation in P_2, but both A_1 and B_1 are linear functions of P_2 with coefficients that are known at this stage of the game. Hence we can summarize the functional equations (60) and (61) as

$$\begin{aligned} v_{N,1}(c_1,c_2,c) &= A_1(P_2)+B(P_2)\phi(v_{N,1}(c_1,c_2,c),v_{I,1}(c_1,c_2,c)) \\ A_1(P_2) &= a_{10}+a_{11}P_2 \\ B(P_2) &= b(1-P_2) \\ v_{I,1}(c_1,c_2,c,P_2) &= c_{10}+c_{11}P_2 \end{aligned}$$

The linearity of A, B and $v_{I,1}(c_1,c_2,c)$ turns out to be very useful when deriving the analytical solution for $v_{N,1}(c_1,c_2,c)$, the best response functions and equilibrium investment probabilities. If we express the value functions in probability space, and substitute in the expressions for $v_{I,j}$, A_i and B it is easy to see that (the inverse best) response functions for firm 1 and firm 2 is easily found as the solution to a second order polynomial

$$\underbrace{D_{10}-\eta b\ln P_1+\eta\log\left(\frac{P_1}{1-P_1}\right)}_{D_{10}(P_1)}+\underbrace{(D_{11}+\eta b\ln P_1)}_{D_{11}(P_1)}P_2+D_{12}P_2^2 = 0 \qquad (71)$$

$$\underbrace{D_{20}-\eta b\ln P_2+\eta\log\left(\frac{P_2}{1-P_2}\right)}_{D_{10}(P_1)}+\underbrace{(D_{21}+\eta b\ln P_2)}_{D_{21}(P_2)}P_1+D_{22}P_1^2 = 0 \qquad (72)$$

where

$$\begin{aligned} D_{i0} &= \alpha_{i0}+(b-1)\,c_{i0} \\ D_{i1} &= a_{i1}+(b-1)\,c_{i1}-c_{i0}b \\ D_{i2} &= -bc_{i1} \end{aligned}$$

That is, for a given value of the best response of firm 1, P_1, we can compute the implied value of P_2 by solving (71) with respect to P_2. The roots corresponds to the abscissa values of firm 1's best response function and we have thus expressed the *inverse best response function* , $P_2 = f_1^{-1}(P_1)$ as the solution to a second order polynomial. A similar equation exists for firm 2 and in a similar way we can solve for $P_1 = f_2^{-1}(P_2)$ as the roots to (72). For each of these polynomials there is maximum two roots and the inverse best response functions is described by real roots of these equations which are inside the unit interval.

When $\eta > 0$, standard topological index theorems can be applied to show that for almost all values of the underlying parameters, there will be an odd number of separated equilibria. In this case our algorithm searches for fixed points on the inverse of the second order best response function using a combination of bisections and successive approximations. When $\eta \to 0$, the results of Harsanyi (1973) as extended to dynamic Markovian games by Doraszelski and Escobar (2009) show that η serves as a "homotopy parameter" and for sufficiently small η the set of equilibria to the "perturbed" game of incomplete information converge to the limiting game of complete information. Rather than using the homotopy approach, we found we were able to directly solve for equilibria of the problem in the limiting pure Bertrand case where $\eta = 0$.

When there are no idiosyncratic shocks to investment, i.e. when $\eta = 0$, the best response functions have bang-bang solutions, where the best response functions jumps discontinuously from zero to one at the indifference points. Finding the discontinuity points using a gradient based method or successive approximations is out of the question, and the use of a bracketing algorithm is a daunting task, as it requires repeatedly numerical solution of value functions and best response functions to locate multiple discontinuities. Instead we can solve directly for the threshold values of the opponents investment probability that make firm j indifferent between investing and not investing, which can be found as the solution to the second order polynomials in (71) in (72). This is robust, fast and is always guaranteed to find all equilibria.

C Solving the Alternating Move Pricing and Investment Game

Changing the order of moves from an alternating to simultaneous fashion, introduces a new state *non-directional* state variable $m \in \{1,2\}$ that covers which one of the firms that has the right to

move. As mentioned in section 4, the partial ordering of the directional component, $d = (c_1, c_2, c)$ is unaffected by changing the order of moves, and thus with respect to the state variables c_1, c_2 and c, the state recursion precedes in exactly the same manner. First we solve $(\mathcal{T})$-end game, which is the $(c_1, c_2, c) = (0,0,0)$ corner, we then move to solving the $(c_1, 0, 0)$ and $(0, c_2, 0)$ edges in the bottom layer of the state space pyramid, i.e. the $(\mathcal{T}-1)$-stage game. Given the solution at corners and edges we can solve the for the $(c_1, c_2, 0)$ interior points in the $(\mathcal{T}-2)$-stage game. This recursion completes the lower layer where the state of the art has reached its absorbing state $c = 0$.

For the simultaneous move game we detailed out how the stage game simplifies, when $c = 0$. We will not do that here, but instead derive the solution at corners, edges and interior points for a generic value of c. Note that value functions for $c > 0$ depend on values at lower levels of the game and due to the multiplicity of equilibria there could be many solutions to these value functions at earlier stages. Given a equilibrium selection rule $\Gamma()$ that picks out a particular equilibrium to be played at each point in the state space, we can recursively solve for smaller values of c before larger values of c and assume that value functions at lower levels of the game are "known". It is therefore useful to rewrite the Bellman equations to emphasize that there are parts of the Bellman equation that we consider as "known" at a given point, τ in the state recursion. Specifically, at stage τ' the conditional expectation of the future value function given that technology improves and can be considered "know", because the it depends only on the value functions calculated at previous stages $\tau < \tau'$. Specifically, let $H_1(c_1, c_2, c, m)$ denote the conditional expectation of the future value function for firm 1 given that technology improves and given that it is firm $m \neq m'$ has the turn to invest

$$H_1(c_1, c_2, c, m) = \int [f(m|m)\phi(v_{I,1}(c_1, c_2, c', m), v_{N,1}(c_1, c_2, c', m)) \\ + f(m'|m)[P_2(c_1, c_2, c')v_{I,1}(c_1, c_2, c, m') + (1 - P_2(c_1, c_2, c'))v_{N,1}(c_1, c_2, c', m')]]\pi(dc'|c, dc' < 0), \tag{73}$$

such that the Bellman equations for firm 1 in (41) can be rewritten as

$$\begin{aligned}
v_{I,1}(c_1,c_2,c,1) &= r_1(c_1,c_2) - K(c) + \beta\pi(c|c) f(1|1)\phi(v_{I,1}(c,c_2,c,1), v_{N,1}(c,c_2,c,1)) \\
&\quad + \beta\pi(c|c) f(2|1) ev_1(c,c_2,c) + \beta(1-\pi(c|c)) H_1(c,c_2,c,1) \\
v_{N,1}(c_1,c_2,c,1) &= r_1(c_1,c_2) + \beta\pi(c|c) f(1|1)\phi(v_{I,1}(c_1,c_2,c,1), v_{N,1}(c_1,c_2,c,1)) \\
&\quad + \beta\pi(c|c) f(2|1) ev_1(c_1,c_2,c) + \beta(1-\pi(c|c)) H_1(c_1,c_2,c,1) \\
v_{I,1}(c_1,c_2,c,2) &= r_1(c_1,c_2) + \beta\pi(c|c) f(1|2)\phi(v_{I,1}(c_1,c,c,1), v_{N,1}(c_1,c,c,1)) \\
&\quad + \beta\pi(c|c) f(2|2) ev_1(c_1,c,c) + \beta(1-\pi(c|c)) H_1(c_1,c,c,2) \\
v_{N,1}(c_1,c_2,c,2) &= r_1(c_1,c_2) + \beta\pi(c|c) f(1|2)\phi(v_{I,1}(c_1,c_2,c,1), v_{N,1}(c_1,c_2,c,1)) \\
&\quad + \beta\pi(c|c) f(2|2) ev_1(c_1,c_2,c) + (1-\pi(c|c)) H_1(c_1,c_2,c,2) \qquad (74)
\end{aligned}$$

where

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

A similar set of equations exist for firm 2, which we have omitted to save space, since they are defined similarly.

C.1 Solving the (c,c,c) corner stage games

If we set $c_1 = c_2 =$c in (74) above, it is easy to se that we must have

$$\begin{aligned}
v_{I,1}(c,c,c,1) &= v_{N,1}(c,c,c,1) - K(c) \\
v_{I,1}(c,c,c,2) &= v_{N,1}(c,c,c,2)
\end{aligned}$$

Hence, when $c_1 = c_2 = c$ and both firms have already invested in the current state of the art, there is no further strategic room for investment for other than for pure idiosyncratic reasons. Further, it is easy to verify that $P_2(c,c,c)$ completely drop out of the Bellman equations, and that the value functions becomes a simple equations that we can solve analytically.

$$\begin{aligned}
v_{N,1}(c,c,c,1) &= \frac{A_{N,1}(c,c,c,1)(1-B_{2,2}) + A_{N,1}(c,c,c,2) * B_{2,1}}{1 - B_{1,1} + B_{2,2} - B_{1,1} * B_{2,2} + B_{1,2} * B_{2,1}} \\
v_{N,1}(c,c,c,2) &= \frac{A_{N,1}(c,c,c,2)(1-B_{1,1}) + A_{N,1}(c,c,c,1) * B_{1,2}}{1 - B_{1,1} + B_{2,2} - B_{1,1} * B_{2,2} + B_{1,2} * B_{2,1}}
\end{aligned}$$

where $B_{i,j} = \beta * \pi(c|c)\, f(i|j)$, $A_{N,1}(c,c,c,1) = r_1(c,c) + \beta(1-\pi(c|c))H_1(c,c,c,1) + B_{1,1} * \phi(0,-K(c))$ and $A_{N,1}(c,c,c,2) = r_1(c,c) + \beta(1-\pi(c|c))H_1(c,c,c,2) + B_{1,2} * \phi(0,-K(c))$ are coefficients of the linear system. The value functions for firm 2 can be derived by following similar steps.

C.2 Solving the (c_1,c,c) and (c,c_2,c) edge stage games

This is the case where firm 1 has not yet invested, but firm 2 has already acquired in the state of the art technology. Thus the value function of firm 1, does not depend on whether firm 2 invest or not. It is easy to see this if we set $c_2 = c$ in the the value functions for firm 1, when it is not it's turn to invest. Specifically, we obtain $v_{I,1}(c,c,c,2) = v_{I,1}(c,c,c,2)$. Further, the value of investing, $v_{I,1}(c_1,c,c,1)$, depends only on variables that we have already solved for and can be expressed in closed form

$$\begin{aligned} v_{I,1}(c_1,c,c,1) =& r_1(c_1,c) - K(c) + \beta(1-\pi(c|c))H_1(c,c,c,1) \\ &+ \beta * \pi(c|c)[f(1|1)\phi(v_{N,1}(c,c,c,1), v_{I,1}(c,c,c,1)) + f(2|1) * v_{N,1}(c,c,c,2)] \end{aligned}$$

Substituting the resulting solution for $v_{I,1}(c_1,c,c,1)$ into the equation for $v_{N,1}(c_1,c,c,1)$ results in the nonlinear equation

$$\begin{aligned} v_{N,1}(c_1,c,c,1) = & r_1(c_1,c) * (1+f_1) + H_1(c_1,c,c,1) + f_1 * \beta(1-\pi(c|c))H_1(c_1,c,c,2) \\ &+ \beta * \pi(c|c)[f(1|1) + f(1|2)\, f_1]\phi(v_{N,1}(c_1,c,c,1), v_{1,1}(c_1,c,c,1)) \end{aligned}$$

where $f_1 = \beta * \pi(c|c)\, f(2|1)/(1-\beta * \pi(c|c)\, f(2|2))$. This equation has a unique solution that can be computed by Newtons method. Given the solution of $v_{N,1}(c_1,c,c,1)$ and $v_{I,1}(c_1,c,c,1)$ we can easily compute the remaining two value functions, $v_{N,1}(c_1,c,c,2) = v_{I,1}(c_1,c,c,2)$, in closed form

$$\begin{aligned} & v_{N,1}(c_1,c,c,2) = \\ & \frac{r_1(c_1,c) + \beta(1-\pi(c|c))H_1(c_1,c,c,2) + \beta\pi(c|c)\, f(1|2)\phi(v_{N,1}(c_1,c,c,1), v_{I,1}(c_1,c,c,1))}{(1-\beta\pi(c|c)\, f(2|2))} \end{aligned}$$

Given the values of for firm 1, we obtain it's investment probability by the standard logit formula

$$P_1(c_1,c,c) = \frac{\exp(v_{I,1}(c_1,c,c,1)/\eta)}{\exp(v_{I,1}(c_1,c,c,1)/\eta) + \exp(v_{N,1}(c_1,c,c,1)/\eta)}$$

We now turn to the value functions for firm 2. Note that once $P_1(c_1,c,c)$ is known, the value functions for firm 2 in the (c_1,c,c,m) depends only on $v_{I,2}(c_1,c,c,1)$, $v_{N,2}(c_1,c,c,1)$, $v_{I,2}(c_1,c,c,2)$ and $v_{N,2}(c_1,c,c,2)$ and functions at previously computed stage games. Given equilibrium selection rule, we can treat the latter as single valued "known" entities. Note that from the perspective of firm 2, there is no strategic incentive to invest, since firm two has already acquired the state of the art technology, therefore $v_{I,2}(c_1,c,c,2) = v_{N,2}(c_1,c,c,2) - K(c)$. This simplifies the computation of the value for the low cost leader in the (c_1,c,c) edge games. If we substitute out $v_{N,2}(c_1,c,c,2)$, it has easy to see that the remaining value functions for firm 2 can be found as a solution to a set of linear equations. The solution is

$$v_{I,2}(c_1,c,c,1) = r_2(c_1,c) + H_2(c,c,c,1) + (B_{2,1}\phi(v_{N,2}(c,c,c,2), v_{I,2}(c,c,c,2)) + B_{1,1}v_{I,2}(c,c,c,1)])$$

$$\begin{aligned} v_{N,2}(c_1,c,c,2) = & \frac{r_2(c_1,c)(1+f_1(c_1,c,c)) + H_2(c_1,c,c,2) + f_1(c_1,c,c)H_2(c_1,c,c,1)}{(1-B_{2,2}-B_{2,1}f_1(c_1,c,c))} \\ & + \frac{(B_{2,2}+B_{2,1}f1)\phi(0,-K(c)) + v_{I,2}(c_1,c,c,1)P_1(c_1,c,c)(B_{1,2}+B_{1,1})}{(1-B_{2,2}-B_{2,1}f_1(c_1,c,c))} \end{aligned}$$

$$\begin{aligned} v_{N,2}(c_1,c,c,1) = & \frac{r_2(c_1,c) + H_2(c_1,c,c,1)}{(1-B_{2,2}(1-P_1(c_1,c,c)))} \\ & + \frac{B_{2,1}\phi(v_{N,2}(c_1,c,c,2), v_{N,2}(c_1,c,c,2) - K(c)) + B_{1,1}P_1(c_1,c,c)v_{I,2}(c_1,c,c,1)}{(1-B_{2,2}(1-P_1(c_1,c,c)))} \end{aligned}$$

where $f_1(c_1,c,c) = B_{1,2}(1-P_1(c_1,c,c))/(1-B_{1,1}(1-P_1(c_1,c,c)))$ and $B_{i,j}$ are defined above. Given the values of for firm 2, we obtain it's investment probability by the standard logit formula similar to what we did for firm 1.

In a complete analogous way we can compute the value functions in the (c,c_2,c) game by switching firm indices of value functions, transition probabilities and investment probabilities.

C.3 Equilibrium solutions at (c,c_2,c) interior stage games

In the alternating move game the value of investing can easily be computed, since investment by either of the two firms will imply transition to from (c_1,c_2,c) points of the state space that are already computed at this point. Hence, given the solution of value functions and investment

probabilities at lower levels of the game, and given the solution at the (c,c,c) corners and the (c_1,c,c) and (c,c_2,c) edges of the game, we can compute $v_{I,1}(c_1,c_2,c,m)$ in closed form for both $m=1$ and $m=2$

$$\begin{aligned}
v_{I,1}(c_1,c_2,c,1) &= r_1(c_1,c_2)-K(c)+B_{1,1}\phi(v_{I,1}(c,c_2,c,1),v_{N,1}(c,c_2,c,1) \\
&\quad +B_{2,1}[P_2(c,c_2,c)v_{I,1}(c,c_2,c,2)+(1-P_2(c,c_2,c)v_{N,1}(c,c_2,c,2)] \\
&\quad +(1-\pi(c|c))H_1(c,c_2,c,1) \\
v_{I,1}(c_1,c_2,c,2) &= r_1(c_1,c_2)+B_{1,2}\phi(v_{I,1}(c_1,c,c,1),v_{N,1}(c_1,c,c,1) \\
&\quad +B_{2,2}[P_2(c_1,c,c)v_{I,1}(c_1,c,c,2)+(1-P_2(c_1,c,c)v_{N,1}(c_1,c,c,2)] \\
&\quad +(1-\pi(c|c))H_1(c_1,c,c,2)
\end{aligned}$$

The values of *not* investing, $v_{N,1}(c_1,c_2,c,1)$ and $v_{N,1}(c_1,c_2,c,2)$, depends on the value functions in the interior of the state space as well as the opponents investment probability, $P_2(c_1,c_2,c)$. But if we rearrange $v_{N,1}(c_1,c_2,c,2)$ and substitute back into the equation for $v_{N,1}(c_1,c_2,c,1)$, we obtain a single non-linear equation in $v_{N,1}(c_1,c_2,c,1)$ and $P_2(c_1,c_2,c)$. We write $v_{N,1}(c_1,c_2,c,1,P_2)$ to emphasize that the value function can be viewed as an implicit function of P_2, i.e. the value of $v_{N,1}$ that satisfies the non-linear equation (75) below for an arbitrary value of $P_2\in[0,1]$.

$$v_{N,1}(c_1,c_2,c,1,P_2)=A_1(P_2)+B_1(P_2)\phi(v_{N,1}(c_1,c_2,c,1,P_2),v_{I,1}(c_1,c_2,c,1)) \tag{75}$$

where

$$\begin{aligned}
A_1(P_2) &= (1+f_1(P_2))r_1(c_1,c_2)+(1-\pi(c|c))H_1(c_1,c_2,c,1)+f_1p(c)H_1(c_1,c_2,c,2) \\
&\quad +(B_{2,1}+B_{2,2}f_1(P_2))v_{I,1}(c_1,c_2,c,2)P_2 \\
B_1(P_2) &= B_{1,1}+B_{1,2}f_1(P_2) \\
f_1(P_2) &= \frac{B_{2,1}(1-P_2)}{1-B_{2,2}(1-P_2)}
\end{aligned}$$

Hence, for a given value of P_2, we can easily express $v_{N,1}(c_1,c_2,c,1)$ as the solution to a simple nonlinear equation in $v_{N,1}(c_1,c_2,c,1,P_2)$. This equation can easily be solved using Newtons method since $B_{P_2}\leq 1$ for any $P_2\in[0,1]$. Given the firm 1 value functions $v_{I,1}(c_1,c_2,c,1)$ and $v_{N,1}(c_1,c_2,c,P_2)$, we can compute best response function of firm 1 by the standard logit formula

$$P_1(c_1,c_2,c,P_2)=\frac{\exp(v_{I,1}(c_1,c_2,c,1)/\eta)}{\exp(v_{I,1}(c_1,c_2,c,1)/\eta)+\exp(v_{N,1}(c_1,c,1,P_2)/\eta)}$$

Following similar steps we can obtain a similar equation for firm 2 to obtain $P_2(c_1, c_2, c, P_1)$, i.e. firm 2's best response to firm 1's investment probability, P_1.

Whenever $\eta > 0$, (c_1, c_2, c) stage game equilibria can be found as the fixed point to the second order best response function, i.e. the mapping from P_1 to P_1 that we obtain by substituting the best response function for firm 2, $P_2(c_1, c_2, c, P_1)$ into the best response function for from 1, $P_1(c_1, c_2, c, P_2)$

$$P_1(c_1, c_2, c, P_1) = \frac{\exp(v_{I,1}(c_1, c_2, c, 1)/\eta)}{\exp(v_{I,1}(c_1, c_2, c, 1)/\eta) + \exp(v_{N,1}(c_1, c, 1, P_2(c_1, c_2, c, P_1))/\eta)}$$

When $\eta > 0$ our algorithm searches for fixed points the second order best response function using a combination of bisections and successive approximations. It can be shown that this algorithm is guaranteed to find *all* fixed points on the second order best response function when $\eta > 0$. When $\eta = 0$ we found we were able to directly solve for all mixed strategy equilibria as solution to second order polynomials. We will show this below.

C.4 Polynomial representation of the best response functions

To solve for the equilibrium investment probabilities it is useful to express value functions in choice probability space. Using that $P_1(c_1, c_2, c)$ is uniquely determined by $v_{I,1}(c_1, c_2, c, 1) - v_{N,1}(c_1, c_2, c, 1)$ via the standard logit formula, we can rewrite the problem (75) in choice probability space

$$A_1(P_2) - \eta \log(\frac{1 - P_1}{P_1}) - v_{I,1} + B_1(P_2)(v_{I,1}(c_1, c_2, c, 1) - \eta \log(P_1)) \quad = \quad 0 \qquad (76)$$

where $v_{I,1}(c_1, c_2, c, 1)$ is known at this point. This equation fully describes firm 1's best response as an implicit function of P_2.

If we substitute in the values of $A_1(P_2)$ and $B_1(P_2)$ it can be shown that the resulting expression is a *rational function* of P_2, i.e. an algebraic fraction, $R(P_2; P_1)/Q(P_2)$, where both the numerator and the denominator are polynomials of P_2 (for a given value of P_1). The denominator of this fraction of polynomials is $Q(P_2) = 1 - B_{2,2}(1 - P_2)$ which is never zero since P_2 limited to unit interval and since $B_{2,2} < 0$. Hence, if we want to find the roots of (76) it is sufficient to find the roots of the numerator, $R(P_2; P_1)$ which linear function of P_2

$$R(P_2, P_1) = D_{1,0}(P_1) + D_{1,1}P_2(P_1) = 0 \qquad (77)$$

where

$$
\begin{aligned}
D_{1,0}(P_1) &= (B_{2,1}+(1-B_{2,2}))r_1(c_1,c_2) \\
&+ (B_{2,2}+B_{1,1}+B_{2,1}b_{11}-B_{2,2}B_{1,1}-1)v_{I,1}(c_1,c_2,c,1) \\
&+ \beta(1-\pi(c|c)((1-B_{2,2})H_1(c_1,c_2,c,1)-B_{2,1}H_1(c_1,c_2,c,2)) \\
&- \eta\left[(B_{1,1}+B_{2,1}B_{1,2}-B_{2,2}B_{1,1})\ln P_1+(1-B_{2,2})\ln\left(\frac{1-P_1}{P_1}\right)\right] \\
D_{1,1}(P_1) &= (B_{2,2}-B_{2,1})r_1(c_1,c_2)+B_{2,1}v_{I,1}(c_1,c_2,c,2) \\
&+ (B_{2,2}B_{1,1}-B_{2,2}-B_{2,1}B_{1,2})v_{I,1}(c_1,c_2,c,1) \\
&+ \beta(1-\pi(c|c))(B_{2,2}H_1(c_1,c_2,c,1)-B_{2,1}H_1(c_1,c_2,c,2)) \\
&+ \eta\left[(B_{2,1}B_{1,2}-B_{2,2}B_{1,1})\ln P_1-B_{2,2}\ln\left(\frac{1-P_1}{P_1}\right)\right]
\end{aligned}
$$

When $\eta > 0$ the coefficients of polynomial $R(P_2;P_1)$ are not fixed, but depends on the investment probability of firm 1, P_1. Thus, for given value of the best response of firm 1, P_1, we can compute the implied value of P_2 by solving $R(P_2;P_1)=0$ with respect to P_2. Roots that are located on the unit interval corresponds to the abscissa values of firm 1's best response function and we can there by compute the *inverse best response function for firm 1* as $P_2 = P_1^{-1}(c_1,c_2,c,P_1) = -D_{1,0}(P_1)/D_{1,1}(P_1)$. A similar equation exists for firm 2 and in a similar way can compute $P_1 = P_2^{-1}(c_1,c_2,c,P_2) = -D_{2,0}(P_2)/D_{2,1}(P_2)$. When $\eta > 0$ we can solve for the d-subgame equilibria at (c_1,c_2,c) interior points, either by finding the fixed point of the second order best response function as outlined in the previous subsection, or we could find the fixed point of the inverse second best response function. We can obtain the latter by substituting the inverse best response function for firm 2 in to the inverse best response function of firm 1.

When $\eta = 0$, the best response correspondence is either zero or one except for values of P_2 that makes firm 1 indifferent between investing or not. Clearly, any mixed strategy equilibrium must therefore be located located at these indifference points. We can solve directly for the values of firm 2's investment probability that make firm 1 indifferent between investing and not investing. It is easy to show that the roots of (77) corresponds to exactly this value. This can easily be verified by setting $v_{I,1}(c_1,c_2,c,1) = v_{N,1}(c_1,c_2,c,1)$ in (75) above and finding the root. In fact, the best response function equals $P_1(c_1,c_2,c,P_2) = 1[D_{1,0}+D_{1,1}P_2 > 0]$, where $1[]$ is the indicator function. Note that the coefficients of $R(P_2;P_1)$ do not depend on P_1 when $\eta = 0$, i.e.

$D_{1,0}(P_1) = D_{1,0}$ and $D_{1,1}(P_1) = D_{1,1}$. Hence the root $P_2 = -D_{1,0}/D_{1,1}$, and thus any candidate for a mixed strategy equilibria can be found in closed form.