Constraint satisfaction on dynamic environments by the means of coevolutionary genetic algorithms

Hisashi Handa
Okayama University

Osamu Katai
Kyoto University

Tadaaki Konishi
Okayama University

Mitsuru Baba
Okayama University

Constraint Satisfaction on Dynamic Environments by the means of Coevolutionary Genetic Algorithms

H. Handa*   O. Katai   T. Konishi*   M. Baba***

*Dept. of Information Technology
Faculty of Engr, Okayama Univ.
Tsushima-naka 3-1-1, Okayama, JAPAN
handa@sdc.it.okayama-u.ac.jp

**Dept. of System Science
Graduate School of Informatics, Kyoto Univ.
Yoshida-Hommachi, Kyoto, JAPAN
katai@i.kyoto-u.ac.jp

Abstract

We discuss on adaptability of Evolutionary Computations in dynamic environments. Hence, we introduce two classes of dynamic environments which are utilizing the notion of Constraint Satisfaction Problems: changeover and gradation. The changeover environment is a problem class which consists of a sequence of the constraint networks with the same nature. On the other hand, the gradation environment is a problem class which consists of a sequence of the constraint networks such that the sequence is associated to two constraint networks, i.e., initial and target, and all constraint networks in the sequence metamorphosis from the initial constraint network to the target constraint network. We compare Coevolutionary Genetic Algorithms with SGA in computational simulations. Experimental results on above dynamic environments confirm us the effectiveness of our approach, i.e., Coevolutionary Genetic Algorithm.

1 Introduction

In this paper, we investigate the adaptive property of GAs in two kinds of dynamic environments, namely, changeover and gradation, which are utilizing the notion of Constraint Satisfaction Problems. The changeover environment is a problem class which consists of a sequence of the constraint networks with the same nature. On the other hand, the gradation environment is a problem class which consists of a sequence of the constraint networks such that the sequence is associated to two constraint networks, i.e., initial and target, and all constraint networks in the sequence metamorphosis from the initial constraint network to the target constraint network.

We adopt the notion of Constraint Satisfaction Problems (CSPs) [1] to construct dynamic environments. The notion of CSPs have been used to solve many practical problems, and have been studied by many researchers. We employ General CSPs, which is defined by two stochastic parameters, to make up instances of CSPs. Moreover, we introduce Dynamic CSPs as a sequence of static instances of General CSPs. By using a framework of DCSPs, there are several benefits for various fields, e.g., decision making problems in politics, economics, game playing and so on. If such problems are only formulated in static CSPs, we have to formulate all about simulation environment including internal states in the other agents or players. By using the framework, however, we only formulate a DCSP of own status and define effects from the other agents' policy or action. Also, some huge-scale problems may be easily solved by using the framework of DCSPs. The huge-scale problems are divided into several partial problems. Each of several partial problems is formulated as sub-DCSPs, and then defines constraints against adjacency but external nodes. Like as the islands-model [2], the best solutions in the previous interval are used as candidates of the external nodes. That is, we only concentrate on solving the part of a huge-scale problem at each of sub-DCSPs.

Related works are described as follows: Dynamic Constraint Satisfaction Problems in believe maintenance are discussed by Dechter and Dechter [3]. We adopt almost same formulation of the DCSPs as Dechter's. Coevolutionary approaches have been studied by many researchers [4] Especially, coevolutionary approach for solving Constraint Satisfaction Problems is proposed by Paredis [5]. He used two populations where an inverse fitness interaction, more precisely, the predator-prey relationship, between these populations is set. Also, schemata-oriented search methods in evolutionary computation have been adopted in several problem solving methods such as Cultural Algorithms and Stochastic Schemata Exploiter [7, 8]. In Cultural Algorithm, usual GA model is associated with a belief space that is similar to the schema space and is used to promote directed evolution of individuals in the GA model. Our method is similar to Cultural Algorithm in the sense that both methods use additional mechanisms to promote the evolution of usual GA. However, the specificity of our approach is to use a coevolutionary mechanism.

In next section, we describe about Constraint Satisfaction Problems with their difficulty indices. And then, we introduce our Coevolutionary Genetic Algorithms. Finally, several computational examinations on dynamic environments are carried out, and this paper is concluded.

2 Constraint Satisfaction Problems

Constraint Satisfaction Problems (CSPs) are a class of problems consisted of variables and constraints on the variables [1, 6]. In particular, a class of the CSPs such that each constraint in the problems is related only to two variables are called binary CSPs. In this paper, we treat a class of discrete binary CSPs, where discrete means that each variables in given problems are associated to a finite set of discrete labels. Especially, we introduce and examine General Constraint Satisfaction Problems used in following experiments. At first, we introduce two indices representing the characteristics of instances: tightness and density. The tightness T_{ij} of an arc ij denotes the proportion of existing constraint between two variables i and j, that is,

$$T_{ij} = \frac{\text{the number of all constraints on the arc } ij}{\text{the number of all compound-labels on the arc } ij}.$$

Further, the tightness of a problem is the average value of T_{ij} over all arcs. The density D of a problem indicates the proportion of constraint that actually exists between any pair of variables, i.e.,

$$D = \frac{\text{the number of all constraint-relations in the problem}}{\text{the number of all pairs of variables in the problem}}.$$

The general CSPs are randomly generated as follows: First, specify the tightness and density in the same sense above. Next, for all combination of two indices, decide whether unit constraint relation is set to each of the pairs of variables by taking account of the value of density. Finally, for all unit constraint relations, the number of the unit-label constraint relationship is set to be directly proportional to the tightness. Furthermore, we incorporate compound-labels, which denote a satisfiable solution, into each problem instance in order to guarantee at least one satisfiable solution in each problem instance exist.

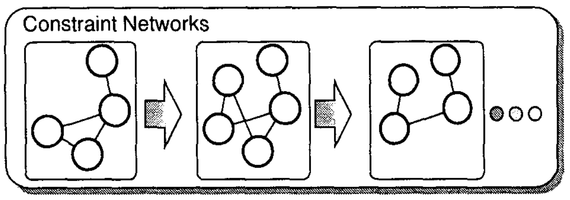

Figure 1: An instance of Dynamic Constraint Satisfaction Problems: At every interval C_t, the nature of instance is randomly changed.

2.1 Dynamic Constraint Satisfaction Problems

In this paper, Dynamic Constraint Satisfaction Problems are defined as a sequence of instances of static Constraint Networks. That is, at intervals of constant C_t, the nature of instances, such as the number of variables, the topology of unit constraint relations or unit-label constraint relation, and so on, is changed as delineated in Figure 1. By using this definition, we can improve the ability to represent instances associated to practical problems.

Hence, there are several ways to change the property of instances: As for changing the shape of constraint networks, add a node, delete a node, increase the size of a domain, and decrease the size of a domain is enumerated. Also, as for changing the topology of constraints in constraint networks, there are modifying constraint relations and modifying compound-constraints. In practice, all changes of constraint networks are represented by combining these ways. In the case of applying GAs to solve CSPs, we have to change the coding method in GAs according to the changes of the shape of constraint networks. In this paper, thus, we adopt only the modifying constraint relations as way to change the property of the constraint network.

3 Coevolutionary Genetic Algorithm

We adopt Coevolutionary Genetic Algorithm to solve Dynamic CSPs. As depicted in Figure 2, we have two GA populations: H-GA and P-GA. The H-GA is a traditional GA, in other words, it searches for good solutions in the given problem. In this paper,

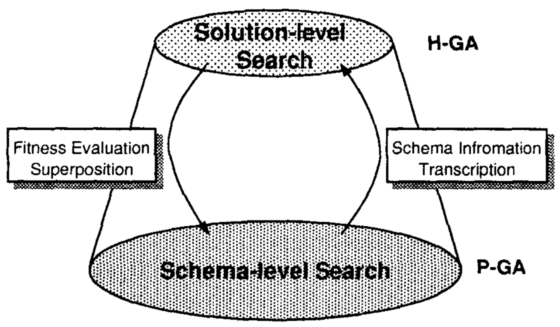
Solution-level Search
H-GA
Fitness Evaluation Superposition
Schema Infromation Transcription
Schema-level Search
P-GA

Figure 2: Process of Coevolutionary Genetic Algorithm

as the traditional GA, we use SGA including tournament selection, two point crossover and normal mutation. The P-GA searches for the good schemata in the H-GA. Each individual in P-GA consists of alleles of H-indiv. and "*", which are representing a schema in the H-GA. As depicted in the figure, two genetic operators, i.e., superposition and transcription, play the role to communicate (propagate) genetic information between the H-GA and the P-GA. The superposition operator copies the genetic information of a P-indiv., except for "don't care symbol" (denoted by "*" in the figure), onto one of H-Indiv.'s in order to calculate the fitness of the P-Indiv, where H-Indiv. and P-Indiv. denote an individual of H-GA population and an individual of P-GA population, respectively. The transcription operator serves as a mean of transmitting effective genetic information searched by the P-GA to the H-GA. For further details see also [9, 10]. Following subsection describes some explanation of an important part of our algorithm, i.e., fitness evaluation of P-Indiv.

In this section, several experimental results on Dynamic General Constraint Satisfaction Problems described below are examined.

4 Experimental Results

In this paper, we examine two kinds of dynamic environments: changeover and gradation. The changeover environment is a problem class which consists of a sequence of the constraint networks such that all constraint networks in the sequence have different constraint networks but are associated to the same parameters. On the other hand, the gradation environment is a problem class which consists of a sequence of the constraint networks such that the sequence is associated to two constraint networks, i.e., initial and target, and all constraint networks in the sequence metamorphosis their constraint each other structure and form a chain from the initial constraint network to the target constraint network.

4.1 Changeover Environment

As described in section 2, Dynamic General CSPs is represented by a sequence of the constraint networks associated to General CSPs. In the changeover environment, at every interval C_t, the nature of the constraint networks is changed. In this paper, the interval C_t is set to be 200000 times of fitness evaluations.

The Dynamic General CSPs are set such that both of the number of variables and the size of domain are set to be 10. The fitness function is defined as $1/(1 + \text{the number of violated constraints})$. In all experiments, the GA parameters for the SGA and the H-GA are set to be of the same value, the probability P_c of crossover, the probability P_m of mutation and the numbers of the elitist are set to be 0.8, 0.01 and 5, respectively. Those for the P-GA are set to be $P_c = 0.8$ and $P_m = 0.05$. Further, don't care symbol in the P-GA is generated with a high probability. Also, the number of runs for each of the tuple (*tightness*, *density*) is set to be 100.

Furthermore, when we try to solve CSPs by using GAs, there are two categories of changing property as follows:

KNOWN GAs can observe which constraint relations are changed. In this case, for changed variables, new alleles are inserted to corresponding gene locus.

UNKNOWN GAs cannot observe which constraint relations are changed. Note that we incorporate the case, such that GAs can observe which variables are changed but cannot know which variables are affected by constraint propagation from changed variable.

Figure 3 shows the experimental results on Dynamic General CSP, (*density*, *tightness*, C_t) = (70, 30, 200000). We examined comparisons on DGCSPs for a variety of the couple of *tightness* and *density*. For all couples of them, CGA outperforms SGA or performs roughly the same as SGA. This figure is one of typical results, where the horizontal axis, the vertical axis, and each lines in the all graphs denote the number of fitness evaluations, the Max fitness value in the population, the Max fitness value for a variety of the proportion of changed nodes. The left column and the right column in this figure denote *KNOWN* and *UNKNOWN* problems, respectively. The first and second

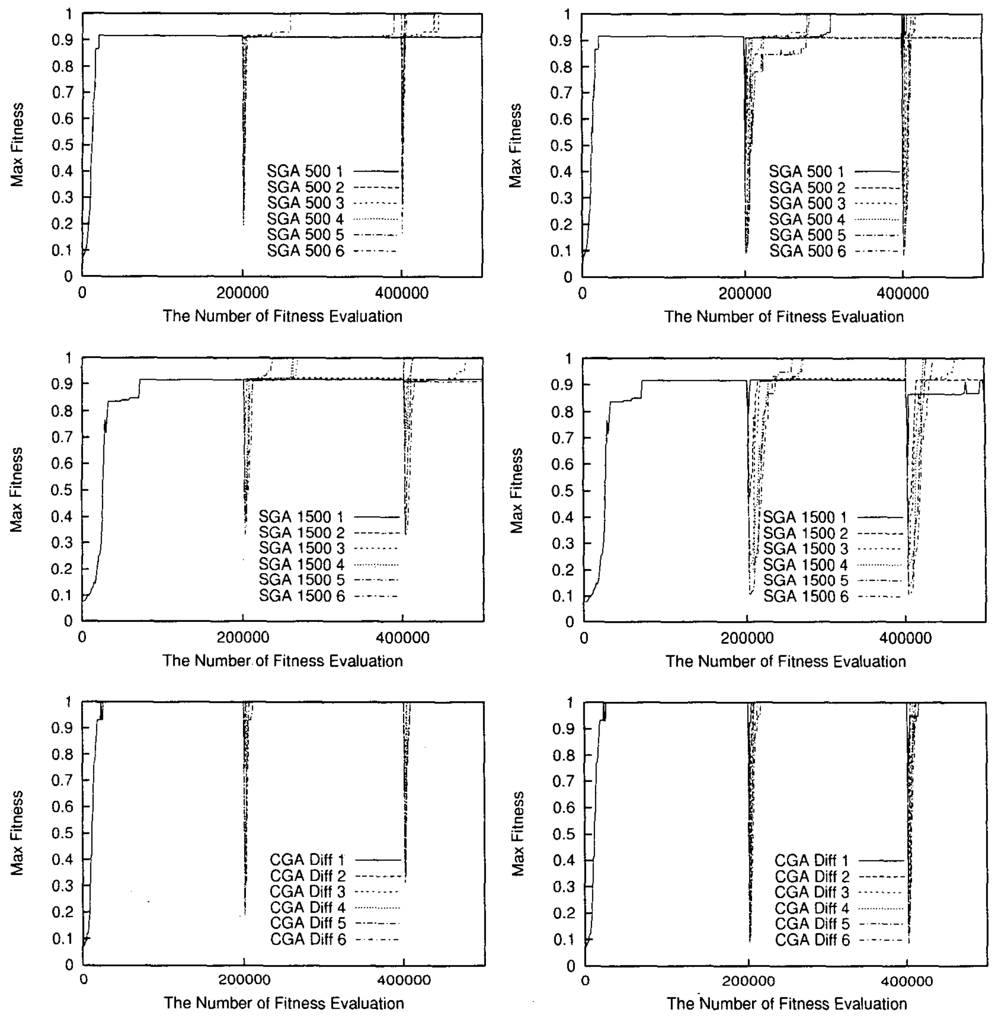

Figure 3: Experimental Results: Dynamic General Constraint Satisfaction Problems (*density*, *tightness*, C_t) = (70, 30, 200000); LEFT COLUMN:*KNOWN*, RIGHT COLUMN:*UNKNOWN*; 1st ROW:SGA–population size is 500, 2nd ROW:SGA–population size is 1500, 3rd ROW:CGA–H-GA's population size is 400 and P-GA's population size is 100

rows in the figure denote SGA such that the population size is 500 and 1500, respectively. The third row denotes CGA such that H-GA's is 400, and P-GA's is 100.

As depicted in this figure, CGA can search for new satisfiable solutions after environmental changes quickly. On the other hand, SGA cannot search for new satisfiable solutions effectively, if GAs don't know when environmental changes are occurred. It seems that such recovering ability of CGA results from P-GA, that is, P-GA serve as a means of the diversity preserving.

4.2 Gradation Environment

We also introduce two kinds of classes of DCSPs metamorphosing their constraint structure as the gradation environment. These classes use a target instance I of static CSPs and a control parameter C, which denotes the proportion of existential constraint, to represent an instance belonging each class. Detailed descriptions about the formalization and utilization of these classes are described as follows:

Subsuming
In this class, the nature of problem instances is

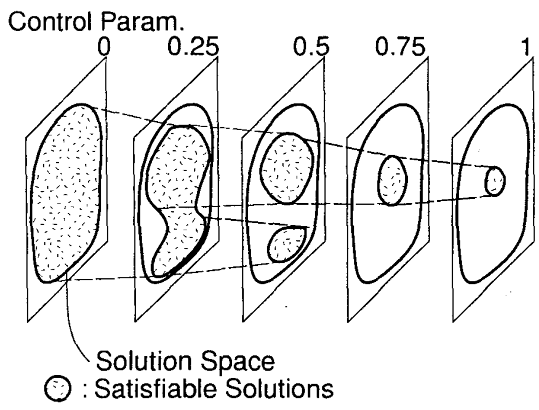

Figure 4: A conceptual depiction of the distribution of satisfiable solutions in solution spaces for each control parameter: Subsuming Problem

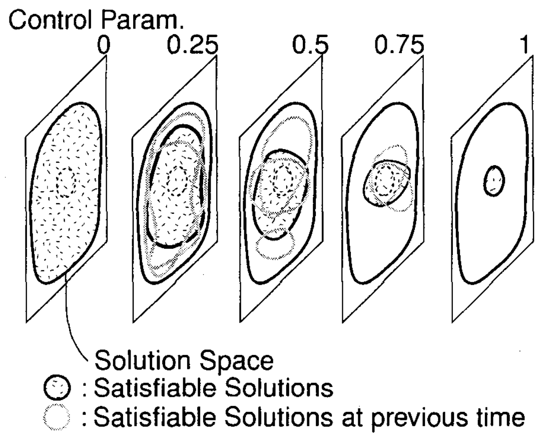

Figure 5: A conceptual depiction of the distribution of satisfiable solutions in solution spaces for each control parameter: Random-Changing Problem

changed in accordance with the control parameter. For each problem instance, unit constraint-relations are the same as the target instances'. An inhibited compound-label $(< i, a >, < j, b >)$ in certain constraint relations (i, j), which will be violated any constraints in the constraint relation (i, j), are associated real value $r_{<i,a><j,b>}$ randomly, where $0 < r_{<i,a><j,b>} < 1$ for any inhibited compound-lables $(<i, a>, <j, b>)$. Furthermore, constraint violation by assigning such inhibited compound-labels is neglected while associated real value is greater than the control parameter. That is, the number of constraint violations N^S_{cv} is calculated as follows:

$$N^S_{cv} = \sum Q^S((<i, x_i>, <j, x_j>), C),$$

where,

$$Q^S(t, C) = \begin{cases} 1 & \text{if compound label } t \text{ violates constra-} \\ & \text{int and } r_t < C \\ 0 & \text{otherwise,} \end{cases}$$

and x_i and x_j respectively denote label of variable i and j assined by individual. As depicted in Figure 4, all solutions become as satisfiable solutions when $C = 0$. On the other hand, when $C = 1$, satisfibale solutions in the problem instance is the same as the ones in targets instance. Furthermore, the set of satisfiable solutions in a lower control parameter subsumes the set of ones in a higher control parameter.

Random-Changing

The control parameter in this class is regarded as a threshold in order to decide whether inhibited compound-labels are activated or not probabilistically. By using a random variable v_r with uniform distribution in $[0 : 1]$, the number of constraint violations N^R_{cv} is calculated as follows:

$$N^R_{cv} = \sum Q^R((<i, x_i>, <j, x_j>), C),$$

where,

$$Q^R(t, C) = \begin{cases} 1 & \text{if compound label } t \text{ violates constra-} \\ & \text{int and } P\{v_r < C\} \\ 0 & \text{otherwise,} \end{cases}$$

In other word, all inhibited compound-labels in a problem instance are activated with probability $P\{v_r < C\}$. Thus, if C=0, any inhibited compound-labels in the problem instance are not activated. Further, if C=1, all inhibited compound-labels are activated, namely, the problem instance is the same as the target problem instance. Moreover, as depicted in Figure 5, the expected size of the set of satisfiable solutions is getting small as the control parameter increases. The sets of satisfiable solutions in any problem instance involve the set of satisfiable solutions in the target instance.

We examine two DCSPs mentioned above, i.e., subsuming and random-changing. In our experiments, the control parameter C is incremented if the best fitness in GA population exceeds 0.9 that means the best individual satisfies 90 percent of constraints for a current problem instance. Furthermore, initial value and step size of the control parameter C are set to be 0.7 and 0.04, respectively. Experimental results for these problems are shown in Figure 6. In the figure, left side and right side denote results for the subsuming problem and the random-changing problem, respectively. Left graphs in the figure delineate the success ratio

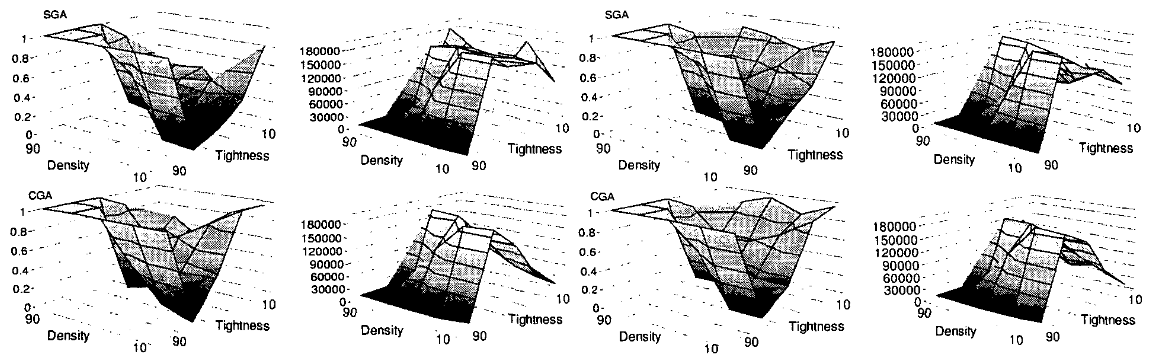

Figure 6: Experimental results on the subsuming problems (left side) and the random-changing problems (right side). The instances of General CSPs for various couples of (*tightness*, *density*) is adopted as the target instances: the success ratio (LEFT) and the number of fitness evaluations (RIGHT); SGA (UPPER), CGA (LOWER)

over 20 runs for each of *tightness* and *density*, and right graphs indicate the average value of the number of fitness evaluations until a satisfiable solution is found over 20 runs. Upper and lower rows denote SGA and CGA. These graphs are results of the target instances for various couples of (*tightness*, *density*). In tighter problems, the CGA can solve these problems quite effectively. It is important to find the partial satisfiable solutions as quickly as possible in such problems. The PGA serves as finding such partial solutions. In graphs of the success ratio, the murky part of surface curves in the SGA is the widest of them.

5 Conclusion

In this paper, we examined two dynamic environments, namely, changeover and gradation, generated by using Dynamic Constraint Satisfaction Problems. Experimental simulations on these dynamic environments confirmed us the effectiveness of Coevolutionary Genetic Algorithms. That is, in the changeover environment, the CGA could rediscover new satisfiable solutions quickly. Besides, in the gradation environment, the CGA outperformed SGA for tighter target problems.

References

[1] E. Tsang, *Foundation of Constraint Satisfaction*, Academic Press, 1993.

[2] D. E. Goldberg *Genetic Algorithm in Search, Optimization, and Machine Learning*, Addison-Wesley, 1989.

[3] R. Dechter and A. Dechter, "Belief Maintenace in Dynamic Constraint Networks", *Proc. of the 11th IJCAI*, Vol. I, pp.37-42, 1989.

[4] W. D. Hills, "Co-Evolving Parasites Improve Simulated Evolution as an Optimization Procedure", *Artificial Life II*, pp.313-324, 1992.

[5] J. Paredis, "Co-evolutionary Constraint Satisfaction", *Proc. of PPSN III*, pp.46-55, 1994.

[6] E. Marchiori, "Combining Constraint Processing and Genetic Algorithms for Constraint Satisfaction Problems", *Proc. of the 7th ICGA*, pp.330-337, 1997.

[7] R. G. Reynolds and C. Chung, "A Self-adaptive Approach to Representation Shifts in Cultural Algorithms", *Proc. of the 3rd ICEC*, pp.94-99, 1996.

[8] A. N. Aizawa, "Evolving SSE: A Stochastic Schemata Exploiter", *Proc. of the 1st ICEC*, pp.525-529, 1994.

[9] H. Handa, K. Watanabe, O. Katai, T. Konishi and M. Baba, "Coevolutionary Genetic Algorithm for Constraint Satisfaction with a Genetic Repair Operator for Effective Schemata Formation", *Proc. of the 1999 IEEE SMC Conference*, Vol. III, pp.617-621, 1999.

[10] H. Handa, O. Katai, T. Konishi and M. Baba, "Coevolutionary Genetic Algorithms for Solving Dynamic Constraint Satisfaction Problems", *Proc. of the 1999 GECCO*, Vol. I, pp.252-257, 1999.

[11] H. Handa, O. Katai, T. Konishi and M. Baba, "A Study of the Adaptability of Coevolutionary GAs in the Case of Metamorphosing Constraint Structure", *Proc. of the 2nd APGA*, pp.308-314, 2000.