NBER WORKING PAPER SERIES

A FRAMEWORK FOR APPLIED DYNAMIC ANALYSIS IN I.O.

Ariel Pakes

Working Paper 8024
http://www.nber.org/papers/w8024

NATIONAL BUREAU OF ECONOMIC RESEARCH
1050 Massachusetts Avenue
Cambridge, MA 02138
December 2000

This paper was originally written as an invited lecture to the European Meetings of the Econometric Society in Berlin in 1998. The ideas surveyed in this paper have been developed through interaction with a rather large set of colleagues. Though the list is too long for a single footnote, I owe special thanks to coauthors who have been directly involved in the development of this framework, especially Richard Ericson, Steve Berry, Chaim Fershtman, and Paul McGuire. The views expressed in this paper are those of the author and not necessarily those of the National Bureau of Economic Research.

A Framework for Applied Dynamic Analysis in I.O.
Ariel Pakes
NBER Working Paper No. 8024
December 2000
JEL No. L0, L1, C0

ABSTRACT

This paper outlines a framework which computes and analyzes the equilibria from a class of dynamic games. The framework dates to Ericson and Pakes (1995), and allows for a finite number of heterogeneous firms, sequential investments with stochastic outcomes, and entry and exit. The equilibrium analyzed is a Markov Perfect equilibrium in the sense of Maskin and Tirole (1988). The simplest version of the framework is supported by a publically accessible computer program which computes equilibrium policies for user-specified primitives, and then analyzes the evolution of the industry from user-specified initial conditions.

We begin by outlining the publically accessible framework. It allows for three types of competition in the spot market for current output (specified up to a set of parameter values set by the user), and has modules which allow the user to compare the industry structures generated by the Markov Perfect equilibrium to those that would be generated by a social planner and to those that would be generated by "prefect collusion". Next we review extensions that have been made to the simple framework. These were largely made by other authors who needed to enrich the framework so that it could be used to provide a realistic analysis of particular applied problems.

The third section provides a simple way of evaluating the computational burden of the algorithm for a given set of primitives, and then shows that computational constraints are still binding in many applied situations. The last section reviews two computational algorithms designed to alleviate this computational constraint; one of which is based on functional form approximations and the other on learning techniques similar to those used in the artificial intelligence literature.

Ariel Pakes
Department of Economics
Littauer 117
Harvard University
Cambridge, MA 02138
and NBER
apakes@kuznets.fas.harvard.edu

Most of applied work in Industrial Organization is based on static models. This is in spite of the fact that the issues analyzed often have dynamic implications[1], and is largely due to the fact that the appropriate dynamic model is often quite complex. Indeed realistic dynamic models of even relatively simple market settings typically have to be too complex to admit *analytic* results with much applied content. This is the reason that most of the good theoretical work on the dynamics of markets use "stylized" environments; environments that are designed more for the clarity they allow in developing intuitions than for their realism. Our framework is an attempt to provide a complementary tool for dynamic analysis of more detailed environments. It gives up on the elegance of analytic results entirely, focusing instead on an algorithm that computes and analyzes policies for user-specified market conditions.

The ultimate goal of the framework is to enable the researcher to analyze dynamics in markets that mimic the markets that we actually observe. That is we would like to provide a tool to the applied analyst who knows something about demand, costs, and the nature of both policies and equilibrium in a particular setting, and wishes to either translate that knowledge into its likely implications, or to undertake a substantive numerical analysis of a policy or an environmental change. As we shall see the framework can also be used to check for the robustness of the intuitions developed from the stylized theoretical environments, and to investigate the implications of richer behavioral assumptions than theory has been able to deal with to date. Finally the framework has also been used as a teaching aid; usually to illustrate the possible dynamic effects of traditional policies.

It works out that building a single framework general enough to encompass many of the situations that are important to applied researchers is both too difficult a task, and an inefficient way of proceeding. It is inefficient because the peculiarities of a given setting often enable the use of special computational techniques which cut the burden of computing the needed equilibria significantly. Thus what we have tried to do is provide an "expandable" framework; one whose core is very simple, but which can be (and has been) extended to accommodate more complex market institutions[2].

[1]For an empirical example of the importance of dynamic considerations, see Pakes Berry and Levinsohn's(1993) analysis of the response of the miles per gallon of new car sales to the increase in gas prices in 1973.

[2]There are undoubtedly other frameworks that could be used and expanded in a similar fashion. My goal here is not to survey what is, or could be made, available. It is only to

The framework is presented in the next section. It is designed to accommodate the limited "empirical facts" that we seem to find in virtually all industries. Thus it allows for heterogeneity among firms, idiosyncratic or firm specific outcomes (so it can generate phenomena like rank reversals in the fortunes and sizes of firms), and (simultaneous) entry and exit.

The core version is supported by a publically available program that has separate subroutines to compute the equilibrium for the dynamic analogues to each of three standard static models. One is a differentiated product model. Here equilibrium in the "spot" market for current output is Nash in prices, and firms invest to improve the qualities of their products. The other two models are homogeneous product models; one has differences in marginal costs across firms and the other has differences in capacities. In these models equilibrium in the spot market is Nash in quantities and investment is directed at decreasing marginal cost (in the model with differences in marginal costs) or at increasing capacity (in the model with capacity constraints). For each model the user is allowed to specify values for parameters that set demand, production costs, sunk entry and exit fees, the efficacy of investments, and the discount rate. In addition the user specifies an initial market structure for the analysis. Finally there are additional modules which compute the solutions to a social planner's problem and to a "perfect colluders" problem based on the same parameters, and provide statistics which allow the user to compare these outcomes to those from the Markov Perfect equilibrium.

The core version does limit the set of possible policies and states in ways that rule out many markets of interest. Section 2 of the paper goes on to a discussion of extensions. These extension were originally designed by different authors to allow for particular applied phenomena, but they contain ideas which can be applied more broadly. We review extensions designed to allow for; multiple state variables per firm, collusion, price (or quantity) decisions which have an independent effect on the demand or cost function in future periods (as well as on current profits; examples include models with learning by doing, or with durable or experience goods), non-pecuniary externalities to investment decisions (or "spillovers"), and mergers. Throughout we maintain the assumption of a Markov Perfect equilibrium, so most of the extensions in section 2 simply mimic theoretical developments in the analysis of such equilibria (see the discussion in Maskin and Tirole 1995). That

acquaint the reader with the framework that I have been using.

is by incorporating different state variables and modifying the action space appropriately we can use the Markov Perfect concept in a wide variety of situations.

In describing our framework in section 1 we outline a simple "backward" solution technique for computing its equilibria. Section 3 provides a way of determining the computational burden of the backwards solution technique when applied to any given specification. This section makes it clear that the simple computational algorithm is not sufficiently powerful to enable realistic analysis of many empirically important situations. Consequently section 4 provides a brief outline of how two more powerful computational algorithms can be applied to our framework. One is based on polynomial approximations similar to those discussed in Judd (1999). The other is based upon ideas from the artificial intelligence literature (see Barto, Bradtke, and Singh,1995), and has an interpretation which is similar to that of the economic theory literature on learning (see Pakes and McGuire, forthcoming).

I conclude this introduction by noting three topics that this paper *does not* cover. First there is no publically accessible computer program which computes the extensions to the framework discussed in section 2, and the publically accessible version of our algorithm is computationally inefficient. As a result, though the publically accessible version of the algorithm has been used extensively as a teaching tool, researchers using variants of our framework to analyze more detailed problems have largely built their own computer programs. I should note, however, that the artificial intelligence algorithm described in section 4 is extremely easy to program, and for many problems is several orders of magnitude more efficient than the point-wise techniques that we describe in section 2.

Second, this survey does not pay any attention to the problem of determining which of the alternative institutional structures are relevant for which applied situations. Partly this reflects the spirit of the analysis: those choices ought to be made differently for different situations, and made by those who are familiar with the relevant primitives. On the other hand, there is still an estimation problem (the problem of obtaining a reasonable range of parameter values for a given institutional setting), and there *is* a need for a discussion of currently available estimation techniques. The literature on estimation and testing of I.O. models has grown rapidly in the last few years, and even if I felt comfortable with my reading of it, I could not do justice to it as a section of this paper.

Finally this paper does not consider the problem of computing models with assymetric information. This is an important topic, and various papers that are just appearing make it clear that we should be able to incorporate assymetric information into our, or related, frameworks (see for e.g. Cole and Kocherlakota, 1998, Judd, Yeltkin and Conklin,2000, and Fershtman and Pakes, in process).

1 A Simple Model.

The core version of our model is taken from Ericson and Pakes (1995, henceforth EP), and this section introduces it. In doing so we will also describe the the publically accessible algorithm which computes its equilibrium.

The model has firms investing to explore profit opportunities. Successful investments lead to states where the firm earns more profits. Unsuccessful investments, those that leave the firm behind its competitors both inside and outside the industry, lead to a deterioration of profits and may eventually induce owners to exit.

1.1 The Profit Function.

Profits in any period depend on the quality of the firm's own product or the efficiency of its production technology, as well as on the levels of quality or efficiency of other competing firms. We let i index the values of the firm-specific state variable, and assume i takes on values in the positive integers ($i \in \mathcal{Z}^+$). s_i will be the number of firms at i ($s_i \in \mathcal{Z}^+$), so the vector $s = [s_i;\ i \in \mathcal{Z}^+]$ is the "industry structure" (the number of firms at each different efficiency level). We consider the important extension in which i is a vector below (among other extensions, this enables us to allow firms to differ in both the efficiency of their production technology and in the quality of their product, and to allow product "quality" to have many dimensions).

Profits for a firm at i when the market structure is s are given by $\pi(i, s)$. To derive this profit function from primitives we need to specify:

1. a demand system: this delivers the quantity demanded of each product as a function of i,s and the prices of all products,

2. a cost function: this delivers costs as a function of i, the quantity produced, and possibly factor prices, and

3. an equilibrium assumption: this determines quantities or prices as a function of (i, s).

The core version of the framework uses the simplifying assumption which underlies the standard "static-dynamic" breakdown in teaching IO. It assumes that the distribution of future values of the state of the system (in our notation of (i, s)) conditional on the investments made in the interim do not depend on prices or quantity choices. When this assumption is appropriate changes in current price or quantities affect only current profits. Consequently we can analyze the equilibrium in the "spot market" for current output without even specifying the dynamics of the system. Those dynamics are analyzed at a later stage which begins by substituting the output of the first stage (equilibrium prices or quantities as a function of (i, s)) into the demand and cost functions and computing the implied profit function. The dynamics then analyzes investment, entry, and exit, as a response to the opportunities posed by this profit function. Usually the static or "spot" equilibrium concept is either Nash in prices (in which case quantities are determined by the demand system) or Nash in quantities (in which case prices are determined by that system)[3].

The computational advantage that results from this simplifying assumption is that the Nash equilibrium values for price (or quantity) can be computed knowing only the functional form of demand and costs, and (i, s) (in particular, it does *not* require knowledge of the value function). Formally that solution gives us price as $p(i, s)$ (or quantity as $q(i, s)$), which we then substitute into the demand equation to determine $q(i, s)$ $(p(i, s))$, which in turn allows us to compute profits as $\pi(i, s) = p(i, s)q(i, s) - c[q(i, s)]$, where $c(\cdot)$ is the cost function. That is the profit function can be computed "off line"and simply imported (point-wise) into the part of the program that computes the "dynamic" policies (i.e. entry exit and investment). The cost, of the assumption is that it does not allow us to study environments where, for any of a number of reasons, current price choices have an effect on future states that is independent of any investments made in the interim. We come back to incorporating to these more complicated environments in our discussion of extensions in Section 2.

The core version of our program asks the user to choose among three

[3]Though it is not difficult to think of alternatives that are relevant for particular applied problems. For e.g. Berry, Levinsohn and Pakes, 1999, consider a model in which certain agents choose quantities and others choose prices.

frequently used combinations of (1) to (3) and then calculates the resultant $\pi(i, s)$. The three examples of $\pi(\cdot)$ available in the public program (each is programmed up to a set of parameter values determined by the user)[4] are:

- A differentiated product model where i indexes the quality of a firm's product, equilibrium in the product market is Nash in prices, investment leads to improvements in the "quality" (average utility from) the product, and there is entry and a possibility of exit;
- A homogeneous product market in which equilibrium is Nash in quantities, i indexes inter-firm differences in a constant marginal cost of production, investment improves i, and there is entry and exit;
- A homogeneous product market in which equilibrium is Nash in quantities but i indexes inter-firm differences in capacities, investments increase capacity, and there is entry and exit.

In each case the equilibrium assumption is used to solve for the quantities and prices as a function of state vector, and these are substituted into the profit function to obtain $\pi(i, s)$. [5]

Before concluding we remind the reader that the publically accessible program assumes i is a scalar, and that, conditional on investment decisions, the distribution of future values of the state of the system (of (i, s)) are independent of the choice of prices or quantities. These are strong assumptions and section 2 is primarily concerned with extending the algorithm in a way that allows us to circumvent them. However as we will explain below those extensions require modifications to the publically available program.

[4]To access a description of, and code for, this algorithm (as well as the number of auxiliary programs designed to help analyze its results); either download the need directories directly from my Harvard web page, or FTP to "econ.yale.edu", use "anonymous" as login, and your own username as "password". In the latter case you need to change directory to "pub/mrkv-eqm" and copy all needed files. There is a "read.me" file to start you off.

[5]Examples of applied work that have used these alternatives, are, respectively: Pakes and McGuire (1994), Gowrisankaran and Town(1996), Marcovich(1999)(who computes the solution to an advertising game), Liu (1999), Fershtman and Pakes (2000), and Benkard (2000); EP(1995); and Berry and Pakes(1993) and Gowrisankaran(1999). Older (1998) uses the core version of the program to compute a differentiated product advertising model whose profit function is taken from Sutton (1995).

1.2 Entry, Exit and Investment Decisions.

The public version of the algorithm is broken down into four basic modules. The first calculates $\pi(i, s)$ as discussed above. This then becomes input into the second module which calculates entry, exit, and investment policies. We now outline this second module.

Given $\pi(\cdot)$ an incumbent has two choices. It chooses whether to exit or remain active, and if it remains active it chooses an amount of investment. If it exits it receives a sell-off value of ϕ dollars (and never reappears). If it invests x it incurs a cost of cx and has a probability distribution of improvements in i which is stochastically increasing in x[6].

Thus if we let β be the discount rate, and $pr(i', s'|x, i, s)$ provide the firm's perceptions of the joint probability that its own efficiency in the next period will be i' and the industry structure will be s' conditional on (x, i, s), the Bellman equation which determines the firm's value $[V(i, s)]$ is

$$V(i,s) = \max\{\phi, \pi(i,s) + \sup_{(x \geq 0)} [-cx + \beta \sum V(i',s')pr(i',s'|x,i,s)]\}. \quad (1)$$

The max operator determines if the sell-off value of the firm, our ϕ, is greater than its continuation value (the expression on the right hand side of ϕ). If so the firm shuts down. If not the firm chooses an amount of investment (an $x \geq 0$) which determines the probability distribution of the increment in the firm's state over the period.

We let the increment to that state be

$$\tau_{t+1} \equiv i_{t+1} - i_t,$$

and assume τ can be written as a difference of two independent random variables

$$\tau_t \equiv \nu_t - \zeta_t.$$

[6]Throughout we assume that the response of the state variable to investment is stochastic, rather than deterministic, as in standard models of capital accumulation (for a discussion of the relevance of stochastic versus deterministic models of accumulation, see Pakes,1993). Different forms of the stochastic accumulation assumption have been used in the I.O. literature for some time, probably most often in the patent race literature. For a brief review of that literature, and an interesting application of numerical techniques to investigate the robustness of the conclusions of that literature to the stylized assumptions used in prior analytic treatments, see Doraszelski,2000. Daraszelski's uses different techniques than those we focus on. He considers a continuous time model and numerical approximations similar to those discussed in Judd,1999 (and discussed in section 4 below.)

ν represents the outcome of the firm's investment process and has probabilities given by the family

$$\mathcal{P} = \{\lambda(\cdot|x), x \in \mathcal{R}^+\},$$

which is assumed to be stochastically increasing in x (i.e. $\lambda(\cdot|x_1)$ is better, in the first order stochastic dominance sense, then $\lambda(\cdot|x_2)$ whenever $x_1 > x_2$). ζ is an exogenous random variable with density $\lambda(\zeta)$. Its precise interpretation depends on the structure of the profit function (see below). The distributions of the ν of different firms are independent of one another, but the realization of ζ is common across firms. Both ν and ζ are non negative, integer valued, random variables; $\nu = 0$ with probability one if $x = 0$ (a firm cannot advance without some investment), and $\lambda(0) > 0$ as is $\lambda(0|x)$ for all finite x.

In the publically available version of the program the interpretation of ζ depends on the primitives used for demand and costs. Thus in the differentiated product model ζ is the mean utility of the "outside alternative"; that is the mean utility from the alternative of not purchasing any of the goods marketed in this industry. In the homogeneous goods models it represents either exogenous factor prices, or an index of the demand curve for that good. Note that in either case the fact that ζ exists provides a reason for *positive* correlation between the profits of the different firms in an industry. Since positive realizations of a given firm's ν generate less profits for that firm's competitors, we would not get this positive correlation without the ζ. That is we need the ζ (the aggregate demand and cost changes) to predict the positive correlation among the profits of firms within an industry that exists in many data sets.

The publically available version of the model confines both ν and ζ to take on values of zero or one. So in any given decision period the firm's state can only move up one, stay the same, or move down one in any period. The required probabilities are set at: $\lambda(\nu = 1|x) = [ax/(1+ax)]$ (so $\lambda(\nu = 0|x) = 1 - [ax/(1 + ax)]$), while $\lambda(\zeta = 1) = \delta$. Note that there can be more than one decision period per period observed in any data set, and by changing the number of decision periods per data period we can generate a quite rich set of transition functions for the data to choose between. Of course the discount rate and the profit function need to be adjusted to the length of the model period.

Now let $\hat{s_i}$ be the vector providing the states of the competitors of a firm at state i when the industry structure is s, and $q[\hat{s_i}'|i, s, \zeta]$ provide the firm's perceived probability that the states of its competitors in the next period will be $\hat{s}_i{}'$ conditional on a particular realization for ζ. Then

$$pr(i' = i^*, s' = s^*|x, i, s) = \tag{2}$$
$$\Sigma_\zeta \lambda(\nu = i^* - i - \zeta|x) q[\hat{s}_i{}' = s^* - e(i^*)|i, s, \zeta]\lambda(\zeta),$$

where $e(i)$ is a vector which puts one in the i^{th} slot and zero elsewhere so that $s^* - e(i^*)$ lists the states of next years competitors if $s' = s^*$ and the firm's own future state is i^*. Note that $q[\cdot|i, s, \zeta]$ embodies the incumbent's beliefs about entry and exit.

Substituting equation (2) into (1) we obtain the (Bellman equation for) the value function as

$$V(i, s) = \max\{\phi, \pi(i, s) +$$
$$\sup_{(x \geq 0)} [-cx + \beta \sum V(i + \nu - \zeta, \hat{s}_i + e(i + \nu - \zeta))q[\hat{s}_i|i, s, \zeta]\lambda(\nu|x)\lambda(\zeta)]\}. \tag{3}$$

We still need to specify the entry model. The publically accessible version of the program assumes that there is only one potential entrant a period who pays an amount x_e $(> \beta\phi)$ to enter, and enters one period later at state $\imath^e - \zeta$ with probability $\lambda(\zeta)$ where $i^e \in \mathcal{Z}^+$ is specified by the user. The entrant only enters if the expected discounted value of future net cash flows from entering is greater than the cost of entry. The cost of entry can be specified as either the constant, x_e, or as a random variable which distributes uniformly on $[x_{e,l}, x_{e,u}]$. When random entry costs are used only the potential entrant knows the realization of the entry costs, the other incumbents know only that entry costs will be a random draw from this uniform distribution.[7]

As noted in EP different entry models are easy to accommodate provided the distribution of $i's$ at which entry occurs is fixed over time. That is the "ability" of entrants must progress at the same pace as the "ability" of the outside alternative; else entry would eventually go to zero and stay there.

[7]Random entry costs tend to "smooth out" the value functions of incumbents in the region in which there might be entry of an additional competitor. As a result it can be quite helpful in overcoming convergence problems in computing the equilibrium when such problems occur (see below).

1.3 Dynamic Equilibrium:Characterization Results.

This section reviews the properties of the model's equilibria detailed in EP(1995). The proof that these properties hold use the regularity conditions given in A.1 to A.7 of that article [8].

EP show that a rational expectations MPE exists for our model. In the model firm behavior depends on the perceived distributions of its future competitors, formalized in the transition probabilities, $q[\hat{s}_i{}'|i, s, \zeta]$, and derived as the distribution of $s' - e(i')$ from some distribution of s' given s. Yet the investment, entry, and exit choices generated by these perceptions, together with the known distributions of ν given alternative values of x and the distribution of entry locations, generate an objective distribution of industry structures. A distribution of future industry structures is an equilibrium distribution given the current structure if and only if the investment, entry, and exit decisions which it generates produces an objective distribution of industry structures which is identical with the distribution which generated it (for more detail see the discussion in Star and Ho, 1969).

EP also show that we will only observe firms at a finite set of integer states. We let that set be $\Omega = \{1, ..., K\}$. Also there will never be more than a *finite* number, say $\overline{n}$, of active firms[9]. K is referred to as the dimension of the grid, and $\overline{n}$ as the number of state variables. The fact that they are finite

[8]I should note that no one has reconstructed these proofs for the extensions of the algorithm discussed in the next section. Still there seems not to have been any difficulty in verifying the computational properties discussed below (with the possible exception of ergodicity) for all computed versions of these extensions I am aware of. In addition, as pointed out to us by Gautum Gowrisankaran, the current existence proof should have incorporated random entry and exit costs (see Pakes and McGuire, 1995), though again most of the computed versions of the model suffice without this complication.

[9]Heuristic explanations for these results follow. First the regularity conditions used in EP insure that if a firm's state falls low enough it will exit and this produces the lower bound for the observed i. Those conditions also insure that the value function is bounded. Since the returns to investment are determined by the slope of the value function, the boundedness of that function insures that at a high enough i the firm will stop investing, and if the firm stops investing its i cannot increase. This produces the upper bound to observed states. In general both the upper and lower bounds will depend on the distribution of competitors, but the assumption that the profits can be driven to a sufficiently low number by increasing the number of incumbents insures that there will never be more than a finite number of active firms. Hence there will be a finite number of industry structures and a finite maximum and minimum to the levels of i ever observed.

implies that we need only compute equilibria for (i, s) tuples that satisfy

$$(i, s) \in \Omega \times S, \text{ where } S \equiv \{s = [s_1, ..., s_k] : \sum s_j \leq \overline{n} < \infty\}$$

Note that the number of distinct elements in $\Omega \times S$, or $\#(\Omega \times S)$, is *finite*, so it is possible to compute and store equilibrium policies for each possible (i, s). On the other hand we still need a way of determining what K and $\overline{n}$ are for any given set of primitives.

The heart of the equilibrium is a stochastic process for industry structures [for $\{s_t\}$]. This process is a homogeneous Markov process, *i.e.* if s^t provides the history of industry structures, i.e. $s^t = (s_t, s_{t-1}, ..., s_1)$, then

$$Pr[s_{t+1} = s'|s^t] = Pr[s_{t+1} = s'|s_t] \equiv Q[s'|s_t].$$

EP also prove that the Markov transition kernel for $\{s_t\}$, or $Q[\cdot|\cdot]$, that is associated with each possible equilibrium (and there may be more than one of them) is *ergodic*. This implies that no matter the initial s_0, s_t will, in finite time, wander into a subset of the states, say $R \subset S$ or the recurrent class, and once $s_t \in R$ it will, with probability one, stay in R forever (once in R there is a no probability of communicating outside of R; see figure 1 in EP for a two dimensional illustration).

Consequently if $s_t \in R$ then we can analyze behavior from s_t without knowing anything about the policies outside of R. Further once we know the identities of the points in R we will know whether the s_t of interest (usually its current value) is in R (and it will be if the process has gone on for sufficient time). As discussed below the economics of many applied I.O. problems imply that R will be *much* smaller than S. Thus one way of reducing the computational burden of the algorithm is to find a method that enables us to compute policies for points in R only (see below).

Note that the actual nature of the states in R (e.g. does it include both relatively fractured and relatively concentrated structures?), and the pattern of likely transitions between those states (do we cycle over the divergent types of structures, or are their sudden events that take us more directly from one to another?), depends on the primitives of the model; $\pi(\cdot), \beta, x_e, \phi, \lambda(\cdot)$ *and* $\mathcal{P}$. These in turn, depend on demand patterns, technological opportunities, and the institutional structure of the industry; objects that are likely to vary from problem to problem. Thus, as noted, to actually use this framework for applied work we need some idea of the appropriate primitives and a computational algorithm that allows us to analyze their implications.

1.4 Dynamic Equilibrium: Computation of Policies.

This section outlines the backward solution method provided in Pakes and McGuire (1994; henceforth PM) for computing equilibrium investment, entry and exit policies for our problem. Like all backward solution techniques it is an iterative procedure which starts each iteration with a set of numbers in memory, provides a rule which updates those numbers in the course of the iteration, and checks to see if the updated numbers satisfy a convergence criteria at the end of the iteration. If not the algorithm begins a new iteration[10].

Our algorithm holds estimates of the value function and policies associated with each $(i, s) \in \Omega \times S$ in memory, and uses equation (3), the Bellman equation, to update them. The updating procedure is *synchronous*; i.e. it circles through the points in S in some fixed order and updates all estimates associated with *every* $s \in S$ at each iteration. If the values and policies from successive iterations are the same, then the algorithm is said to have converged. As we explain below when the algorithm converges we will have computed an equilibrium.

Assume temporarily that Ω, the possible values for i, and that $\overline{n}$, or the maximum number of firms ever active, are both known (so $\Omega \times S$ is known). We can rewrite the Bellman equation (equation 3) for each $(i, s) \in \Omega \times S$ as

$$V(i,s) = \max\{\phi, \pi(i,s) - \sup_{x\geq 0}[-cx + \beta \sum_{\nu} w(\nu; i,s)\lambda(\nu|x_1)]\}, \qquad (4)$$

where

$$w(\nu;\ i,s) = \sum_{(\hat{s}_i^{'},\zeta)} V(i+\nu-\zeta, \hat{s}_i^{'} + e(i+\nu-\zeta)|w)q[\hat{s}_i^{'}|i,s,\zeta]\lambda(\zeta), \qquad (4a)$$

and, as before

$$q[\hat{s}_i^{'} = s_i^{*}|i,s,\zeta] \equiv Pr\{\hat{s}_i^{'} = \hat{s}_i^{*}|i,s,\zeta, \text{ and equilibrium policies}\} \ (4b).$$

The term, $w(\nu; i, s)$, provides the expected discounted value of future net cash flow conditional on the current year's investment resulting in a particular value of ν, and the current state being (i, s). It is an expectation because

[10]For good overviews of backward solution algorithms for dynamic programming problems see Bertsekas,1995, and Judd, 1999, chpt.12.

it is constructed by integrating out over the possible outcomes of both the investment strategies of competitors (the $\hat{s_i'}$), and over the outside alternative (the ζ).

Note that the $\{w(\cdot|i,s)\}$ are all an agent needs to know in order to determine both his optimal investment policy, and, given that investment, whether to exit. As a result the equations in (4) make it easy to show how we update at each iteration of our algorithm.

Index a particular iteration's value and policies by a superscript. Each iteration determines a new set of policies and values from the old set that is in memory. In addition to investment, the policies include exit policies for incumbents ($\chi(\cdot)$ will be an indicator which takes on the value of zero if the firm exits) and an entry policy for the potential entrant ($\chi_e(\cdot)$ will be an indicator which takes a value of one if entry occurs). Thus the iteration's goal is to determine $\{V^k(\cdot,\cdot), x^k(\cdot,\cdot), \chi^k(\cdot,\cdot)\ \textit{and}\ \chi_e^k(\cdot,\cdot)\}$ from $\{V^{k-1}(\cdot,\cdot), x^{k-1}(\cdot,\cdot), \chi^{k-1}(\cdot,\cdot),\ \textit{and}\ \chi_e^{k-1}(\cdot,\cdot)\}$.

The iteration circles over the points in S in a predetermined over. If $\overline{n}$ is small relative to K (and it usually is) the points are often stored as vectors that specify the states of active firms (instead of as vectors that specify the number of firms active at each possible state), i.e. they are often stored as vectors $\underline{i} = (i_1, \ldots, i_{\overline{n}})$, with the i_j in their natural order (i.e. $i_j \geq i_{j-1}$) and $i_j > 0$ only if there are j or more firms active[11]. The computational algorithm is easier to understand if we proceed as if the memory has been stored as the $\underline{i}$ vectors (rather than by the implied s), so this section proceeds with that assumption (even though this sacrifices both a bit of elegance and notational consistency with previous work).

Under each $\underline{i}$ we have stored last iterations policies for each $(i, \underline{i})$ and for the potential entrant. We update these policies as follows

- for each incumbent (i.e. for each $i_j > 0$), calculate $w^{k-1}(\cdot|i,\underline{i})$ from the information in memory (by substituting $V^{k-1}(\cdot)$ and $q^{k-1}(\cdot|\cdot)$ into 4a), where $q^{k-1}(\cdot|\cdot)$ is calculated from the policies at iteration $k-1$ for the other incumbents and potential entrant,

- then substitute $w^{k-1}(\cdot)$ for $w(\cdot)$ in (4), and solve the resultant *single agent* optimization problem for the k^{th} iteration's exit and investment

[11]Since we are restricting the policies to be symmetric (or more precisely exchangeable) in the policies of competitors, we can reorder the elements of any vector to this order, and use the policies computed from this vector.

polices by choosing (χ, x) to

$$\max_{\chi\in\{0,1\}}\{[1-\chi]\phi + \chi \sup_{x\geq 0}[\pi(i,\underline{i}) - cx + \beta\sum_{\nu} w^{k-1}(\nu; i, \underline{i})\lambda(\nu|x)]\}. \quad (5)$$

This is done in the following steps:

- Assume the first firm continues ($\chi = 1$) and solve for x^k when $\chi = 1$. Kuhn Tucker theory tells us that neccesary conditions for an x^k to solve this problem are

$$x^k \geq 0, \quad \textit{and}$$

$$x^k\{-c + \beta\sum_{\nu} w^{k-1}(\nu; i, \underline{i})\frac{\partial\lambda(\nu|x)_{x=x^k}}{\partial x}\} = 0.$$

  Second order conditions will be satisfied automatically with the functional form for $\lambda(\cdot|\cdot)$ used in the publically accessible version of our algorithm, but will also have to be checked if more general forms are used.
- Given this x^k, we determine χ^k by substituting x^k for x in equation (5), and then checking whether the implied continuation value is greater than ϕ.

- Next we substitute this (x^k, χ^k) for (x, χ) in (5) and take the value of the resultant expression as $V^k(i, \underline{i})$. That completes the update for the incumbents at $(i, \underline{i})$.
- To find out whether the potential entrant enters, calculate $w^{k-1}(0; i_e, \underline{i} + e(i_e))$ from the information in memory as above. If $w^{k-1}(0; i_e, \underline{i} + e(i_e)) > x_e$, set $\chi_e = 1$; otherwise $\chi_e = 0$.

That completes the updates for an iteration. At the end of this process the iteration calculates a norm in the difference between current iteration and previous iterations values (i.e. $||V^{k-1}(\cdot) - V^k(\cdot)||$ where $||\cdot||$ refers to the sum of squares of the differences) and either stops, or continues, depending on whether the difference is sufficiently small.

If $||V^{k-1}(\cdot) - V^k(\cdot)|| = 0$, then the policies derived from V^k are identical to those derived from V^{k-1}, and each incumbent and potential entrant

- uses, as its perceived distribution of the future states of its competitors, the actual distribution of future states of those competitors, and

- chooses its policy to maximize its expected discounted value of future net cash flow given this distribution of the future of its competitors.

Consequently those values and polices are a Markov Perfect Nash equilibrium (see Star and Ho, 1969).

To complete the description of the algorithm all we need to do is explain how we determine Ω and $\overline{n}$. Prior to the procedure described above we solve the monopoly problem (the problem with $\overline{n} = 1$) with an oversized state space. This is a standard contraction mapping which solves easily and will produce a lowest i at which the firm remains active and a highest i at which the firm invests. The lowest i becomes the 1 in the definition of Ω. Since our regularity conditions insure that a firm that does not invest does not have a positive ν, the highest i *plus* the maximum value of ν with positive probability becomes K. EP(1995) show that if a monopolist would exit at a given state so would a firm facing any amount of competitors, so the lower bound for the monopoly problem is a lower bound for any Nash equilibrium. The upper bound for the monopoly problem need not be the upper bound for a problem that allows for more than one firm, but it has sufficed in every problem we have computed. Nonetheless, the publically available version of the program checks to see if any firm with $i = K$ is actually investing in the equilibria. If it is, it restarts the program with an increased K.

The program determines $\overline{n}$ iteratively. It first calculates a Markov Perfect Nash equilibrium when the maximum number of firms is set arbitrarily to 2 [12]. Then it pushes the arbitrary limit to n up to 3 and does the iterative calculations again starting at $V^0(i_1, i_2, i_3) = V^*(i_1, max(i_2, i_3))$. In principal the program should continue in this fashion until we reach an n so high that whenever there are $n-1$ firm's active there is no possible structure at which an entrant would want to enter. The minimal n at which this occurs (and there will be such an n if the regularity conditions in EP,1995, are satisfied) is the equilibrium $\overline{n}$. What the program actually does is let the user choose a maximum value for $\overline{n}$ and then check whether it in fact meets the equilibrium condition.

[12] In this case we need to modify the program here to instruct it not to evaluate entry if two firms are active. For this and other details see Pakes, Gowrisankaran and McGuire, 1997.

1.5 Notes on the Publically Available Algorithm.

This publically available program is split into four modules, each of which performs a different task. It also comes with a set of auxiliary programs designed to enable the researcher to compare the results from the MP equilibrium to other familiar institutional structures. This section begins with a brief discussion of convergence and uniqueness, then goes on to a description of each of the four modules, and concludes by outlining the auxiliary programs available. Section 3 provides a more detailed discussion of computational efficiency, including methodology for improving the efficiency of the simple algorithm discussed above (see also the related discussions in Judd,1999, and Bertsekas,1995).

1.5.1 Convergence and Uniqueness.

There is no guarantee that the public access program will converge for any given set of primitives. If non-convergence occurs in the public access program when using a deterministic entry cost, the first thing one should try is a problem with the the same value for all primitives but the entry costs, and the entry costs changed to the random entry cost option. Other ways of getting around convergence problems are discussed in PM(1994). Though convergence problems do occur periodically, we have always found that use of one or more of the suggestions in PM has lead the algorithm to converge. Unfortunately the alternative computational strategies PM(1994) suggests for non-convergent problems are not currently programmed as options in the public access program, so if there is good reason for staying with a set of primitives for which the algorithm does not converge, the user will have to go into the program and change some of its features.

There is no guarantee that there is only one equilibrium for a given set of primitives. Indeed to my knowledge there has been very little discussion of sufficient conditions for uniqueness of Markov Perfect Equilibria in any literature[13]. Since part of what we want to do with the program is to analyze responses to alternative possible policies, this lack of a guarantee of uniqueness is troublesome (that is we do not know that the set of policies that we compute and use to evaluate the change are the only policies consistent with the new primitives). On the other hand we have experimented quite a bit

[13]A notable exception is the discussion of uniqueness in Maskin and Tirole's (1987) linear quadratic alternating move game.

with the core version of the algorithm, and never found two sets of equilibrium policies for a given set of primitives (we frequently run the algorithm several times using different initial conditions or different orderings of points looking for other equilibria that might exist). We should emphasythe here that the core version, and indeed most other versions that have been used, all use quite simple functional forms for the primitives of the problem, and multiplicity of equilibrium may well be more likely when more complicated functional forms are used. Of course most applied work suffices with quite simple functional forms.

1.5.2 The Four Modules of the Basic Program.

As noted, the first module of the program calculates profits. The profit function is treated as an input in the second module. This module uses the algorithm discussed in the last subsection to calculate entry, exit and investment policies for each $s \in S$. These policies serve as the input to the third and fourth modules; modules which contain programs designed to characterize the equilibrium. The third module simulates a sequence of industry structures from a user specified initial condition, and then provides a set of statistics that describe the industry generated by that simulation. The descriptive statistics contain data on entry, exit, the number of firms active, lifespans and values of those firms, prices and investment policies, and concentration ratios (see PM,1994, for details)[14]. The fourth module again simulates from a user specified initial condition, but this time simulates repeatedly from the same initial condition and calculates the mean and variance of the distribution of consumer and producer surplus generated by the simulations.

1.5.3 Auxiliary Programs In the Public Access Version.

The auxiliary programs enable the user to compare the Nash equilibria to two other institutional structures. One is the institutional structure that would be generated by a benevolent social planner. The planner determines all prices and investments in each period, as well as entry and exit decisions, to maximize the expected discounted value of consumer surplus. The four modules for this auxiliary program do precisely the same things as the

[14]It is not difficult to add to the tables outputted by this procedure. Indeed, we often require our students to do this as part of a classroom exercise.

four modules for the program that computes the MPE. A comparison of the output from these programs to the output from the MP programs tells us whether there is any room at all for institutional change with improves on the Nash equilibrium, and suggest properties that the improvement might have. Note that our planner need not worry about incentive problems, and receives all the information needed costlessly.

The second set of auxiliary programs computes the policies and simulates industry structures and welfare results for a "perfect" cartel. The cartel acts as if it is a monopolist who determines all pricing, entry, exit and investment decisions to maximize producer surplus (the expected discounted value of the sum of the net cash flows accruing to the different firms). The collusive results tell us just how profitable collusion could be in this industry, and what it might look like if it occurred. Note that this program pays no attention to the problem of determining whether (and how) this form of collusion could be enforced (a topic we return to in the extension section).

2 Extensions to the Core Version.

There are a large number of potentially interesting extensions. I suffice with a short review of those that have been computed, and some straightforward extensions of obvious applied importance. I discuss each of the extensions separately and concentrate almost entirely on the conceptual problems that arise in analyzing them. It is clear, however, that applied researchers might want to combine two or more of them, and that once we move to applied work computational, as well as conceptual problems, will arise. Computational issues are discussed in the next section.

Recall that the core version considers models where there is no independent effect of current price (or quantity) choices on future states, and that this enabled us to mimic the standard textbook separation between static and dynamic analysis (we can first consider the profits from a Nash equilibrium in prices or quantities conditional on the "state" variables of the problem, and then consider the entry exit and investment decisions that determine the evolution of those states). We begin with problems that arise when we allow for multiple states per firm in that framework. Only then do we move on to investigate situations in which the price (or quantity) decisions are inherently dynamic. Here we split the discussion into a subsection on collusion and one on dynamic costs (e.g. learning by doing, or adjustment costs) and/or dynamic demand (e.g. models with experience or durable goods). In the first subsection current price (or quantity) choices are determinants of the price (or quantity) strategies of a firm's competitors in future periods. In the second subsection current price or quantity choices help set the level of cost or demand functions in the future. Less is known about alternative ways of incorporating non-pecuniary externalities to investment ("spillovers" in the terminology of the R&D literature), and/or merger activity, into the framework. In the last two subsections we discuss how this could be (and in the case of merger activity, has been) done.

2.1 Multiple States Per Firm.

Additional state variables are often needed before we can get an adequate approximation to the real world institutions that determine the interactions of interest. Whether or not the additional state variables generate new conceptual problems depends on how the new state variables are added.

Exogenously Evolving States.

This includes

- states that evolve as exogenous Markov processes
- states that differentiate between different (time invariant) types of either firms or products.

The addition of such states are used in a variety of ways, and I am not aware of any new conceptual problems they generate.

In applied work movements in industry wide demand, technology, or factor prices, are often incorporated by adding exogenous Markov processes whose realizations take on the same value for all firms.

Gowrisankaran and Town (1997) extend the core version to allow for two different types of firms, for profit and not for profit hospitals, and then investigate the likely impacts of health policy changes on the evolution of the hospital industry. Not for profit hospitals care about quality, as well as profits, and have certain tax advantages[15]. Using (largely) estimated parameters, they investigate the impact of policy changes that were designed

[15]One could also differentiate between different types of goods, rather than by different types of firms. For e.g. Benkard(2000) differentiates between types (sizes) of the aircraft developed and marketed.

primarily to change the conditions determining demand for hospital services (changes in Medicare reimbursement rates, insuring uninsured patients, and taxing not for profit hospitals) on supply conditions in that industry. That is, by endogenizing the impact of the demand side changes on market structure (through induced changes in investment, entry, and exit), they are able to analyze the *equilibrium* impacts of the policy changes on patient welfare. They find that the induced changes in market structure can have large (and largely unintended) impacts on patients.

States that Evolve Endogenously.

The conceptual problems that arise in incorporating multiple endogneously evolving states per firm differ with the details needed for the application at hand. Two extensions which are important for applied work are

- allowing for products that are differentiated by more than one characteristic, and
- allowing for firms that market more than one product.

Virtually all of the recent characteristic based demand (and cost) studies use a multidimensional characteristic space and find that there is more than one characteristic that is important to consumers. To dynamize these models we need to formulate investment and entry processes that operate on a multidimensional space. I have computed several such models for the new car market, focusing on the trade off between miles per gallon and car size, and on how various policies can effect it. The problems that arose were more related to the parameterization of the model than to the nature of the equilibrium, and were largely familiar from the literature on technological change. For e.g. does investment in cost saving technology shift down the entire cost surface or are its effects limited to a neighborhood of a particular value for the "(size, mpg)" couple? Similarly what is the choice set of vehicles for the new entrant, and how do the costs of entry vary with the characteristics of the vehicle chosen?

To my knowledge the framework has not been used to study markets with multi-product firms. I include this in my description of models with multiple states per firm solely because of its importance to applied work (especially to the analysis of the impacts of merger activity and to the analysis of the relative incentives to develop new products). Once we allow for multi-product firms it is natural to allow incumbents (as well as new entrants) to introduce new products. Even in the simplest such case, say when a single quality dimension is the sole source of differentiation among products and entry occurs at a random draw from an exogenously specified distribution of these qualities, when incumbents (as well as the potential entrant) can introduce new products, there will be many possible equilibria.

The researcher will then have to formulate mechanisms that direct the algorithm to computing one of them. These can be as simple as allowing random draws to determine the order of moves. More realistically we could let the firm for whom entry generates the highest increment to its value have a higher probability of moving first (taking explicit account of differences in the investment required before launch due, say, to differences in the products produced in the past). More detailed empirical knowledge, say on the determinants of the probability of being a first mover, would be extremely useful in this context.

The issues get even more interesting when we allow for multiprouct firms *and* more than one characteristic per firm, for then we get into the discussion of the development of products by competing incumbents in related but different market segments (which dates back at least to e.g. Schmalensee's (1978) paper on cereal's and Judd's (1985) comment). Interestingly many of the developments required to empirically obtain and analyze static profit functions in settings with multi-product firms producing products with many characteristics has already been done. A good example is Petrin(2000) who analyzes the introduction of the Mini-van and shows, *inter alia* how the fact that Ford had a disproportionate share of the station wagon market made it a less likely candidate to introduce the Mini-van than Chrysler (it was Chrysler that went ahead with development expenditures and eventually introduced the first Mini-van, even though the idea was put forth by a Ford engineer).

2.2 Equilibria In Which Price or Quantity Choices are Determined by Dynamic Incentives.

In these cases the logic underlying how we construct the equilibria (the discussion of section 1.4) must be altered, so we discuss these changes in more detail. We begin with models which allow current price (or quantity) strategies to be functions of previous choices of these controls (as in models of collusion). Then we consider cases where the current level of the demand or costs function depends explicitly on past price or quantity choices.

Collusion.

Most of the theoretical work on collusion and price wars assumes identical firms and an unchanging environment. Though these assumptions help clarify what determines when a collusive agreement can be enforced and hence when it breaks down, they often make the existing collusion models unappealing to applied researchers who are trying to understand the workings of particular industries. In many applied problems we are simply not willing to assume pricing mechanisms that ignore differences in the policy cum profitability options of different firms. This is despite the fact that applied people often find that the simple static Nash pricing models that we generally bring to data are not rich enough to explain actual behavior, and hence would welcome alternatives. Further the assumption of an unchanging environment limits the investigation of the implications of collusion to its impact on prices; ignoring the (possibly equally important) effects of collusion on the costs, qualities, and varieties of the products marketed.

If we are willing to give up on the elegance of analytic results and rely instead on numerical analysis, it is not difficult to analyze collusive models that allow for heterogeneity among firms who invest (sequentially) to develop their products, and can enter and exit. Fershtman and Pakes (2000) provide one such model, and it illustrates just how the analysis would proceed. As they note however, there are many possible collusive models that could be developed and the choice among the possibilities should probably be made with a particular industry in mind. De Roos (in process) computes an alternative collusive model designed to match up to the facts he has gathered on the lysine cartel recently adjudicated by the DOJ. He then considers data relating to the DOJ's investigation of the vitamin cartel in light of his and other models of collusion.

The Fershtman-Pakes model is a model with symmetric information in which it is hard to sustain collusion when either; one of the firms does not keep up with the advances of its competitors, or a "low quality" entrant enters. In either case there will be an active firm that is quite likely to exit in the near future. Not only is it hard to punish a firm who is likely to exit after it deviates, but if one of the competitors is near an exit state the other incumbent(s) has an incentive to price predatorily (that is to deviate themselves) in order to hasten that exit.

They assume that firm's either collude to set prices or set prices as in a static Nash pricing equilibrium. Collusive prices and profits are determined by the outcomes of a Nash bargaining game in which the threat value is the profits from the static, Nash in prices, equilibrium. The choice of which price vector to play depends on whether any incumbent has deviated from collusive prices in the past, and on whether the punishments currently available are sufficient to insure no firm has an incentive to deviate in the current period. If there is an incumbent who has deviated, the static Nash in price solution is played as long as that incumbent remains active. As in much of the repeated game literature (see Green and Porter,1984 , and Abreu Pearce and Stachetti,1986) no incumbent ever deviates. However there are tuples of states for which the punishment of reverting to non-collusive prices is not sufficient to support collusion, and this generates "price wars" (reversions to a Nash pricing equilibrium).

Formally the difference between the Fershtman-Pakes model and our core model is that they introduce a second state variable for each firm and let price choices depend on it. The second state variable is an indicator function which is one if the given firm has ever deviated from the collusive agreement in the past. The Bellman equation is then more complicated; profits are collusive profits if no incumbent has either; (i) deviated in the past *or*, (ii) has an incentive to deviate today. For values of the state vector in which no one has colluded in the past we compute both the vector of collusive profits and the vector of non-collusive profits. These are computed off line and then imported into the dynamic module of the program. The modification to the Bellman equation (equation 2) is that now at each state vector for which no one has deviated in the past, we must check to see that no one has an incentive to deviate in the current period. If either someone has deviated in the past, or someone has an incentive to deviate in the current period, then a static Nash in prices equilibrium ensues. If neither of these conditions are satisfied, then the collusive prices are played. With this modification to the Bellman equation one can compute equilibrium values iteratively, using techniques analogous to those in section 1.4. Note that this implies that policies are computed for values of the state vector in which each incumbent has deviated in the past, as well as for cases when none have ever deviated. Since, in the equilibria Fershtman and Pakes (2000) compute, no firm ever deviates, this implies that they need to compute the value function for states which should never actually be observed (states that are "off the equilibrium path").

Their numerical results are instructive for at least two reasons. First given the fixed values they chose for their "primitive" parameters (including the discount rate), they could only find two equilibria, a single equilibria in which collusion is observed, and an equilibria in which no firm ever colludes. Second, their results illustrate the potential importance of dynamic considerations in evaluating the benefits and costs of collusion. In particular for the parameter values they chose consumers prefer the collusive equilibria to the equilibria with no collusion; i.e. the greater quality and variety of goods that are marketed in the collusive equilibria more than compensates consumers for the higher prices they have to pay when there is collusion.

Dynamic Costs and/or Demand.

The core version of our model assumes that the distribution of future states, conditional on current states and all investments, is independent of the price or quantity choices of the agents. This assumption is inappropriate whenever either the current cost or the current demand function depends on the quantities sold (or the prices set) in previous periods. Quantities have an independent effect on future costs when learning by doing is important or when there are adjustment costs, and they have an independent effect on future demand when the goods being marketed are either durable, evaluated through personal experience, or addictive. Network effects can cause either future demand or future costs to depend on current price and/or quantity choices.

Recall that in the core version of the algorithm we can compute the current profit function "off line" and then simply import a "table" of the needed values for the profit function into the algorithm designed to compute equilibrium entry, exit and investment policies. Once current demand has an independent impact on either future costs, or future demand, this is no longer true. In these cases the Nash first order condition for equilibrium quantities (prices) has a term for the impact of current quantities (or prices) on future net cash flow as well as a term for their impact on current profits. As a result we cannot obtain either the "static" control or profits without computing the entire value function, and this makes for a more difficult computational problem.

The modification to the first order condition needed to accommodate these cases differs depending on whether the choice of the control for one firm effects the transition probabilities for the states of *all* firms, or just the transition probabilities for the states of the given firm. In games of quantity competition in which the transition probability for a firm's own states are determined only by own quantities, the simpler case where a firm's choice of controls only effects its own state prevails. Benkard (2000) analyzes a model of this sort and since his work has the added realism of a model build up from estimated parameters, we begin with an outline of it. When the control of a given firm is a determinant of the distribution of quantity outcomes for all competitors, as is usually the case when the control is price (see Berry and Pakes, in process), or when the control is a bid in a repeated auction with capacity constraints (see Joffre Bonet and Pessendorfer, 2000), then the first order condition must account for the fact that a given firm's choice of control affects the evolution of the states of all competitors. Later we illustrate what happens in these cases by modifying Benkard's model to allow for price, rather than quantity, competition.

Benkard (2000) incorporates learning by doing into a model similar to the one introduced in the last section. His goal is to analyze competition in the market for wide bodied commercial aircraft. He ignores investment in increasing the quality of the product (i.e. investment of the type we focused on in the early section), but allows current quantity choices to effect an experience variable, which in turn affects cost of production in future years[16]. Using estimated parameters for both the demand and cost functions, he then computes and analyzes a model of dynamic quantity competition among producers which is a reasonably realistic approximation to the competition that occurred in the commercial aircraft market at the time the Lockheed Tristar was being introduced.

Letting i index experience levels (or cost functions), and taking some liberties to place his model in the confines of our notation, he has $i_{t+1} = i_t + \tau_{t+1}$, where if Q is quantity produced, the distribution of τ_{t+1} is determined by the family $\{\lambda(\cdot|Q) \mid Q \in R^+\}$ which is stochastically increasing in Q. Quantities are a choice variable and the vector of quantities determine all prices. As a result the profits of the j^{th} firm can be written as

$$\pi(i_j, Q_j, Q_{-j}) = Q_j \left(p(Q_j, Q_{-j}) - mc(i_j) \right),$$

where $p(\cdot)$ provides the pricing function, and m represents a set of state

[16]Benkard's model could have been modified to allow for the possibility of investing in the quality of the product, as well as learning, at the cost of an increased computational burden. On the other hand he does allow for a number of exogenously evolving state variables that we omit in our discussion to keep the notation simple.

variables that evolve exogenously. Again letting $\underline{i}$ be the vector of states of the active firms (ordered so $i_r > i_{r-1}$), the Bellman equation in (5) becomes

$$V(i_j, \underline{i}) = max_{\chi \in \{0,1\}} \{[1-\chi]\phi \tag{6}$$
$$+\chi sup_{Q \geq 0}[\pi(i_j, Q_j, Q_{-j}) + \beta \sum V(i'_j, \hat{\underline{i}_j}' + e(i'_j)) q(\hat{\underline{i}_j}' | i, \underline{i}) \lambda(i'_j | i_j, Q_j)\}$$

where $\hat{\underline{i}_j}'$ is notation for the locations of firm $j's$ competitors in the next period. Note that there is no independent effect of the firm's own quantity choice on the distribution of its competitor's future states.

Now assuming that $\chi = 1$ (the firm continues in operation) and that $Q_j > 0$, the first order condition which sets Q_j is modified to read

$$\frac{\partial \pi(i_j, Q_j, Q_{-j})}{\partial Q} + \beta \sum V(i'_j, \hat{\underline{i}_j}' + e(i'_j)) q(\hat{\underline{i}_j}' | i, \underline{i}) \left(\frac{\partial \lambda(i'_j | i_j, Q_j)}{\partial Q_j} \right) = 0. \tag{7}$$

That is current quantity choices not only depend on the effect of quantity on current profits, but also its effect on future profits through its effect on experience. Since the value function is increasing in experience (in i) and the distribution of future experience is stochastically increasing in quantity, this model implies that more output will be put on the market than standard static Nash in quantities model would predict. The discrepancy between the output's generated by this and the standard model will be greatest when the "derivative" of the value function with respect to experience is steep. For a given distribution of competitors, this will tend to occur for values of the state vector at which the derivative of the learning curve is steep.

With this change in the first order condition, the equilibrium for Benkard's model can be computed iteratively as in section 1.4 (see also the next section). In prior work Benkard (forthcoming) did an extensive empirical analysis of learning by doing in the commercial aircraft market. In Benkard (2000) he computes the equilibrium corresponding to his estimated parameters and then analyzes the effect of learning on the nature of competition in the wide-bodied aircraft market. His numerical analysis finds that the effect of current quantity on future costs, and through future costs on its future competitive status, will induce the firm to produce large quantities in the early production years (the experience curve is steep early on in the production process). In fact production in the early years is pushed so far that price falls well below marginal cost in those years. This implication is clearly borne out by the price and cost data, and is inconsistent with a static quantity (or price) setting model (though consistent with some of the prior theoretical work on competition in industries with large learning effects; see Cabral and Riordan,1994 , and the literature cited their). Benkard then proceeds to an analysis of the producer and consumer surplus generated by the outcomes of the interactions in this market, and then goes through a series of counterfactuals which allows him to analyze what would have been likely to happen if we had imposed other institutional constraints on the market.

We now go back to Bellman equation for this problem and consider what would happen if the firms were playing a price, rather than a quantity setting game. Recall that the perception of the distribution of a firm's competitor's states in the future is given by $q(\hat{\underline{i}}'|i_j, \underline{i})$ and in equilibrium must satisfy

$$q(\hat{\underline{i}}'|i_j, \underline{i}) = \Pi_{r \neq j} \lambda(i'_r | i_r, Q_r).$$

If we were in a price setting game the quantity of the r^{th} firm would be a function of the prices set by all firms, i.e. $Q_r = Q_r(p_r, p_{-r})$. As a result the price the j^{th} firm sets not only effects its own quantity, and hence its own likely future experience levels, but it also effects its competitors quantities, and hence the competitors likely future experience levels.

As a result the first order condition that sets price (the analogue of (7) in the quantity setting model) is the more complicated expression

$$\frac{\partial \pi(i_j, p_j, p_{-j})}{\partial p_j} \tag{8}$$

$$+\beta \sum V(i'_j, \hat{\underline{i}}') \left(\sum_r \frac{\partial \lambda(i'_r | i_r, Q_r)}{\partial Q_r} \frac{\partial Q_r}{\partial p_j} \Pi_{l \neq r} \lambda(i'_l | i_l, Q_l) \right) = 0.$$

Berry and Pakes (in process) compute equilibria for their experience good model using the analogue of this first order condition and the iterative technique discussed in section 1.4.

There are a number of other related papers. One of the earliest is Judd's (1996) investigation of the robustness of the Bertrand and Cournot's assumption to the ability of firms to set both price *and* quantity. Judd considers a duopoly (with no entry or exit), lets (costly) inventories pick up the difference between produced and sold quantities, and allows for a cost of adjustment which insures that production does not equal sales. Judd's assumptions generate a linear quadratic game which, though restrictive in the institutions it can mimic, enables alternative and generally simpler computational strategies then those I review here[17]. He finds that if marginal costs are constant and adjustment costs are absent the equilibrium outcome mimics Bertrand models of competition, but if either marginal costs rise steeply, or their are high adjustment costs, the equilibrium outcome mimics Cournot.

Markovitch (1998) has modified the core version of the model discussed in this paper to allow for network interactions. She models the dynamics caused by the interactions between hardware and software choices. Consumers make a hardware choice that lasts two periods. Software is designed to run on one, and only one, of the two types of hardware. Software firms must commit to one of the two types of hardware when they enter, and then can invest to improve the quality of their product, or exit, just as in the core version of our model. The demand for a given software product depends not only on the vectors of qualities of software products available for each of the two hardware types, but also on the number of consumers who have purchased the different types of hardware in the past. Thus consumer's demand for hardware products depends on their beliefs about the likelihood of future software products available for each hardware type, while the entry exit and investment decisions of software firm's depends on their beliefs on the future hardware purchases of consumers. She solves for a rational expectations Markov Perfect equilibrium and finds that if the industry's competitors (the outside alternative) are not progressing too quickly the equilibrium is one where both types of hardware are produced (the "variety" equilibrium), while if the competitors to the industry are growing quickly we see an equilibrium with only a single type of hardware produced (an equilibrium with "standardization".)

We are clearly just at the beginning of working with models in which price or quantity choices are determined by dynamic incentives. What seems to be clear is that the number of potentially important applications here is enormous. Much of manufacturing produces durable goods, our marketing colleagues tell us that experience is an important determinant of demand for most consumer products, and network effects are said to be pervasive in "new economy" industries.

[17]Though Judd also reports doing robustness analysis using numerical approximations similar to those discussed in section 4 below.

2.3 Allowing for Investment Externalities.

The framework has not been used to study models with externalities. This then is the second case that I include in my description solely because of its potential importance to applied work. Externalities can be appended to our core models in several different ways. For simplicity I will stick to the spirit, but not the letter, of the models used by Steve Klepper and his coauthor(see Klepper,2000, and the literature cited their) in their study of industry evolution.

We will allow for two types of investment. There is a "traditional" investment which is similar to the investment in the core model in that it only increases the value of the firm's own state which for simplicity we will take as an index of the quality of the firm's own product, and an R&D investment. The R&D investment, if successful, decreases the firm's marginal cost in the coming period (marginal costs are constant over quantity levels). However the firm can only appropriate the gains from the output of its R&D activities for a single period. The output of last periods R&D activity gives the firm a cost advantage in the current period, but during this period all firms learn how to imitate the new "best practice" technique[18]. Of course in the interim each firm will have done further R&D which, if successful, will again give them a one-period cost edge, and so on.

If we let u be the output of this period's R&D activity, and b be current best practice (the minimum cost of production in the previous period) then current marginal cost is

$$mc(u_t, b_t) = b_0 + b_t exp[-u_t]$$

while best practice evolves as

$$b_t = min_j mc(u_{t-1,j}, b_{t-1})$$

where the minimum is taken over the firms active in $t-1$. There is a family of distributions for u which is stochastically increasing in the amount or

[18]The interpretation here is that the externality shifts the cost function down equally for all different qualities of the product. An alternative would be to make the externality larger in a local neighborhood of the quality, or characteristic vector, of the product being produced (see the discussion in section 2.1). Note that spillovers that enabled better qualities to be produced at the same cost could be treated in an analogous way to the analysis in the text.

research investment (in r)

$$\{P_u(\cdot|r), r \in R^+\}.$$

The family of distributions for next periods quality (i') given i and traditional investment (x) is as specified in section 1.2.

This model has two firm-specific state variables, i or "quality" and the current values of u, and one industry specific state variable, b, or best practice technology. The vector which counts up the number of firms at each possible (i, u) couple is

$$s \in S \equiv \{[s_{i,u}] \text{ with } s_{i,u} \in \mathcal{Z}^+ \text{ for } i \in \Omega, u \in U, ; \text{ and } \sum_{i,u} s_{i,u} \leq \overline{n}\},$$

and to specify the state of the industry we have to specify an $s \times b \in S \times B$ where B is the set of possible best practice technologies.

To complete the model we need assumptions which generate demand conditional on the quality vector and prices, and an equilibrium assumption for the "spot market" which determines current production, say Nash in prices. These assumptions, when combined with the cost function above, allow us to determine profits as in the core version, and calculate them "off line". Let these profits be $\pi(i, u, s, b)$[19]. The Bellman equation for the value function can then be written as

$$V(i, u, s, b) = \max\{\phi, \pi(i, u, s, b)$$

$$\sup_{r\geq 0, x\geq 0}[-c(x+r)+\beta \sum V(i+\nu-\zeta, u', \hat{s}_i+e(i+\nu-\zeta, u'), b')\lambda(u'|r)\lambda(\nu|x)q[\hat{s}_i|i, u, s, \zeta]\lambda(\zeta)]\},$$

and b' is the known function of current s given above.

We can now proceed pretty much as before, realizing that we have two controls, and noting that the first order condition for one of them (r) might effect the distribution of a firm's competitors costs in future periods as well as its own (producing a Bellman equation analogous to that in equation 8 above.)

[19]Note that this way of proceeding assumes that current marginal costs are public information even though it takes a period for that public information to become embodied in the cost functions of all competitors.

2.4 Horizontal Mergers.

Almost all of the formal models of merger activity *condition* on the cost, qualities, and variety of products sold in the market. These models hold the distribution of characteristics of the products being marketed (as well as the nature of competition) fixed, and analyze the impact of the ownership change on producer and consumer surplus. The producer surplus analysis provides a vehicle for analyzing the incentives to merge. When the intention is to analyze whether the merger is beneficial to society, we combine the producer surplus analysis with analysis consumer surplus.

As noted in Stigler's(1965) investigation of the US Steel merger, the results from such a "static" analysis of mergers can easily be overturned by simple dynamic considerations (his discussion allowed for adjustment costs in an analysis of mergers in a homgenous homogeneous goods industry). The first attempts I know of to build a model to analyze the dynamic effects of mergers are in articles by Cheong and Judd (forthcoming) , and Berry and Pakes (1993). They both analyze the impact of a "one-time" exogenously specified merger in a dynamic model which allows for investment but does not allow for any further mergers. These papers show that mergers can be beneficial to *both* the firms merging and to society, even if the profits of the merging firms *and* consumer surplus falls at the time of the merger. The predominant reason is that there is less of an incentive to invest in the merged industry, and the Markov Perfect equilibrium generates more investment than a social planner would (see Mankiw and Whinston,1986, for the intuition underlying these arguments).

To be realistic a model which investigated the dynamic impacts of mergers would want to allow mergers to arise endogenously, and not just investigate the impacts of a "one-time" exogenously specified merger. There are many unsolved problems here, not least among them being the diversity of views on the factors motivating merger activity in different industries. In addition to specifying the possible sources of gains from mergers, a merger model must also specify a market mechanism for determining which among the possible profitable mergers at any point of time are in fact consummated.

One such mechanism is provided in Gowrisankaran(1999). He takes the capacity constrained homogeneous goods version of our framework and adds to it a merger game. The merger game occurs at the beginning of each period and proceeds in the following sequential manner. The largest firm is allowed to choose a merger partner first. All other firms present the largest firm with a "take it or leave it" price at which they are willing to be bought. Information is symmetric except that the largest firm draws a "synergy" value for each merger which is known only to it; i.e. the synergy value for a given firm is not known to any of the firms that might be acquired. The largest firm chooses to merge with the firm which generates the highest net merger value provided that value is positive. The net merger value is the the expected discounted value of future net cash flow if a merger would take place net of the price of the acquisition and what the value of the firm would be if the merger did not take place. If a merger takes place the process restarts (there are new take it or leave if offers, and new synergy values), and the (new) largest firm can choose another merger partner. When the largest firm chooses not to merge further, the second largest firm gets to choose a merger partner in the same way. This process continues until the smallest active firm chooses not to merge further. At that point production, investment, and then exit followed by entry and investment decisions are made. All offers and actions are made to maximize the expected discounted value of future net cash flows given the agents' information sets, and the equilibrium is Markov Perfect.

Perhaps the most striking part of this analysis is that it in fact can be done. Given the quantitative magnitude of the merger phenomena in recent years and the extent that it can be impacted by policy, any step in developing a usable models of mergers is welcome. Still, it is clear that we are only at the beginnings of developing an ability to provide realistic dynamic models that allow for mergers; alot of work remains to be done.

3 The Computational Burden of the Simple Algorithm.

We begin with a simple procedure for determining the computational burden of the algorithm, and then use examples to show how that burden changes with increases in the number of state variables needed for the problem. The next section provides an overview of two alternative algorithms designed to make computation easier.

The Determinants of the Computational Burden.

The description of the algorithm in section 1.4 makes it clear that its computational burden is (essentially) the product of three factors,

1. the number of points evaluated at each iteration;
2. the time per point evaluated;
3. the number of iterations.

I now consider the rate of increase in the computational burden of each of the three components of the algorithm as the number of state variables in the problem increase. Again K denotes the number of distinct tuples for the firm specific state variable, or $K = \#\Omega$. So if i is a $d-$*component* vector each of whose elements can each take on k values then, without further restrictions, $K = k^d$. $\overline{n}$ is the maximum number of firms ever active.

The rate at which (1) above increases with an increase in the number of state variables differs depending on whether the increase is in d or in $\overline{n}$. The relationship between (1) and d is typically model specific, so we discuss it in the context of our examples below, and focus here on the relationship of (1) to $\overline{n}$.

Since each of the $\overline{n}$ active firms can only be at K distinct states, the number of points we need to evaluate at each iteration, or $\#S \leq K^{\overline{n}}$. However symmetry, or more precisely exchangeability, of the value and the policy functions in the state variables of a firm's competitors implies that we do not need to differentiate between two vectors of competitors that are permutations of one another. As shown in Pakes (1993), this insures that an upper bound for $\#S$ is given by the combinatoric

$$\binom{K+\overline{n}-1}{\overline{n}},$$

but for $\overline{n}$ large enough this bound is tight. The bound increases *geometrically* (rather than exponentially) in $\overline{n}$. Still, even with geometric rates of increase, computational constraints often become binding for applied problems of interest (see below).

The computational burden in (2), or at a given point, is primarily determined by the cost of calculating the expected values of future states (of obtaining the $w^k(\cdot; i, s)$'s from the information in memory). The burden of obtaining the optimal polices and the new value function given $w^k(\cdot; i, s)$ need not depend on either $\overline{n}$ or K.

Say iteration $k-1$'s policies determine that m firms will be active at point s (accounting for entry and exit). Also assume that there is positive probability on each of κ points for each of the $m-1$ active competitors of a given firm. Then to compute each $w^k(\cdot)$ we need to sum over κ^m possible future states and there are $\kappa \times m$ values of $w^k(\cdot)$ needed at that s[20]. Thus the computational burden at a given s grows exponentially with the number of firms active at s. The relationship between the computational burden per point and the number of state variables per firm is determined by the relationship between those state variables and κ. This varies with the characteristics of the model being analyzed, a point we return to below.

There is very little known about the relationship between the number of iterations and the number of state variables of the problem. Our experience has been that the number of iterations did not increase dramatically (often not at all) as we moved to larger problems, but we have not consistently kept track of this dimension of the problem. We have also tried adding policy and value iteration steps to the algorithm (these will be described in section 4.1 below). These are techniques that have been used to decrease the number of iterations in related problems, but they were not helpful in speeding up convergence of the point-wise calculations.

An Extended Example.

As an example we consider the calculations in PM (1994). That paper presents an analysis of a differentiated product market in which the products were unidimensional (so each 'i' is an integer), $K = 21$, and $\overline{n}$, the maximum number of firms ever simultaneously active, was 6. A typical run took about three hours on our Sun Sparc 2 workstation. If we would have increased market size until $\overline{n} = 10$ then we would have had to compute equilibria with *forty seven* times as many points. Since $\kappa = 2$ the increase of $\overline{n}$ from 6 to 10 would also entail a *twenty two fold* increase in the computational burden per point evaluated. Thus if we optimistically assume both that the number of iterations would not increase when we increased $\overline{n}$, and that there was no memory problems, the increase from $\overline{n} = 6$ to $\overline{n} = 10$ would increase computational time by a factor of over a *thousand*, to over three months, and one with $\overline{n} = 12$ would take several years. A dual processor pentium II is eight or nine times faster for these problems, so we could imagine that with top of the line desktop hardware and sufficient RAM we might soon be able to

[20]We sum over a function of each possible future value of the tuple $(\{\nu_j\}_{j=1}^{m-1}, \zeta)$. We could reduce this by using the symmetry restrictions discussed above, but this would require us to find the probabilities associated with each unique $\hat{s}_i^{'}$ vector; a task whose computational burden generally outweighs the gains from using symmetry.

analyze a $K = 21, \overline{n} = 10$ case, but not a larger problem.

Unfortunately the computational problem gets much more severe if we allow for many state variables per firm (or $d > 1$), as we are likely to need to do in applied work. Without further restrictions both K and κ grows exponentially in d. Plugging this into the formulae for the number of points and for the computational burden per point generates results which makes the limits of point-wise computationally techniques apparent. For example to accommodate two state variables per firm with 21 grid points each, $\overline{n} = 6$, and no further restrictions, we would have had to increase the number of points evaluated at each iteration by a factor of 1.9×10^8; which makes it a computational problem which is out of range of even the most sophisticated of supercomputers.

Further restrictions *are* often available when $d > 1$, and they can be quite useful. A few examples will serve to illustrate this point. We begin with the collusion problem analyzed by Fershtman and Pakes(2000). Their model has two state variables per firm, one that is similar to that in PM (it takes on 21 values), and an indicator function which is one if the firm has deviated from the collusive regime in the past. Since in equilibrium nobody actually deviates, to determine behavior it suffices to compute the fixed point over the tuples of points in which only one (though any one) of the active firms deviates. Thus in this case, the addition of the second state variable increases the number of points by a factor of $< \overline{n}$. There is an increase in the computational burden per point; but it is independent of both $\overline{n}$ and K. The collusive runs with $\overline{n} = 4$ took about forty-five minutes, and those with $\overline{n} = 5$ took about three hours, both on our dual processor pentium 2.

Models in which the second state variable is a fixed location (or a fixed "type" of any form as in Gowrisankaran and Town,1997, or Benkard,2000) are similar in that the computational burden per point does not increase at all, and the number of points grows linearly (rather than exponentially), in d. In models with multi-product firms the order in which the alternative products of the same firm are listed does not matter, and this also allows computational savings (see Gowrisankaran, 1999a). On the other hand if we allow for separate investments in each product κ can grow exponentially in the number of products per firm (again see the Gowrisankaran 1999a article for discussion and some bounds on the computational burden).

I hope the preceding discussion has made it clear that though there are both substantive and pedagogic problems which can be analyzed with the s-tandard algorithm, there are also many important applied problems for which the computational constraints of obtaining an "exact" solution are overly demanding. At that point the problem is either abandoned or modeling choices become dominated by their computational (rather than their substantive) implications. With this in mind the next section outlines two computational techniques which, at the cost of introducing some approximation error, have the potential of computing equilibria for models that are several orders of magnitude more complex than those that have been computed with the point-wise techniques discussed above.

4 Approximation Techniques.

This section is solely for the reader who needs computational tools that are more powerful than those described above. It outlines two algorithms which produce "approximate" solutions to the computational problem at a much lower computational burden than that of the standard algorithm. I do not know of publically available versions of these algorithms, though the "stochastic" algorithm introduced below is much easier to program than any of the other algorithms discussed in this paper.

The first of these techniques is essentially a curve fitting technique. It finds a member of a parametric class of functions that "closely approximates" the value function at a few points, and then uses that function to derive policies on all of S as needed. It can use many of the tools from the more general literature on numerically approximating fixed points (see Judd,1999, and the literature he cites). The second is an artificial intelligence algorithm which can be endowed with a behavioral interpretation that is familiar from the recent economics literature on learning (see Lettau and Uhlig,1999. for applications in macroeconomics, and Fudenberg and Levine,1999, for applications to matrix games). As noted below for many purposes we only need policies for the recurrent class of points (see below). The stochastic algorithm confines its efforts to obtaining accurate policies on the recurrent class and this can reduce the number of points we need to update dramatically. This algorithm reduces the computational burden per point by substituting a monte carlo estimate for the explicit summation which defines continuation values. Both algorithm's can, at least in principal, break the "curse of dimensionality" associated with increasing the number of state variables in the problem.

The goal of the section is to provide a simple overview of how each of these algorithms work. I will rely on references to the existing literature for details where such references are easily accessible.

4.1 Deterministic Approximations.

Approximation techniques have been used to compute value or policy functions in economics for some time (see Judd,1993, and the literature cited their for a review). These techniques begin by specifying a set of functions considered rich enough to contain an element which provides a good approximation to the value function (we consider polynomials of order d). It then uses a small subset of the points in $\mathcal{S}$ to find an approximating member of the set of functions, and uses this function to predict the value function at other points as needed.

Briefly, take our problem with a single state variable per agent (for more detail see PM,1994, PM, 1995, and the literature they cite). Then our polynomial approximating functions will be maps from $\Omega^{\overline{n}} \rightarrow \mathcal{R}$. The collection of all polynomials of order r in $\overline{n}$ arguments is a vector space, say $\mathcal{V}(r, \overline{n})$, and can be generated as the set of linear combinations a set of $J(r, \overline{n})$ basis functions [$J(r, \overline{n})$, or just J, is the dimension of the space]. We are looking for an $\alpha \in \mathcal{R}^{J(r,\overline{n})}$ [i.e. a $\hat{V} \in \mathcal{V}(r, \overline{n})$] which generates a linear combination of the basis functions which is a good approximation to the value function.

The algorithm for finding the approximating function is iterative. In memory at iteration k is a value for α say α^{k-1}, and last iterations policies *for a set of basis points*, in our case vectors $(\omega, s) \in \Omega \times S$ at which we will evaluate the value function. If there are J basis functions, the only restriction on the set of basis points is that they generate at least J linearly independent values for the basis functions. At each iteration we cycle through the basis points in some predetermined order and update only the policies associate with those points. The updating procedure is precisely the same as the updating procedure discussed above except that now when we calculate continuation values we use the α^{k-1} and the appropriate basis functions to generate the values at each possible future locations (these locations will not, in general, be members of the set of basis points). In the "exact" calculations we simply called these values up from memory.

The last step of this updating procedure generates an estimate of the value function at each of our basis points, say $V^{*k}(\cdot)$. These estimates will not (in general) be polynomials of order d, and we do not keep them in memory (as we did in the point-wise procedure). Instead we want to find an α^k that generates polynomial estimates of the value function at our basis points that are "as close as possible" to $V^{*k}(\cdot)$. Let U be the matrix whose rows are the values of the basis functions associated with the basis points. Then if we choose our α^k to minimize the sum of squared deviations between $V^{*k}(\cdot)$ and $U\alpha$ we obtain the familiar OLS formula

$$\alpha^k = [U'U]^{-1}U'V^{*k}.$$

This α is held in memory, and we continue iterating until the $\alpha's$ from successive iterations are "close enough" to one another. We now consider the computational burden of this procedure.

The Number of Basis Points

At each iteration we update at least $J(r, \overline{n})$ basis points. Without further restrictions the number of these points still grows geometrically in $\overline{n}$, i.e. at the same rate as the number of points grew in the point-wise algorithm, though this time as a r^{th} instead of a K^{th} order polynomial. However, when we were considering the point-wise algorithm we used the fact that the value and policy functions were exchangeable in the state vector's of a firm's competitors to restrict the number of points that had to be held in memory. An obvious way to use the same information when using the curve fitting technique is to restrict the set of approximating functions to have the properties of the true value function; i.e. to be exchangeable in the states of the firm's competitors.

The space of polynomials of order r that are exchangeable in the state vectors of a firm's competitors together with the usual operations of addition and scalar multiplication, is also a vector space, and the number of basis points that will be needed to span that space is equal to its dimension. Pakes (1993) proves that the dimension of the exchangeable basis is *independent* of $\overline{n}$ and provides a formula for an upper bound to it (the bound is tight for $\overline{n}$ large enough). Pakes and McGuire,1994, compute the bound for alternative r and provide the needed basis functions.

So if we are willing to use exchangeable polynomials to approximate the value function, the number of points we need to use in our computational algorithm is independent of $\overline{n}$. For intuition on why this is true consider an approximation by polynomials of order 1; i.e. $\hat{V}(i_1; i_2, \ldots, i_n) = \sum \alpha_j i_j$. To be exchangeable in the state vector of the competitors we need $\alpha_j = \alpha_{j'}$ for all couples (j, j') such that neither j nor j' equals one (consider tuples where all i but the firm itself and one of its competitors are zero, and vary the single competitor with $i \neq 0$). This implies that there are two first order basis functions; i_1 and $\sum_{j\neq 1} i_j$. Analogous logic show there are four second order, seven third order, and twelve fourth order bases functions. In fact a tenth order polynomial approximation requires only 407 basis points, *regardless* of $\overline{n}$.

Computational Burden Per Point.

As noted, the computational burden of computing the optimal policies once the continuation value for each possible outcome of the firm's own state (or the $w^k(\cdot)$) are known is independent of $\overline{n}$. So we focus on the burden of computing the $w^k(\cdot)$.

Without further simplification this burden is *larger* when using the polynomial approximations than it is in the point-wise calculations. In the point-wise calculations we can simply call up the elements of the $V^{k-1}(\cdot)$ corresponding to the possible future states needed to compute the $w^{k-1}(\cdot)$ from memory. In the polynomial approximations we need to hold the basis functions corresponding to these future states in memory, and then compute the $V^{k-1}(\cdot)$ as a multiple of these basis functions and α^{k-1}. However, as we now show, a suggestion due to Kortum(1992) to use moment generating techniques in single agent problems can also be applied to our problems, and this decreases the computational burden per point significantly.

It will suffice to consider computing $\hat{w}^k(v; i_1, i_2 \ldots, i_{\overline{n}})$ when we use a first order exchangeable basis. Then

$$\hat{w}^k(v; i_1, i_2 \ldots, i_{\overline{n}}) =$$

$$\sum_{\zeta} \sum_{v(2)\ldots v(\overline{n})} \left((i_1 + v - \zeta)\alpha_1^{k-1} + \sum_{j=2}^{\overline{n}} (i_j + v_j - \zeta)\alpha_2^{k-1} \right) \Pi_{j=2}^{\overline{n}} \lambda(v_j | x_j^{k-1}) \lambda(\zeta),$$

where Π is the product operator. This can be rewritten as

$$\sum_{\zeta} \left((i_1 + v_1 - \zeta)\alpha_1^{k-1} + \left(\sum_{j=2}^{\overline{n}} \mu_j(1,\zeta)^{k-1} \right) \alpha_2^{k-1} \right) \lambda(\zeta),$$

where

$$\mu_j(q,\zeta)^{k-1} \equiv \sum_{v(j)} (i_1 + v_j - \zeta)^q \lambda(v_j | x_j^{k-1}),$$

the mean of $(i + v_j - \zeta)^q$ conditional on ζ and iteration $k-1$'s policies. Similarly the r^{th} order basis function can be shown to be a function of $(i_1 + v_1 - \zeta)^q$ and the $\mu_j(q,\zeta)^{k-1}$ for $q \le r$. Further it is straightforward to rewrite all the needed expressions as functions of $\sum_{j=1}^{\overline{n}} \mu_j(r,\zeta)^{k-1}$ and $(i_1 + v + \zeta)$.

Since we can calculate $w^{k-1}(\cdot)$ directly from these conditional moments, once we know those moments there is no need to calculate $V^{k-1}(\cdot)$ at all. So the computational burden at each point becomes the burden of computing the $\mu_j(r,\zeta)^{k-1}$ and then summing them. If there are m firms active at the point, then the number of moments we need to calculate is proportional to $m \times r$, but these moments suffice for computing the $w(\cdot)$ for all (i_1, i_{-1}) tuples generated by a given s. Thus for a given r the number of moments we need to calculate need not increase in $\overline{n}$ at all. Of course we have to sum these moments over the active firms and the number of terms in this sum grows linearly in the number of firms active at the point. Still this is slower than the growth in the computational burden per point in the "exact" calculations.

Number of Iterations.

As noted are at least two ways of adding steps to each iteration of the algorithm in an attempt to decrease the number of iterations until convergence, and though these were not useful in the point-wise calculations, they proved quite helpful when we moved to polynomial approximations.

First we can add a *value iteration* step to each iteration. This step holds the continuation values fixed at their values in iteration $k-1$ and iterates on the policies for iteration k until they are a Nash equilibrium to the game whose outcomes are evaluated by these continuation values. Before when we calculated the k^{th} iteration policies of incumbents and potential entrants we assumed that the policies of their competitors are fixed at the $k-1$ iteration values. Now we define a subsequence of policies for each fixed α^{k-1}, and continue iterating until that subsequence converges. In addition we could add a *policy iteration* step. This step, which is familiar from the dynamic programming literature (see Rust,1994 , Bertsekas,1995 , and the literature cited in these articles), holds the policies fixed at those out-putted by the algorithm for iteration k and then iterates on the α (and hence the continuation values) until it converges for those policies.

Though the value iteration step did not seem helpful in the point-wise calculations once we moved to the approximations it seemed almost essential (without it the algorithm with the polynomial approximations seldom converged). Moreover sometimes the series generated by the sequence of policies for the given value function did not converge, and when this happened we substituted a Newton method for the iterative procedure used to find the iteration k policies. It can be shown that a matrix inversion of dimension equal to the number of basis points can be substituted for the policy iteration step when we use polynomial approximations. If the number of basis points is small enough this is very easy to do, and we found it to be quite useful also.

Concluding Comments on Polynomial Approximations.

A few final comments on polynomial approximations are in order. First we were not terribly successful when we used them on problems the size of the problem computed in PM(1994). Part of the reason is that the value functions for an incumbent can be discontinuous in the value of the competitor's to that incumbent (this because the incumbents value function will tend to jump when the competitor's value passes over a margin which either induces entry or exit). This tends to cause convergence problems. There are many ways of modifying the algorithm to take better account of this behavior (for example we could use different polynomial approximations for different numbers of firms active). Relatedly the discontinuities are likely to be less problematic when $\overline{n}$ is larger; and we have only tried problems when $\overline{n}$ is small. This was apparently true in the work of Liu 1999, who with a much larger $\overline{n}$ and approximations similar (but not identical) to those introduced here had little difficulty in computing his value functions.

Second once we move to polynomial approximations, the exchangeable basis, and the moment generating technique discussed in this section, the computational burden of the approximation algorithm becomes *independent* of the fineness of the grid (at least provided the calculation of the moments designated by $\mu_j(q, \zeta)$ does not depend on the grid). Thus it should be relatively easy to check whether the choice of grid has had a significant impact on the analysis of the issues of interest.

Third, note that when there is a convergence problem for the iterative procedure described above, and in our examples we often encountered such problems, we could transfer to a nonlinear search procedure. Thus, for example, we could search for an α which minimizes $||\alpha - \alpha[x(\alpha)]||$ where $x(\alpha)$ is the solution for the optimal policies when the value functions is given by the α, and $\alpha[x(\alpha)]$ is the matrix solution for the optimal α given this x. Finally the reader should know that there are a large number of other modifications that might help improve the performance of the procedures described in this subsection. A good reference for many of them is Ken Judd's(1999) new book.

I now move on to describe ways of adapting ideas from the artificial intelligence literature to compute Markov Perfect equilibria. Again the discussion will be designed to provide only an overview of those techniques. In particular I will not discuss the question of how to combine the "curve fitting" techniques discussed in this subsection with the stochastic approximation ideas presented in the next – even though I view such a combination as an extremely promising avenue for future research[21].

4.2 The Stochastic Algorithm.

The stochastic algorithm is a "learning" or artificial intelligence algorithm that agents might actually use to infer optimal behavior from past outcomes. Computationally it has two distinct properties which combine to break the relationship between the dimension of the state space and the burden of computing equilibria; one changes the number of points that need to be evaluated and the other changes the relationship between the computational burden per point and the dimension of the state space. It turns out that the resulting computational burden is largely determined by the economic properties of the model analyzed. Moreover, as we explain below, there is good reason to believe that the computational burden of many I.O. models will grow quite slowly (if at all) as we increase their state spaces. Numerical results reinforce this view.

The first difference between the stochastic and the other algorithm's discussed thus far is that the stochastic algorithm is *asynchronous*, i.e. it only updates a single location at each iteration. Moreover the selection process which chooses the transitions between updated locations eventually confines its attention to the recurrent class of points, or R, a subset of S whose cardinality does not necessarily depend on the dimension of the state space. So the stochastic algorithm breaks the relationship between the number of points analyzed and the dimension of the state space by simply giving up

[21]This next section is based on Pakes and McGuire, forthcoming, and many of the omitted details can be found in that article. Section 9 of Barto, Bradtke, and Singh, 1995, provides some discussion of combining artificial intelligence techniques with other forms of approximations.

on the problem of computing policies on all of S. On the other hand we know that if $s_t \in R$, then so will be all future s (with probability one), and the algorithm is designed to tell the user whether the user-specified initial condition is in R (there are modifications that can be used when it is not; but they do require additional computation). The computational advantages gained by focusing on R depends on the relative sizes of R and S, but as we note below $\#R$ is likely to be quite small relative to $\#S$ in many I.O. problems.

The second difference between the stochastic and the other algorithms is that it never actually does the explicit summation that determines the continuation values needed to determine policies at a given point (i.e. to determine the $w(\cdot)$ in 4a). Rather it uses a "monte carlo" estimate of these continuation values. The simulation draws that underlie this monte carlo estimate are the outcomes from previous iterations of the algorithm. Use of simulation rather than explicit integration breaks the relationship between the computational burden per point and the dimension of the state space. Simulation is both faster and has less memory requirements than the alternative of determining continuation values by explicit integration. However the simulated estimate of the continuation value is also less precise (particularly in early iterations when there are few past outcomes to average over). This generates a tradeoff between the computational burden per point, and the number of iterations needed for a given level of precision. Since the precision of the estimate does not (necessarily) depend on the dimension of the integral being estimated, while the cost of doing the summation explicitly does, the larger the dimension of the state space the more we expect the tradeoff to favor the stochastic procedure.

The Updating Procedure.

The stochastic algorithm is also iterative (though now asynchronous). The k^{th} iteration is defined by an estimate of w, say w^k, and by its location, an $s^k \in S$. Thus the algorithm's updating rule must update both s and w. The way this is done mimics rules used in the reinforcement learning literature[22]. The reinforcement learning literature is often used to formulate a set of rules that might actually be used by agents attempting to infer optimal behavior

[22]See Bertsekas and Tsikilis, 1996, for a review of related techniques in the context of single agent problems and zero sum games. That book also contains numerous suggestions for improving the efficiency of the techniques outlined here.

from past outcomes (indeed they are used in various "machine learning" situations). As a result it is possible to interpret our algorithm as a decision making process that agents in a particular market might actually use.

With our notation and analogies to the learning literature we can begin with a short verbal explanation of how the stochastic algorithm works. Assume $s = s_t$ and that all agents believe that the expected discounted value of future net cash flows are given by $w(\cdot|\cdot, s_t) = w^+(\cdot|\cdot, s_t)$. Each agent would then choose their policies to maximize the $V(\cdot, s_t|w^+)$ obtained by substituting w^+ for w in (4) above. These choices would generate a distribution of outcomes for each agent's competitors given by $q^{w+}[\cdot|i, s_t]$ (recall that this is the distribution for $\hat{s}'_i$, that is for agent $i's$ competitors, in the next period). Nature would then choose the market outcome, and it would be a random draw from q^{w+}. The current perception of the *value* of this outcome to firm i is obtained by substituting its $\hat{s}'_i$ into the evaluation function given by w^+ and (7). If these values are viewed as random realizations from the integral defining the appropriate components of the true w, which henceforth we will denote as $w*$, the agents might use them to update their estimates of $w*$. Our algorithm for finding $w*$ mimics the learning algorithm just described.[23]

We begin with the update for s. This requires policies for the incumbents and the potential entrant. The investment and exit policies policies for incumbents, say, $x(\cdot, s^k|w^k)$ and $\chi(\cdot, s^k|w^k)$ are obtained as the solution to

$$\max_{\chi\in\{0,1\}}\{[1-\chi]\phi \ + \ \chi sup_x[\pi(i, s^k) - cx + \beta \ \sum_\nu w^k(\nu; i, s^k)\lambda(\nu|x_1)]\},$$

while the entry policy is given by

$$\chi_e(s^k|w^k) = 1 \Leftrightarrow \beta w^k(0; i_e, s^k + e(i_e)) > x_e,$$

where $e(i_e)$ is a K-vector which has one for its i_e element and zero elsewhere.

These policies determine a distribution for s^{k+1}. The actual s^{k+1} is obtained as a random draw from this distribution. To obtain the draw first use

[23]Note that though the interpretation of our algorithm as a learning algorithm is a useful pedagogic device, its empirical relevance depends on several issues. These include the method by which information on past outcomes is made available to the agents currently active, and the number of updates possible per unit of time. Relatedly, I do not know of any attempt to interpret real, as opposed to experimental, data using reinforcement learning techniques. On the other hand the potential usefulness of these techniques for understanding empirical phenomena, especially for explaining behavior in recently established markets, seems obvious.

$\chi(\cdot|w^k)$ to determine which of the firms in s^k remain active, and $\chi^e(\cdot|w^k)$ to determine whether or not their is an entrant. Now take random draws for the $\nu's$ of the incumbents that remain active and a random draw on ζ. This determines the $k+1^{th}$ iteration's state of each agent active in that period, and all that remains is to reorder those states into s^{k+1}. [24] Note that if $w^k = w*$ the process generating s^{k+1} would be an ergodic Markov process, and hence would wander into the recurrent class in a finite number of iterations and stay there.

We now update w^k. For each agent and each possible realization of ν, use $V(\cdot|w^k)$ [equation (7) with w^k substituted for w] to evaluate the state defined by the actual *simulated draws* for ζ and for the locations of the agent's competitors; i.e. use (7) and w^k to calculate

$$V(i+\nu-\zeta^{k+1}, \hat{s}_i^{k+1} + e(i+\nu-\zeta^{k+1})|w^k).$$

This expression is the k^{th} period evaluation of being in location $(i+\nu-\zeta^{k+1})$ when all the other competitors states are determined by their simulated draws. Its expectation conditional on information realized by iteration k is $\sum_{(\hat{s}_i', \zeta)} V(i+\nu-\zeta, \hat{s}_i' + e(i+\nu-\zeta^{k+1})|w^k) q^{w^k}[\hat{s}_i'|i, s^k, \zeta]\lambda(\zeta)$. So if $w^k = w*$, then its expectation is $w*$.

Since $V(i+\nu-\zeta^{k+1}, \hat{s}_i^{k+1} + e(i+\zeta-\zeta^{k+1})|w^k)$ is the current period's perception of the value of a random draw from $w^*(\nu; i, s)$, it is used to update $w^k(\nu; i, s)$. In particular if $V(i+\nu-\zeta^{k+1}, \hat{s}_i^{k+1} + e(i+\zeta-\zeta^{k+1})|w^k)$ is different from $w^k(\nu; i, s)$, then set $w^{k+1} - w^k$ equal to a fraction of the difference. More formally if $\alpha(k, s^k) \in (0,1)$ set

$$\begin{gathered} w^{k+1}(\nu; i, s^k) - w^k(\nu; i, s^k) = \alpha(k, s^k) \times \\ \{V[i+\nu-\zeta^{k+1}, \hat{s}_i^{k+1} + e(i+\nu-\zeta^{k+1})|w^k] - w^k(\nu; i, s^k)\}. \end{gathered} \tag{9}$$

Note that if we set $\alpha(k, s^k)$ equal to the inverse of the number of times the estimate of $w^*(\nu; i, s)$ has been updated in the past, then the $w^k(\cdot; i, s)$ are just the sample average of past draws on the expected discounted value of

[24]More formally let the r^{th} active agent's location be i_r^k and its investment be $x_r^k(\cdot|w^k)$. Then for each active agent draw a random variable from the distribution $\lambda(\cdot|x_r^k)$, say ν_r^{k+1}. Also draw ζ^{k+1} from $\lambda(\zeta)$. We obtain s^{k+1} as follows. Compute $i_r^k + \nu_{1r}^{k+1} - \zeta^{k+1}$ for each active agent. If $\chi_e(w^k) = 1$, also compute $i_e - \zeta^{k+1}$ for the potential entrant. Now count how many of these numbers equal i for each $i \in \Omega$. The vector of integers obtained by this procedure is s^{k+1}.

future net cash flows (i, s). This is a familiar choice for the $\{\alpha(k, s^k)\}$, even though the fact that the values from earlier iterations are less precise implies that there are typically more efficient ways to choose these weights. Also if $V(i + \nu - \zeta^{k+1}, \hat{s}_i^{k+1} + e(i + \nu - \zeta^{k+1})|w^k) = w^k(\nu; i, s)$, then $w^{k+1}(\nu; i, s) = w^k(\nu; i, s)$. Consequently if $w^k = w*$ then the expectation of w^{k+1} is $w*$. I.e. if we are at $w*$ we will tend to stay their.

We have now shown how to update both w^k and s^k. This is the core of the algorithm. We keep updating until we can show that the values and policies in memory are equilibrium values and policies for a recurrent class of points. Pakes and McGuire (forthcoming) provide a detailed definition of what we mean by this, and a testing procedure. They also consider; alternative starting values, different sequences for the α in (9), procedures for estimating policies at an s not in R, and a host of other details.

The Pakes McGuire (forthcoming) paper also provides numerical results for the model in the extended example in section 3. We increase $\overline{n}$ in that example by increasing market size. As we did so we found that the recurrent class of points did grow, but at most linearly in $\overline{n}$ with some indication of $\#R$ being concave in $\overline{n}$ at larger $\overline{n}$ [25]. As expected the computational burden per point grew linearly in the number of firms active at that point. The number of iterations required till convergence is a random variable in the stochastic algorithm. However we found that its distribution did not increase in any discernible way as we increased $\overline{n}$. Consequently to compute equilibria with $\overline{n} = 10$ on our dual processor Pentium 2 workstation, it took less than two hours of time for the actual iterations. This is the problem which we optimistically calculated could not be done in under ten days by point-wise techniques (see section 3). We also computed equilibria for problems with $\overline{n} = 12$ and for problems with two firm specific state variables, and in both cases the actual iteration time was under three hours. Note that even under the most optimistic assumptions, these latter problems would have taken years using the point-wise techniques discussed in section 3.

On the other hand Pakes and McGuire (forthcoming) used a conservative convergence test that required a point-wise calculation for continuation values, so the computational burden of the test did increase exponentially with $\overline{n}$. Accordingly by the time they computed the $\overline{n} = 10$ problem the computer was spending more than three times the time on performing the

[25] This generated market shares in the ergodic distribution that were reminiscent of those predicted by Sutton(1991).

tests than on doing the iterations leading to the test. It is clear that to use these techniques on larger problems we will have to employ more efficient testing procedures. One possibility is to substitute statistical for point-wise testing procedures.

A Note on the Computational Burden of the Stochastic Algorithm.

The economics of MP models typically indicate that a given set of primitives can only support certain configurations of firms in any lasting way. The lasting configurations become those in R. The relationship between $\#R$ and $\#S$ as we increase the number of state variables in the problem is different when the number of state variables increase because market size (and hence $\overline{n}$) increases then when the number of state variables per firm increases.

Though as market size increases the model will support structures with more active firms, it will no longer support structures where there are a small number of active firms. So as market size increases we both add and subtract points from R. Thus the *net* effect of market size on $\#R$ is not obvious, and will depend on the primitives of the problem. As noted we found the cardinality of R to grow in $\overline{n}$ in our problem, but only linearly at low $\overline{n}$, and if anything the rate *fell* as $\overline{n}$ increases. Recall that $\#S$ increases geometrically in $\overline{n}$.

When we increase the number of state variables per firm we have found that the relationship between $\#R$ and $\#S$ can vary greatly with the economics of the problem. For example in differentiated product models where the state vector details different characteristics of the products (e.g.. the size, mpg, and hp of cars), the primitives often indicate that certain combination of characteristics are not demanded at a price greater than their marginal cost of production (e.g.. large cars with a high mpg, or small cars with a low mpg). Alternatively in locational models in which there is an initial locational choice and then a plant specific cost of production (or quality of product) which changes over time as the result of outcomes of an investment process, $\#R$ tends to be linear in the number of locations at which their is entry.

We noted that the computational burden per point grows roughly as a linear function of the number of firms active. More precisely it grows linearly in the number of firms active after we have searched our storage device for the $w(\cdot)'s$ associated with the point we visit at each iteration. We have been using a trinary tree to store the $w(\cdot)$ and the computational burden of finding a point in such a tree grows like the logarithm of the number of points. So the burden per point grows as the product of the logarithm of $\#R$ and the number of firms active at that point. Again this is to be compared to exponential growth in the point-wise techniques, and a similar linear rate of growth in the polynomial approximations that use the moment generating function technique.

We also noted that the number of iterations in the stochastic algorithm neither needs to grow in any particular way in the dimension of the state space, nor seemed to grow in our numerical experiment. Probably more important is that in distinct contrast to the other algorithms considered in this paper (particularly the algorithm which uses polynomial approximations), we have *never* encountered a convergence problem with the stochastic algorithm.

Indeed part of the reason we have not pushed forward on the algorithm based on polynomial approximations is that we have had so much success with the stochastic algorithm. On the other hand the potential of the stochastic algorithm may be application specific. For example it will not be able to be used in problems in which behavior depends on the value of states that are off the equilibrium path (such as in some models of collusion) without a modification to sample off the equilibrium paths. One particular intriguing open question is the potential of the artificial intelligence techniques to compute equilibria in dynamic games with assymetric information.

References.

- Abreu, D., D. Pearce, and E. Stachetti (1986); "Optimal Cartel Equilibria with Imperfect Monitoring", *Journal of Economic Theory*, pp.251-269.

- Barto, AG, SI Bradtke and S Singh (1995); "Learning to Act Using Real-Time Dynamic Programming", *Artificial Intelligence*, Vol 72, pp 81-138.

- Benkard, L. (forthcoming); "Learning and Forgetting: The Dynamics of Aircraft Production", *NBER working paper, #W7127*, forthcoming in the *American Economic Review.*

- Benkard L, (2000); "A Dynamic Analysis of the Market for Wide-Bodied Commercial Aircraft", *NBER working paper, # 7710.*

- Berry S. and A. Pakes (in process); "Estimation from First Order Conditions" *mimeo* Harvard and Yale Universities.

- Berry S. and A. Pakes(1993); "Applications and Limitations of Some Recent Advances in I.O.: Merger Analysis.", *A.E.R., Papers and Proceedings*, pp.247-252.

- Berry, S., J Levinsohn, and A. Pakes (1999): "Voluntary Export Restraints: Evaluating a Strategic Trade Policy", *American Economic Review*, pp.400-430.

- Bertsekas, D. (1995); *Dynamic Programming and Optimal Control*, vol.s 1 &2, Athena Scientific Publications.

- Berstskas, D. and Tsiktsikilus, J (1996): *Neuro-Dynamic Programming*, M.I.T. Press.

- Cabral, L. and M. Riordan (1994); "The Learning Curve, Market Dominance, and Predatory Pricing", *Econometrica* pp. 1115-1140.

- Cheong, K. and K. Judd, (forthcoming); "Mergers and Dynamic Oligopoly", the *Journal of Economic Dynamics and Control.*

- Cole, H. and N. Kocherlakota (1998); "Dynamic Games with Hidden Actions and Hidden States", *Staff Report 254*, Federal Reserve Bank of Minneapolis.

- De Roos, N. (in process); "Essays on Collusion: Dynamic Models for Dynamic Markets", Ph.D. dissertation, Yale University.

- Doraszelki, U. (2000); "An R&D Race with Learning and Forgetting", *mimeo* Northwestern University.

- Ericson, Richard and Pakes, Ariel (1995): "Markov Perfect Industry Dynamics: A Framework for Empirical Work", *Review of Economic Studies*, pp.53-82.

- Fershtman C. and A. Pakes (2000); "A Dynamic Oligopoloy with Collusion and Price Wars", *RAND*, Vol. 31, pp. 294-326.

- Fershtman C. and A. Pakes (in process); "A Dynamic Game with Asymmetric Information", *mimeo*, Harvard University.

- Fudenberg, D. and D. Levine (1999); *Learning in Games*, Cambridge: MIT Press.

- Gowrisankaran, G. (1999); "A Dynamic Model of Endogenous Horizontal Mergers", *RAND* Vol.30, pp.56-83.

- Gowrisankaran, G. (1999a), "Efficient Representation of State Spaces for Some Dynamic Models", *Journal of Economic Dynamic and Control.* pp 1077-1098.
- Gowrisankaran, G. and R. Town (1997): "Dynamic Equilibrium in the Hospital Industry", *Journal of Economic Behavior and Organization.*
- Green, E. and R. Porter (1984); Non-cooperative Collusion Under Imperfect Price Competition" *Econometrica* pp.87-100.
- Jofre-Bonet M. and M. Pessendorfer (2000); "Bidding Behavior in a Repeated Procurement Auction", *mimeo*, Yale University.
- Judd, K. (1985); "Credible Spatial Preemption", *Rand*, pp.153-66.
- Judd K. (1990); "Cournot vs. Bertrand; A Dynamic Resolution", *mimeo* the Hoover Institution.
- Judd K. (1993); "Comment" in in J.J.Laffont and C.Sims (ed.s), *Advances in Econometrics, Proceedings of the Sixth World Congress of the Econometric Society*, Cambridge University Press, N.Y.
- Judd,K. (1999); *Numerical Methods in Economics*, M.I.T. press, Cambridge, Mass.
- Judd, K. , S. Yeltekin and J. Conklin (2000); "Computing Supergame Equilibria", *mimeo*, the Hoover Institution.
- Klepper, S. (2000); "Firm Survival and the Evolution of Oligopoly", *mimeo*, Carnegie Mellon University.
- Kortum, S. (1992); "Value Function Approximation in an Estimation Routine" *mimeo*, Boston University.
- Lettau, M. and H. Uhlig (1999); "Rules of Thumb versus Dynamic Programming", *American Economic Review*, pp 148-74.
- Liu, S. (1999); "Modeling Industrial Evolution In An Import-Competing Industry: With An Application To The Pulp And Paper Industry in Colombia", *Unpublished Ph.D. Dissertation*, Georgetown University.
- Mankiw N. and M. Whinston (1986); "Free Entry and Social Inefficiency", *Rand Journal of Economics*
- Marcovich S. (1998); "The Evolution of Dynamic Oligopolies with Network Externalities", *mimeo*, the University of Chicago,

- Marcovich S. (1999); “Small and Happy: Dynamic Oligopoly with Advertising”, *mimeo*, the University of Chicago.
- Maskin, E. and J.Tirole (1988a); “A Theory of Dynamic Oligopoly 1; Quantity Competition with Large Fixed Costs”, *Econometrica*, pp. 549-69.
- Maskin and Tirole, (1988b); “A Theory of Dynamic Oligopoly II; Price Competition, Kinked Demand Curves, and Edgeworth Cycles”, *Econometrica*, pp. 571-99.
- Maskin and Tirole, (1987); “A Theory of Dynamic Oligopoly III; Cournot Competition”, *European Economic Review* pp.947-968.
- Maskin, E. and J. Tirole (1995); “Markov Perfect Equilibrium, I: Observable Actions”, *mimeo*, Harvard University.
- Older, Allison, (1999); Chapter 2, *Unpublished Ph.D. Dissertation, the London School of Economics.*
- Pakes, A. (1993); “Dynamic Structural Models, Problems and Prospects”, in J.J.Laffont and C.Sims (ed.s), *Advances in Econometrics, Proceedings of the Sixth World Congress of the Econometric Society*, Cambridge University Press, N.Y.
- Pakes, A., S. Berry and J. Levinsohn (1993); ”Some Applications and Limitations of Recent Advances in Empirical Industrial Organization: Price Indexes and the Analysis of Environmental Change”, *AER, P&P* pp. 240–246.
- Pakes, A. and P. McGuire (1994); “Computing Markov Perfect Nash Equilibrium: Numerical Implications of a Dynamic Differentiated Product Model”, *RAND*, pp.555-589.
- Pakes A., and P.McGuire (1995); “Computing Markov-Perfect Nash Equilibria II: Approximations”, *mimeo.*, Yale University.
- Pakes A., and P. McGuire (forthcoming); “Symmetric Markov Perfect Equilibrium, Stochastic Algorithms, and the “Curse” of Dimensionality”, *Econometrica.*
- Pakes, A, G. Gowrisankaran, and P.McGuire (1995); “Code for implementing the Pakes-McGuire Algorithm for Computing Markov Perfect Equilibria”, *mimeo.*, Yale University.

- Petrin, A. (1999); "Quantifying The Benefits of New Products: The Case of the Mini-van", *mimeo*, GSB, the University of Chicago.
- Rust, J. (1994); "Structural Estimation of Markov Decision Processes," pp. 3082-3139 in the *Handbook of Econometrics*, R. Engle and D. McFadden (ed.s), North Holland.
- Schmalensee, R. (1978); "Entry Deterrence in Ready-to-eat Breakfast Cereals", *Bell Journal of Economics*, pp.305-27.
- Star, A.W. and Y.C. and Ho (1969); "Nonzero-Sum Differential Games", *Journal of Optimization Theory and Applications*, pp.1984-208.
- Stigler, G. (1968); *The Organization of Industry*, the University of Chicago Press,
- Sutton, J. (1991); *Sunk Costs and Market Structure*, M.I.T. Press.