ELSEVIER

Journal of Economic Dynamics & Control
24 (2000) 483–499

JOURNAL OF
Economic
Dynamics
& Control

www.elsevier.nl/locate/econbase

Learning the optimum as a Nash equilibrium

Süheyla Özyıldırım[a], Nedim M. Alemdar[b,*]

[a]*Department of Management, Bilkent University, 06533 Bilkent, Ankara, Turkey*
[b]*Department of Economics, Bilkent University, 06533 Bilkent, Ankara, Turkey*

Received 1 February 1998; accepted 1 February 1999

Abstract

This paper shows the computational benefits of a game theoretic approach to optimization of high dimensional control problems. A dynamic noncooperative game framework is adopted to partition the control space and to search the optimum as the equilibrium of a *k*-person dynamic game played by *k*-parallel genetic algorithms. When there are multiple inputs, we delegate *control authority* over a set of control variables exclusively to one player so that *k* artificially intelligent players explore and communicate to learn the global optimum as the Nash equilibrium. In the case of a single input, each player's *decision authority* becomes active on exclusive sets of dates so that *k GA*s construct the optimal control trajectory as the equilibrium of evolving *best-to-date* responses. Sample problems are provided to demonstrate the gains in computational speed and accuracy.

JEL: C61; C63; C73

Keywords: Learning; Optimal control; Nash equilibrium; Parallel genetic algorithms

1. Introduction

In this paper we offer a method that reduces the computational effort needed to numerically approximate high dimensional optimization problems. Our solution procedure uses a *divide and conquer* strategy. More specifically, using

* Corresponding author. Tel.: 90-312-266-48-07; fax: 90-312-266-5140.

0165-1889/00/$ - see front matter
PII: S0165-1889(99)00012-3

the fact that open-loop control trajectories in an optimal control problem can be reconstructed as the Nash equilibrium of a dynamic noncooperative game amongst players with identical preferences, we divide a high dimensional optimization problem into smaller subproblems and delegate each exclusively to an artificially intelligent player (*Genetic Algorithms*). *GA*s evolve their solutions locally according to the shared common objective while exchanging information globally. When further exploration and experimentation bring no additional benefit for any one player, the players will have found the system optimum as the game equilibrium. Our method reduces the complexity of computation at the local level leading to a faster learning and an improved accuracy at the global level.

In dynamic games, open-loop Nash equilibria can be obtained as joint solutions to optimal control problems (Başar and Olsder, 1982). Previously, we used this idea to exploit the 'explicit' parallelism in genetic algorithms to approximate Nash equilibria in discrete and continuous dynamic games amongst players with conflicting interests (Özyıldırım, 1997; Alemdar and Özyıldırım, 1998). This paper emphasizes the proximity of the concepts of Nash equilibrium and the optimum as learned behavior in environments lacking any explicit strategic interactions. In so doing, we bring forth the potential benefits from delegating exclusive decision authority to agents who *learn* to optimize; an important aspect of optimizing behavior which has not received due attention (Marimon, 1996). Analytically, *decentralized* decision making with a common objective is equivalent to a *centralized* decision making with the same objective. If optimum decisions evolve over time, however, as result of learning, there are sharp differences between the two modalities. How agents learn to optimize under two these regimes is the focus of this paper.

Optimizing behavior is an important tenet of positive economic analysis. Even though this belief permeates deep into economic thinking, the behavioral underpinnings of how the optimum, supposing that it exits in some well defined sense, is achieved by human action are not well articulated. There is now a growing recognition that the optimum human action, whether individually or as a group, is not best described by a timeless instantaneous inspiration, but rather as an evolutionary learning process by experimenting and exploring error-bound human actors (see Marimon et al., 1990; Arifovic, 1994). Our study makes contact with this literature to the extent learning by artificially intelligent agents in our genetic algorithms approximates human learning. Few results stand out: First, if human learning evolves in *implicit parallelism*, as do our genetic algorithms, that is, while manipulating individual objects, it in effect evolves *similarities* in groups of objects, then human actors can indeed learn such complex phenomena as intertemporal economic equilibria with fragile saddle path stability. Moreover, if learning takes place in a *decentralized* manner as with the artificial agents in our *parallel* genetic algorithms, then with a proper information exchange facility, in order for the diverse beliefs to converge on the

global optimum, it is sufficient that local agents *share the grand objective* and do their best at the local level. The *global problem* need not be known in its entirety by any one agent.

There are benefits of the method we propose from a purely computational point of view as well. The need to solve optimization problems frequently arises in one form or another in almost all fields of economic inquiry. Researchers' ability to carry out a meaningful economic analysis is often compromised due to the inherent difficulty of the problem at hand. For example, high dimensional nonlinear dynamic economic models do not easily yield to analytical treatment. In the face of intractability, soon stamina for rigor fatigues and recourse to numerical simulations remains as the only viable route to obtain insights about the inner workings of the system under study. Given the diversity of complex problems posed, we believe, our method offers a cheap, reliable, model independent numerical alternative.

The balance of the paper is organized as follows. In section two we discuss how straightforward optimization problems can be reworked as solving for game equilibria. This is followed by a brief introduction to genetic algorithms in section three. We then use the methods developed in section two to delegate control authority to genetic algorithms. In section four we consider two applications to demonstrate the computational gains that accrue from our solution method. Section five concludes with limitations and possible extensions.

2. Control problems as dynamic games

2.1. Multiple input control problems as dynamic games

Consider the following n-dimensional deterministic generic control problem in discrete time, $t \in T$:

$$x_{t+1} = F_t(x_t, u_{1t}, u_{2t}, \ldots, u_{nt}), \tag{1}$$

where $x_t \in X \subseteq \mathbb{R}^r$ and x_1 is given. The objective functional to be maximized is

$$J(x_{t+1}, x_t, u_{1t}, \ldots, u_{nt}) = \sum_{t=1}^{T} R_t(x_{t+1}, x_t, u_{1t}, \ldots, u_{nt}). \tag{2}$$

For an *open-loop* solution,[1] recursively substitute backward Eq. (1) into Eq. (2) to express J as a function of u_t:

$$\tilde{J}(x_1, u_t) = \sum_{t=1}^{T} \tilde{R}_t(x_1, u_t), \tag{3}$$

[1] For analytical solutions, usually optimal control methods are employed which requires continuity and the existence of derivatives of $F_t(.)$ and $R_t(.)$. For genetic implementations, however, restrictions on $F_t(.)$ and $R_t(.)$ are that they only be bounded.

where $u_t \in U_t \subseteq \mathbb{R}^n$. The globally optimal control vector, $u_t^* = \{u_{1t}^*, \ldots, u_{nt}^*\}$, satisfies the following inequality for all $t \in T$ and $u_t \in U_t$:

$$\tilde{J}(x_1, u_{1t}^*, \ldots, u_{nt}^*) \geq \tilde{J}(x_1, u_{1t}, \ldots, u_{nt}).$$

In order to distribute the computational effort to maximize (3), we rework the above optimization problem as a dynamic game. Towards that assign exclusive sets of n_i-control variables to k ($k \leq n$) players such that the ith player's control vector becomes $u_t^i \in U_t^i \subseteq \mathbb{R}^{n_i}$, $\sum_{i=1}^{k} n_i = n$, and $\bigcup_{i=1}^{k} U_t^i \subseteq \mathbb{R}^n$. Player i chooses u_t^i to maximize

$$\tilde{J}^i(x_1, u_t^1, \ldots, u_t^i, \ldots, u_t^k) = \sum_{t=1}^{T} \tilde{R}_t(x_1, u_t^1, \ldots, u_t^i, \ldots, u_t^k), \tag{4}$$

where the state equation (1) is eliminated by substitution.

The k-dimensional vector, u_t^{iN}, will constitute an open-loop *Nash equilibrium solution* of this game if,

$$\tilde{J}^i(x_1, u_t^{1N}, \ldots, u_t^{iN}, \ldots, u_t^{kN}) \geq \tilde{J}^i(x_1, u_t^{1N}, \ldots, u_t^i, \ldots, u_t^{kN}),$$

for all t,i, and $u_t^i \in U_t^i$. In particular, the globally optimal control vector, u_t^*, when partitioned conformably, will also fulfill the inequality

$$\tilde{J}^i(x_1, u_t^{1}*, \ldots, u_t^{i}*, \ldots, u_t^{k}*) \geq \tilde{J}^i(x_1, u_t^{1}*, \ldots, u_t^i, \ldots, u_t^{k}*),$$

for all t,i, and $u_t^i \in U_t^i$. Moreover, since, $\tilde{J}^i(x_1, u_t^{1}*, \ldots, u_t^{i}*, \ldots, u_t^{k}*) \geq \tilde{J}^i(x_1, u_t^{1N}, \ldots, u_t^{iN}, \ldots, u_t^{iN})$, for all i, $\{u_t^{1}*, \ldots, u_t^{i}*, \ldots, u_t^{k}*\}$ is the globally efficient Nash equilibrium solution. Consequently, k-parallel *GAs* can be implemented to maximize $\tilde{J}^i$s to search $\tilde{J}(u_t^*)$ as the game equilibrium.

2.2. Single input infinite horizon control problems as dynamic games

Our methodology is still valid and computationally helpful even if the control space includes only a single input, therefore not permitting partitioning of controls. Due to the open-loop nature of the control action, however, it may be more efficient to distribute the single control (or the state) trajectory 'time-wise' to a number of *GAs* and search it as a Nash equilibrium.

This idea is closely related to discretization of infinite horizon continuous time control problems. Numerical solutions necessarily require reformulating the problem into a discrete-time finite horizon approximation. Mercenier and Michel (1994) propose time aggregation along with steady-state invariance for discretization. The steady state invariance property is achieved by imposing consistency constraints on the joint formulation of preferences and accumulation equations which amount to some simple restrictions on the discount factor in the time aggregated model.

The extent to which a model is time aggregated depends on the trade-offs between the information conveyed along the transient path and the concomitant increase in the computational cost. Using our approach the computational burden can be distributed by partitioning the transient phase during which the control is optimized and designing parallel *GAs* to search for the optimum over the assigned partition. Thus, to capture convergence, sluggish transition dynamics can be countered by increasing the length of the transient phase without having to unduly increase the degree of time aggregation; or faced with rapid transient adjustments, the degree of time aggregation can be set smaller to offset otherwise costly approximation errors. In this manner, better approximation of the transient dynamics can be obtained at a minimal additional cost.

Formally, consider one input infinite horizon control problem, time aggregated in the following *á la* Mercenier and Michel (1994).

$$\max_{\{x(t_v),u(t_v)\}_{v=0}^{V}} J \quad \text{where } J = \sum_{v=0}^{V-1} \alpha_v \triangle_v R_v + \alpha_V S(x(t_V)) \tag{5}$$

$$\text{s.t. } x(t_{v+1}) - x(t_v) = \triangle_v F(x(t_v), u(t_v)), \quad x(t_0) = x_0 \quad \text{given},$$

where $R_v = R(x(t_v), u(t_v))$, $S(.)$ is the terminal state,[2] and at t_V, stationarity is assumed.

The scalar factor, $\triangle_v$, converts the continuous time flow into stock increments, i.e., $(\triangle_v = t_{v+1} - t_v)$. Intraperiod controls are assumed either constant or growing (decaying) at a constant rate therefore adding to approximation errors.

The choice of $\triangle$ reflects the trade-offs, at the margin, between the utility of better approximation and the increase in computational costs and thus determines the degree to which the model is aggregated. One may opt for a less aggregated model (smaller $\triangle$) and therefore avoid approximation errors, if at the same time, computational costs can be reduced. This we will show to be one of the benefits of explicit *GA* parallelism.

Time aggregated discount factor, α_v, satisfies the following recurrence relation:

$$\alpha_v = \frac{\alpha_{v-1}}{1 + \rho\triangle_v} \quad \text{for any } \alpha_0, \quad v > 0 \text{ and } \alpha_V = \alpha_{V-1}.$$

Again after eliminating the state evolution equation via recursive backward substitution into Eq. (5), the optimum solution, $u_{t_v}^*$, will satisfy,

$$\tilde{J}(x_0, u_{t_0}^*, u_{t_1}^*, \ldots, u_{t_{V-1}}^*) \geq \tilde{J}(x_0, u_{t_0}, u_{t_1}, \ldots, u_{t_{V-1}}),$$

[2] The value of the terminal state is given by $S(x) = \int_0^\infty e^{-\rho t} R(x, u(x))\, dt = \frac{1}{\rho} R(x, u(x))$ where ρ is the pure time preference in the continuous time analog of Eq. (5).

for all $v \in V$ and $u_{t_v} \in U_{t_v}$. In order to reformulate the above optimization problem as a game, identify a set of time intervals, v_i with players, i, $i = 1, 2, \ldots, k$ ($\sum_{i=1}^{k} v_i = V$) and define index sets I_i such that $v_i \in I_i$, $\bigcup_{i=1}^{k} I_i = \{1, 2, \ldots, V\}$. Then each player will be assigned a control vector, $u^i \in U^i \subseteq \mathbb{R}^{v_i}$, and a state vector, $x^i \in \mathbb{R}^{v_i}$, whose respective elements are the inputs and the states at the arbitrarily designated dates. The problem for the ith player is then

$$\max_{u(t_{vi}),x(t_{vi})} J^i \quad \text{where } J^i = \sum_{v_i \in I_i} R_{v_i} + \sum_{j \neq i} \sum_{v_j \in I_j} R_{v_j}. \tag{6}$$

$$\text{s.t.} \quad x(t_{v_i+1}) - x(t_{v_i}) = \triangle_{v_i} F(x(t_{v_i}), u(t_{v_i})), \quad \forall i \text{ and } v_i \in I_i.$$

The k-dimensional control vector, u_t^{iN}, is said to be the Nash equilibrium solution of this game if,

$$\tilde{J}^i(x_0, u^{1N}, \ldots, u^{iN}, \ldots, u^{kN}) \geq \tilde{J}^i(x_0, u^{1N}, \ldots, u^i, \ldots, u^{kN}), \quad \forall i,\ u^i \in U^i,$$

where $\tilde{J}^i$ is again obtained by substituting the relevant state constraints into Eq. (6). Note that the globally optimum control trajectory, $u_{t_v}^*$, $v \in V$ when partitioned as above will also satisfy the inequality

$$\tilde{J}^i(x_0, u^{1*}, \ldots, u^{i*}, \ldots, u^{k*}) \geq \tilde{J}^i(x_0, u^{1*}, \ldots, u^i, \ldots, u^{k*}), \quad \forall i,\ u^i \in U^i,$$

so that it is the globally efficient Nash equilibrium.[3]

3. Genetic algorithms and dynamic games

A basic *GA* consists of a set of iterative procedures whereby in each iteration, called a generation, a constant size population of candidate solutions or *structures* are maintained and evaluated to create a new set of potential solutions. Iterative procedures simulate natural evolution: at each generation the 'relatively good' solutions reproduce while 'relatively bad' solutions die. *GA*s essentially initiate a *blind* search, but they '*learn as they solve*'.

*GA*s owe their power and success largely to their *implicit parallelism*: While *GA*s operate on individuals in populations, they exploit the similarities in classes of individuals to filter and process vast amounts of information parsimoniously and effectively. These similarities in classes of individuals, which Holland calls

[3] Or more simply, if we interpret the index v as signifying players rather than the aggregated time, then Eq. (5) will be immediately recognized as a game between V players with identical objectives. Thus, each player v (date) on the globally optimum control trajectory can also be interpreted as in a Nash equilibrium of a game between intertemporally located players with a common interest, i.e.,

$$J^v(u^*(t_v), u^*(t_{-v})) \geq J^v(u(t_v), u^*(t_{-v})),$$

where $-v \neq v$ and $u^*(t_{-v}) = (u^*(t_0), \ldots, u^*(t_{v-1}), u^*(t_{v+1}), \ldots, u^*(t_V))$.

schemata, are defined by the lengths of common segments of bit strings. By manipulating n individuals in one generation, a GA effectively gathers information approximately about n^3 individuals (Holland, 1975). In addition, GAs provide for *explicit* parallelism in the sense that they can generate and collect data independently and that genetic operators can be implemented in parallel.

The advantages of parallel computing have been long recognized: reduced computing time and better approximation (Chazan and Miranker, 1970). Bertsekas and Tsitsiklis (1989) provide extensive discussions about the likely benefits from a numerical point of view. The potential power of parallelized genetic algorithms was first noted by Robertson (1987), Tanese (1989), and Mühlenbein (1989).

The main inspiration for parallel genetic algorithms comes from an analogy in biological evolution of species in isolated locales where local adaptation features prominently. To mimic this evolutionary process, a population is divided into subpopulations and a processor is assigned to each to separately apply genetic operators while allowing for periodic communication between them.

The particular division of population we envision delegates the computational effort needed for function evaluations to *exclusive* subpopulations. Subpopulations, though sharing the same 'grand objective', specialize on one portion of the problem and communicate among themselves to learn about the remainder. For each subpopulation the dimensionality of a complex objective function is thus reduced leading to faster and possibly to fewer function evaluations.

Our decentralized GA search for $\tilde{J}(u_t^*)$ takes place in discrete time, $t\ (=0,1,\ldots,T)$. Equipped with $\tilde{J}^i$s, at each generation s, GA^i maintains a constant size population, M, of solutions, $U^i(s)=\{u^i(s,1),\ldots,u^i(s,m),\ldots,u^i(s,M)\}$, where $u^i(s,m)\in U^i(s)\subset\mathbb{R}^{n_i T}$ is any feasible solution vector such that $\bigcup_{i=1}^{k}U^i(s)=U(s)$, and $\bigcup_{s=1}^{S}U(s)\subseteq U$. For player i, each potential solution vector is evaluated by computing $\tilde{J}^i(x_1,u^i(s,m),u^{-i}(s))$, *given* any solution vector, $u^{-i}(s)$, of the other players denoted by '$-i$'. At any generation s, players explore further if there exists an $m\in M$ such that

$$\tilde{J}^i(x_1,u^i(s,m),u^{-i}{}^*(s-1))>\tilde{J}^i(x_1,u^i(s,l),u^{-i}{}^*(s-1)),\quad \forall i\in k,$$

where $l\ (=1,2,\ldots,m-1,m+1,\ldots,M)$. The best performing control vectors, $u^{i*}(s)=u^i(s,m)$ are mutually exchanged, and genetic operators are applied to form the next generations. Because of the 'elitist selection strategy' employed by each player, the respective best performing structures survive intact into the next generations. If it were not for such a strategy, it may be that the best structures disappear due to mutation stagnating the search process. This procedure will continue with fitter individuals proliferating, thanks to reproduction and crossover operators, until $s'\le S$ whence *no* $m\in M$ exists such that for all i,

$$\tilde{J}^i(x_1,u^i(s'+1,m),u^{i*}(s'))>\tilde{J}^i(x_1,u^i(s'+1,l),u^{-i}{}^*(s'),\quad \forall l\neq m.$$

Therefore,

$$\tilde{J}^i(x_1, u^{i*}(s'), u^{-i*}(s')) \geq \tilde{J}^i(x_1, u^i(s'+1, m), u^{-i*}(s')) \quad \forall i \text{ and } m \in M$$

indicating convergence to the Nash equilibrium (see Appendix).

In essence, our algorithm searches the global optimum as the *evolutionary equilibrium* of *best-to-date responses* in a noncooperative dynamic game wherein players with identical fitnesses employ Nash strategies. The rationale behind searching for the global optimum as a Nash equilibrium is also evolutionary: smaller and more homogenous populations should evolve faster than larger and more complex populations. So, when the size and the complexity of the population is reduced by way of our game construction, the evolutionary process should speed up leading possibly to a more rapid convergence.

4. Applications

As the saying goes, the proof of the pudding is in the eating. So, to test our solution algorithm, first, we optimize a multiple input finite horizon control problem as a dynamic game between two players each with an identical objective and an exclusive decision authority over an assigned control vector. Second, we approximate a single input infinite horizon optimal control problem as the equilibrium of a game between two players with a common interest and a decision authority over the single control on exclusive dates.

4.1. Distributing controls in a nonrenewable resource model

To illustrate our method, we choose the nonrenewable resource model in Hughes Hallett et al. (1996). They use a hybrid algorithm combining the Newton method with Gauss–Seidel iterative techniques to solve the necessary conditions as a system of nonlinear equations. These equations are highly non-linear therefore presenting considerable numerical difficulties to standard gradient techniques in terms of initialization and convergence. Neither the single nor the parallel versions of *GAs* in any of our runs have shown any sign of nonconvergence. Moreover, relative to single *GAs*, our parallel *GA* searches have demonstrated considerable improved computational efficiency and performance. Admittedly, the true value of the algorithm can be better tested and appreciated with higher dimensional problems possibly with non-smooth functional forms.

The nonrenewable resource problem is as follows:

$$\max \quad W = \sum_{t=0}^{T-1} \frac{\log C_t}{(1+\theta)^t}$$

$$\text{s.t.} \quad S_{t+1} = S_t - R_t$$

$$K_{t+1} = K_t + F(K_t, R_t) - C_t, \quad t = 0, 1, \ldots, T-1,$$

where C_t stands for consumption and θ is the rate of pure time preference. S_t denotes the stock of nonrenewable resource, available oil reserves, with the extraction rate R_t. Together with the physical capital, K_t, R_t determines output according to the production technology, $F(K_t, R_t)$, that has a Cobb–Douglas specification: $AK_t^{0.9}R_t^{0.1}$. Since leaving any unexploited oil reserves and capital stock at the end of the fixed planning period, T, will be suboptimal, a backstop technology is assumed which becomes operational at T and remains so thereafter. This stipulation, however, is not further elaborated, and is immaterial to the optimization of T-period welfare.

We delegate control authority to two separate *GAs*, GA^K and GA^S, each having the same fitness function, to search for the optimal capital (K) and resource (S) stocks. GA^K evolves the population of candidate solutions, K_t, while GA^S iterates on the population of chromosomes, S_t. Structures $K_t(m)$ and $S_t(m)$ in each population ($m = 1,2,\ldots,50$) are represented as binary strings ($\{0\ 1\}$) of length L. For string m of length $L = 10$, decoding works as follows: $K_t(m) = \sum_{h=1}^{L} a_t^h(m)2^{h-1}$ and $S_t(m) = \sum_{h=1}^{L} a_t^h(m)2^{h-1}$ where $a_t^h(m)$ is the value $\{0\ 1\}$ taken at the hth position in the string.

In this particular example, initial states are given, and the time horizon is fixed at $T = 10$ so that each GA computes 9 structures with a domain, $D_i = [\underline{d}, \bar{d}\,] \subset \mathbb{R}$; $i = S, K$. The domain, D_i, is cut into $(\bar{d} - \underline{d})2^L$ equal size ranges. Hence, our parallel solution procedure has the search domain of $2 \times 2^{10 \times 9} = 2^{91}$ ($L = 10, k = 2, T - 1 = 9$). When the very same problem is approximated by a single GA, the search domain expands to $2^{10 \times (9+9)} = 2^{180}$.

The genetic operators in this paper were done using the public domain *GENESIS* package (Grefenstette, 1990) on a SUN SPARC-1000 running Solaris 2.5. In a typical run we use the *population size* of 50, the *crossover rate* of 0.60 and the *mutation rate* of 0.03.

For comparison we use single (centralized) and parallel (decentralized) *GA*s to approximate the model for three different number of generations. Since populations are randomly initialized, in each run computation times are random as well. For any given number of generation we repeat the experiment eight times and report our numerical results. First, to be assured of the convergence and to establish a benchmark, we run the experiment for hundred thousand generations for both the single and the parallel GA and report the average sample paths in Table 1. The convergence to a global optimum is apparent as all three methods generate the optimal trajectories with some minor differences.

Table 2 summarizes the averages and the variances of the maximum welfares found and the average CPU times consumed by the single and the parallel genetic algorithm for each number of generations. Reported discounted welfares are the averages of the best individuals found in the last generation of each run over the eight experiments. Variances indicate the deviations of these individuals from the calculated averages. Observe that our game theoretic GA on the average yields better approximations than the conventional GA in every case.

Moreover and more significantly, for every given number of generations, the decentralized *GA* finds these better results on the average in less than half the time the centralized *GA* takes. Note also that parallel *GA* has smaller variances indicating robustness of the results found.[4]

Fig. 1 depicts the gains in computational efficiency from decentralization in striking terms: The parallel Nash search accomplishes by twenty thousand

Table 1
Optimal trajectories in the nonrenewable resource model

Time	Hybrid algorithm		Parallel GA algorithm		Single GA algorithm	
	K_t	S_t	K_t	S_t	K_t	S_t
1990	4.913	11.5000	4.913	11.5000	4.913	11.5000
2000	19.231	9.2230	20.743	8.4773	20.743	8.4870
2010	66.383	7.2283	69.086	6.6748	67.743	6.8015
2020	203.889	5.5051	208.744	5.1063	204.715	5.2400
2030	560.433	4.0413	565.946	3.7691	556.546	3.8847
2040	1381.198	2.8247	1385.093	2.6296	1366.293	2.7188
2050	3038.662	1.8442	3028.759	1.7131	2993.844	1.7757
2060	5870.274	1.0812	5819.230	1.0054	5769.544	1.0514
2070	9502.490	0.5287	9372.449	0.4872	9310.677	0.5179
2080	10936.590	0.1724	10693.830	0.1585	10669.660	0.1696

Note: The parameter values are $A = 3.968$, $r = 0.05$, $K_0 = 4.913$, $S_0 = 11.5$.

Table 2
Comparison of performances

Number of generations	CPU times (s)		Discounted welfares		Variances	
	Single	Parallel	Single	Parallel	Single	Parallel
10,000	152.99	73.78	46.376	46.483	6.473E − 02	2.178E − 03
20,000	305.19	147.25	46.250	46.540	5.113E − 01	1.121E − 03
100,000	1528.70	726.43	46.520	46.530	1.131E − 03	1.331E − 03

Note : The maximum welfare under hybrid algorithm is 46.575.

[4] Eight experiments may seem too small a sample to make this statement statistically significant. However, since populations converge substantially for both algorithms at around ten thousand generations, variances get smaller for larger number of generations. Consequently, variances as a measure of efficiency can be a basis of comparison between the two algorithms for smaller number of generations.

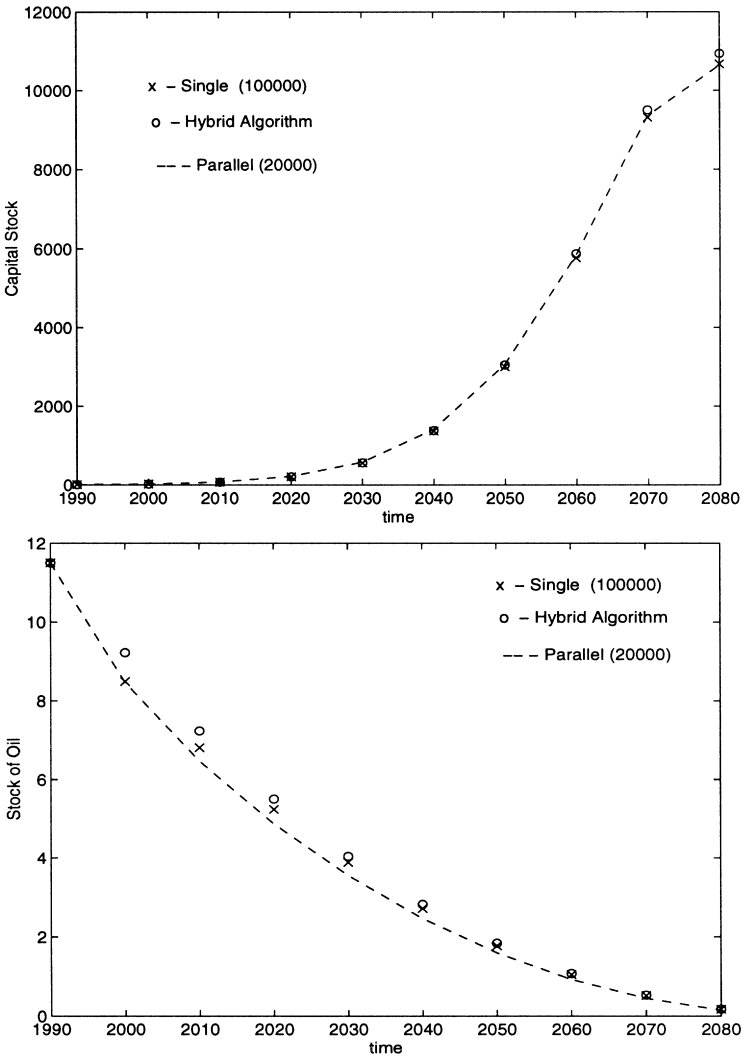

Fig. 1. Time paths of capital and resource stocks under different solution procedures.

generations in 147.25 s (2.454 min) what the conventional *GA* achieves by hundred thousand generations in 1528.70 s (25.478 min). Two features of our search procedure contribute to this significant computational gain: First, the fact that structures evolve independently improves local adaptation and hence yield better approximations; second, due to the specific character of the objects

searched, control specialization, the complexity of fitness evaluations are lessened and thus the speed.

4.2. Distributing time in an optimal growth model

In our search procedure we partition the original problem time-wise into smaller subproblems and distribute them to a number of parallelly running *GAs* to search the optimum as a game. Because of the reasons provided in the preceding section, convergence to the Nash equilibrium will also indicate convergence to the global optimum. We show that by using our algorithm the sample points on the transient phase of the state trajectory can be increased to obtain better approximations at a minimal cost.

In order to demonstrate the computational payoffs stemming from such a revamp, we consider a discrete time version of the one sector optimal growth model in Mercenier and Michel (1994). The formulation assumes logarithmic preferences, Cobb–Douglas production technology, no capital depreciation, constant population growth at rate g, and post terminal growth at rate g over an infinite horizon:

$$\max \quad \sum_{v=0}^{V-1} \alpha_v \triangle_v \log(C_{t_v}) + \alpha_V \log(C_{t_V})$$

$$\text{s.t.} \quad K(t_{v+1}) - K(t_v) = \triangle_v \frac{1}{1+g}(aK_{t_v}^b - C_{t_v} - gK_{t_v}),$$

where the time aggregation term is

$$\triangle_v = \frac{1+g}{g}[1-(1+g)^{t_v - t_{v+1}}]$$

with

$$\alpha_{v+1} = \frac{\alpha_v}{1+\rho\triangle_{v+1}} \quad \text{for any } \alpha_0 \text{ and } v < V-1,$$

and the terminal value is

$$\underbrace{\frac{\alpha_{V-1}}{\rho}}_{\alpha_V} \log(aK_{t_V}^b - gK_{t_V}).$$

Mercenier and Michel approximate this problem for eleven periods ($V = 10$), aggregating two hundred quarters, ($t_{v+1} - t_v = 20$). In order to reduce approximation errors owing to time aggregation, we choose smaller time intervals, ($t_{v+1} - t_v = 10$) to better capture adjustments along the transient phase. The

time horizon, V, is partitioned into two and the resulting subproblems are subsequently approximated as a two person noncooperative dynamic game. Since each GA will be searching for structures at different time segments of the 'same optimal path', the quality and the quantity of information about the respective search topologies is critically important to the learning processes of the GAs. We let GAs search for structures at alternate dates to enhance learning across search spaces.

To obtain respective fitnesses we first substitute the capital evolution equation into the utility function. Then, GA^{o} and GA^{e}, each equipped with the identical fitness, initiate separate searches for odd and even dated capital stocks, K_{v_1} ($I_1 = \{1,3,5,\ldots,19\}$), and K_{v_2} ($I_2 = \{2,4,6,\ldots,20\}$), respectively. Of course, as in the previous example, GAs explore their respective search spaces in a Nash manner; that is, they exchange, synchronize and apply genetic operators sequentially. Execution of the algorithm is similar to the previous example so we do not repeat the details here. In referring to the flowchart of Fig. 2, once it is understood that number of variables controlled by each player, $n_i T$, is now v_i ($v_1, v_2 = 10$), the rest follows.

Again, we run the experiment for three different number of generations, repeat each eight times, and report the numerical results for comparison. In parallel Nash searches, each GA has ten structures so that the total search domain is $2 \times 2^{10 \times 10} = 2^{101}$. Single GAs, on the other hand, will explore 20 structures with the domain $2^{10 \times (10+10)} = 2^{200}$.

In Table 3, we report the findings of our first experiment with 100,000 generations. From the numerical results, 200 quarters seem to be sufficient to capture transition dynamics as all three algorithms indicate convergence to the steady state. Also note that whether implemented in parallel or single GAs replicate the optimal trajectory with some minor differences. Furthermore, as previously claimed, considerable improvements in performance are observed with finer time aggregation: The maximum welfare found by Mercenier and Michel when $t_{v+1} - t_v = 20$ quarters is -142.241. When $t_{v+1} - t_v$ is set to ten, the conventional and parallel GA improve it to -127.8716. This result is further corroborated even with smaller number of generations as reported in Table 4. Also reported in Table 4 are the average run times. In terms of average CPU times, again the parallel GA takes about half as much time to find these better results verifying the returns from 'division of labor' and 'specialization' in the form of increased efficiency.

5. Concluding remarks

The utility of a game theoretic analysis when the underlying dynamics emerges as a result of the strategic interplay between economic agents with differing interests is apparent. A less obvious, but no less significant employment

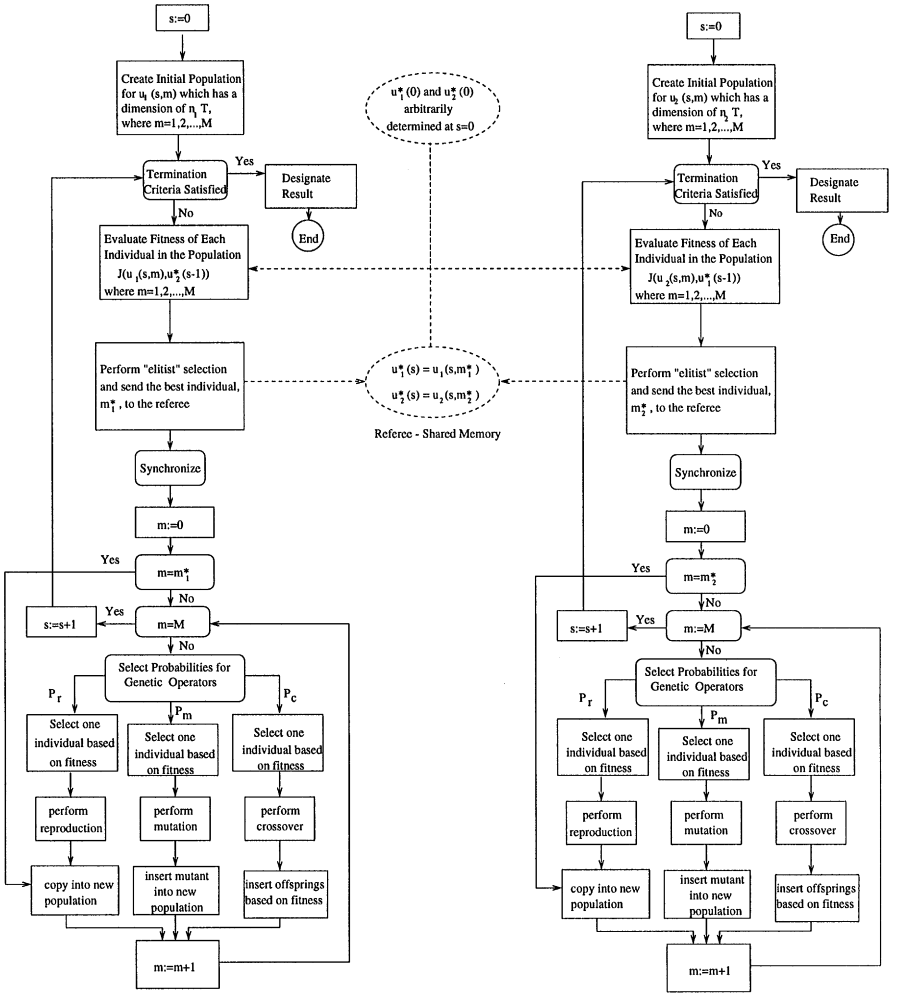

Fig. 2. Flowchart.

of the game theory can be in the computation of high dimensional optimization problems. Indeed one argument in this paper is that game theoretic concepts are still valid and very useful from the computational point of view, when the economic model under consideration is admittedly singular as in the traditional optimal control theory. For instance, a dynamic model with a single

Table 3
Timewise parallelization of the aggregated growth model

Mercenier and Michel			Parallel GA algorithm				Single GA algorithm		
t	t_v	K	t	t_v	K^{GA^o}	K^{GA^e}	t	t_v	K^{GA^s}
0	0	2.400	0	0	2.4000	–	0	0	2.4000
–	–	–	1	10	2.5601	–	1	10	2.5596
1	20	2.675	1	20	–	2.6442	2	20	2.6856
–	–	–	2	30	2.7855	–	3	30	2.7848
2	40	2.858	2	40	–	2.8374	4	40	2.8624
–	–	–	3	50	2.9242	–	5	50	2.9234
3	60	2.978	3	60	–	2.9560	6	60	2.9707
–	–	–	4	70	3.0089	–	7	70	3.0085
4	80	3.056	4	80	–	3.0282	8	80	3.0375
–	–	–	5	90	3.0605	–	9	90	3.0602
5	100	3.107	5	100	–	3.0723	10	100	3.0782
–	–	–	6	110	3.0921	–	11	110	3.0923
6	120	3.140	6	120	–	3.0994	12	120	3.1032
–	–	–	7	130	3.1114	–	13	130	3.1118
7	140	3.161	7	140	–	3.1158	14	140	3.1189
–	–	–	8	150	3.1236	–	15	150	3.1243
8	160	3.175	8	160	–	3.1265	16	160	3.1290
–	–	–	9	170	3.1317	–	17	170	3.1326
9	180	3.186	9	180	–	3.1338	18	180	3.1357
–	–	–	10	190	3.1380	–	19	190	3.1386
10	200	3.194	10	200	–	3.1398	20	200	3.1413

Note: The parameter values are: $a = 0.2$, $b = 0.24$, $\rho = 0.0125$, $g = 0.0075$, $\alpha_0 = 1$, $K_0 = 2.4$.

Table 4
Comparison of performances

Number of generations	CPU times (s)		Discounted welfares		Variances	
	Single	Parallel	Single	Parallel	Single	Parallel
10,000	187.22	90.09	− 127.8723	− 127.8717	2.507E − 07	0
20,000	373.46	173.08	− 127.8718	− 127.8717	5.002E − 09	2.137E − 09
100,000	1867.70	867.97	− 127.8716	− 127.8716	8.315E − 12	0

Note: The maximum welfare under aggregated model by Mercenier and Michel is − 142.241.

performance measure and multiple controls can be recast as a game between players with identical objectives each controlling an arbitrarily assigned set of controls. Obviously, there is no advantage in doing so from an analytical

viewpoint; however, when numerically implemented via genetic algorithms, the resulting computational gains are substantial. Also, the fact that an optimal openloop control trajectory consists of points which are in intertemporal Nash equilibrium can be utilized to distribute computational burden timewise to a number of parallel running genetic algorithms each searching a certain time segment of the optimal trajectory. Again, there are major benefits in the form of reduced computing time and better approximation.

The tenor of our discussions has been the practical computational advantages of the approximation of the optimum as the equilibrium of evolving (sub-optimal) 'best-to-date' responses. One interesting direction our results can be extended in is to focus on the learning aspects of such a decentralized search strategy and its implications for the optimizing behavior. When optimum decisions evolve over time, whether agents learn to optimize under a centralized or a decentralized regime gains significance. For example, we noted that when the decision authority was exclusively delegated to two local artificially intelligent agents with a common objective there was improvement in the quality and the speed with which decisions were undertaken. The efficiency gain was largely due to the reduction in the complexity of performance evaluations and task specialization leading to a more rapid learning throughout the locales. For a more general result, these benefits, however, have to be weighed against the communication costs that come with the decentralization. Therefore, at what point the net benefits from delegation of authority are exhausted depends on the available communication technology and remains open for further empirical research.

Another direction our parallel *GA* algorithm can be extended in is to explore closed-loop solutions in dynamic game models where players face time-inconsistencies. In neural network methods the log-sigmoid function is widely used to approximate unknown nonlinear functions (Sargent, 1993). In approximating time-consistent policy functions, our algorithm can be employed to evolve neural network architectures, each representing a player, in a parallel manner to obtain the time-consistent Nash equilibrium. At the present, we can only report that we are actively pursuing this idea further.

Acknowledgements

We are grateful to William A. Brock, Bülent Özgüler, Tarık Kara and İsmail Sağlam for helpful comments and suggestions. Özyıldırım acknowledges the financial support from Turkish Science Foundation (TÜBİTAK) while visiting University of Wisconsin-Madison and Harvard University. Responsibility for remaining errors and omissions are ours.

References

Alemdar, N.M., Özyıldırım, S., 1998. A genetic game of trade growth and externalities. Journal of Economic Dynamics and Control 22, 811–832.

Arifovic, J., 1994. Genetic algorithm and the Cobweb model. Journal of Economic Dynamics and Control 18, 2–28.

Başar, T., Olsder, G.J., 1982. Dynamic Noncooperative Game Theory. Academic Press, New York.

Bertsekas, D.P., Tsitsiklis, J.N., 1989. Parallel and Distribution Computation. Prentice-Hall, Englewood Cliffs, NJ.

Chazan, D., Miranker, W.L., 1970. A nongradient and parallel algorithm for unconstrained minimization. SIAM Journal of Control 8, 207–217.

Grefenstette, J.J., 1990. A User's Guide to GENESIS Version 5.0. Manuscript.

Hughes Hallett, A., Ma, Y., Yin, Y.P., 1996. Hybrid algorithms with automatic switching for solving nonlinear equation systems. Journal of Economic Dynamics and Control 20, 1051–1071.

Holland, J.H., 1975. Adaptation in Natural and Artificial Systems. University of Michigan Press, Ann Arbor.

Marimon, R., McGrattan, E., Sargent, T.J., 1990. Money as a medium of exchange in an economy with artificially intelligent agents. Journal of Economic Dynamics and Control 14, 329–373.

Marimon, R., 1996. Learning from learning in economics. Working papers, European University Institute.

Mercenier, J., Michel, P., 1994. Discrete-time finite horizon approximation of infinite horizon optimization problems with steady-state invariance. Econometrica 62, 635–656.

Mühlenbein, H., 1989. Parallel genetic algorithms population genetics and combinatorial optimization. In: Voigt, H.-M., Mühlenbein, H., Schwefel, H.-P. (Eds.), Evolution and Optimization. Akademie Verlag, Berlin.

Özyıldırım, S., 1997. Computing open-loop noncooperative solution in discrete dynamic games. Journal of Evolutionary Economics 7, 23–40.

Robertson, G., 1987. Parallel implementation of genetic algorithms in classifier systems. In: Davis, L. (Ed.), Genetic Algorithms and Simulated Annealing. Pittman, London.

Sargent, T.J., 1993. Bounded Rationality in Macroeconomics. Oxford University Press, Oxford.

Tanese, R., 1989. Distributed genetic algorithms. In: Schaffer, J.D. (Ed.), Proc. 3rd Internat. Conf. on Genetic Algorithms. Morgan Kaufmann, San Mateo, CA.