Engineering

Industrial & Management Engineering fields

Okayama University *Year* 2000

A new fitness function for discovering a lot of satisfiable solutions in constraint satisfaction problems

Hisashi Handa
Okayama University

Osamu Katai
Kyoto University

Tadataka Konishi
Okayama University

Mitsuru Baba
Okayama University

A New Fitness Function for Discovering a lot of Satisfiable Solutions in Constraint Satisfaction Problems

Hisashi Handa+ **Osamu Katai*** **Tadataka Konishi**+ **Mitsuru Baba**+

+Department of Information Technology
Faculty of Engineering
Okayama University
Tsushima-naka 3-1-1, Okayama 700-8530, JAPAN
{handa, konishi, baba}@sdc.it.okayama-u.ac.jp
+81-86-251-8250

*Department of System Science
Graduate School of Informatics
Kyoto University
Yoshida-Honmachi, Kyoto 606-8501, JAPAN
katai@i.kyoto-u.ac.jp
+81-75-753-5201

Abstract- In this paper, we discuss how many satisfiable solutions Genetic Algorithms can find in the problem instance of Constraint Satisfaction Problems at single execution. Hence, we propose a framework of a new fitness function which can apply to traditional fitness functions. However the mechanism of proposed fintess function is quite simple, several experimental results on a variety of instances of General Constraint Satisfaction Problems demonstrate the effectiveness of proposed fitness function.

1 Introduction

In this paper, we discuss how many satisfiable solutions Genetic Algorithms can find in the problem instance of Constraint Satisfaction Problems (CSPs) at single execution. The notion of CSPs is a one of the most general and useful frameworks in Artificial Intelligence and has been studied by many researchers. Recently, several genetic approaches for solving CSPs were proposed [handa98, march97, riff96, riff97]. In these approaches including ours, a main topic of these studies is to devise new genetic operators or new fitness evaluations to find satisfiable solution effectively. From the viewpoint of constraint satisfaction, however, it is also desirable that Genetic Algorithms can find larger number of satisfiable solutions in single execution. Hence, in this paper, we propose a new fitness function in order to discover a lot of satisfiable solutions in Constraint Satisfaction Problems. The idea of proposed fitness function is very simple: for satisfiable solutions which are already discovered by GAs, discounted fitness value is assigned. In spite of such simple definition, several experimental results on a variety of instances of General Constraint Satisfaction Problems described in section 4.2 will demonstrate the effectiveness of proposed fitness function. Besides, proposed fitness function has the dynamic property such that fitness landscape is changed whenever GAs are finding novel satisfiable solutions. If the population of GAs keeps the rich diversity, GAs can follow the environmental change, that is, GAs can find other satisfiable solutions effectively. Thus, we adopt our Coevolutionary GA proposed in [handa97, handa98, handa99], which indicate the excellent performance for such environment.

Related works are described as follows: Tsang wrote comprehensive text book about CSPs which is also written about genetic approach for solving CSPs [tsang93]. Eiben summarized how to solve several classes of CSPs by using GAs in [eiben94, eiben95]. Riff proposed a fitness function and genetic operators for solving CSPs effectively, which are utilizing the knowledge with regard to the constraint network [riff96, riff97]. In this paper, we adopt her fitness function as one of standard fitness functions for solving CSPs.

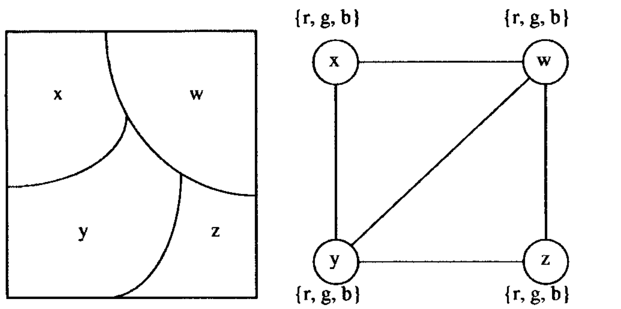
In next section, we introduce the formalization of Constraint Satisfaction Problems, General CSPs used in our experiments, and traditional fitness functions to solve CSPs. Then, we propose a new fitness function in order to discover a lot of satisfiable solutions. In Section 4, several computer simulations are examined and confirm us effectiveness of our fitness function, and finally, this paper is concluded.

2 Constraint Satisfaction Problems

2.1 Formalization of CSPs

Constraint Satisfaction Problems (CSPs) are a class of problems consisted of variables and constraints on the variables [tsang93, march97]. Especially, 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. An example of the graph coloring problem [mint94], one of binary CSPs, which is one of the benchmark problems in CSP is delineated in Figure 1. As depicted in the figure, CSPs are defined by (U, L, T, R): U, L, T and R denote set of units, set of labels, unit constraint relations and unit-label constraint relations, respectively. In this 3-coloring problem, i.e., coloring with three colors, r, g and b, for instance, the set U of units consists of the nodes in the graph of a given problem. The elements

0-7803-6375-2/00/$10.00



(a) Map representation (b) Graph representation of the CSP (a).

- Set of Units : U = {x, y, z, w}
- Set of Labels : L = {r, g, b}
- Unit Constraint Relations : T = {t_1, t_2, t_3, t_4, t_5}
  t_1=(x,y), t_2=(y,z), t_3=(z,w), t_4=(x,w), t_5=(y,w)
- Unit-Label Constraint Relations : R = {R_1, R_2, R_3, R_4, R_5}
  R_1 = {(r,g), (r,b), (g,r), (g,b), (b,r), (b,g)}
  R_2 = {(r,g), (r,b), (g,r), (g,b), (b,r), (b,g)}
  R_3 = {(r,g), (r,b), (g,r), (g,b), (b,r), (b,g)}
  R_4 = {(r,g), (r,b), (g,r), (g,b), (b,r), (b,g)}
  R_5 = {(r,g), (r,b), (g,r), (g,b), (b,r), (b,g)}

Figure 1: An example of CSP: graph coloring problem

in the set L of labels denote three colors that should be coloring. The unit constraint relations T mean as the edge in the graph of the given problem. The unit-label constraint relations R are set of 2-compound labels that the constraints are existing. To solve CSPs is to search for solutions such that no constraints are violated. Also, the graph representation of CSP in Figure 1(b) called Constraint Network is often used.

2.2 General Constraint Satisfaction Problems

In this section, we introduce 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,

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

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.,

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

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. The average number of satisfiable solutions over 10 problem instances associated with each density and

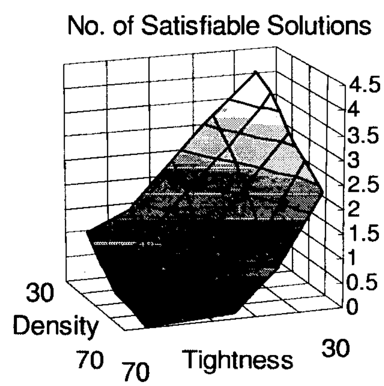


Figure 2: The average number of satisfiable solutions for each density and tightness

tightness is delineated in Figure 2. The front and depth axises in this figure denote the variety of the tightness and the density, respectively. Furthermore, the vertical axis denotes the number of satisfiable solutions for each couple of (tightness, density) and indicates a logarithm axis.

2.3 Fitness Functions to Solve CSPs

In this paper, we examine three kinds of fitness functions used in common, i.e., constraint violation, constraint relations and tightness. These fitness functions are described as follows:

- **constraint violation (CV)** This fitness function F_{cv} is caculated by accumlating the number of constraint violations in chromosome as follows:

$$F_{cv} = N_{cv},$$

where, N_{cv} denotes the total number of constraint relations in certain instance and the number of constraint violations in chromosome, respectively.

- **constraint network (CN)** This fitness function F_{CN} is originally proposed by Riff [riff96, riff97]. When a constraint relation is violated in certain chromosome, labels assigned to variables relevant to the constraint relation will change to other labels in order to satisfy the constraint relation. Then, some satisfied constraint relations with respect to relevant variables may not be satisfied by the change of labels. This fitness is designed by taking account into such effect. In this function, the propagation effect $e(C_\alpha)$ of constrain relation C_α is defined as the number of constraint relations with respect to relevant variables for C_α. For each constraint relation C_α in a problem instance, propagation effect $e(C_\alpha)$ is calculated in advance. Then,

$$F_{CN} = \sum_{\gamma \in V_{violated}} e(C_\gamma),$$

where, $V_{viorated}$ denote the set of violated constraint relations.

- **Information Content (IC)** This fitness function is evaluated by taking account into the difficulty to satisfy constraints. This difficulty is defined by the formula of information content, that is, let I_{ij} is the difficulty to satisy a constraint relation C_α. Then,

$$I_{C_\alpha} = N_\alpha^{inhibited} \ / \ N_\alpha^{all},$$

where, $N_\alpha^{inhibited}$ and N_α^{all} denote the number of compound-labels that violate constraints in constraint relation C_α and the number of all compound-labels defined in the constraint relation C_α. Besides, we define this fitness as follows:

$$F_{IC} = \sum_{\gamma \in V_{violated}} I_{C_\gamma}.$$

Also, in this paper, we adopt following translation function from minimization function to maximization function for above fitness functions: suppose F is one of above three fitness functions, and F_{worst} denote the worst value of F. Then, the translation function g is defined as $g = 1 - F/F_{worst}$. Hence, satisfiable solutions are assigned fitness value 1.

3 Fitness Function to Discover a Lot of Satisfiable Solutions

In order to search a lot of satisfiable solutions, we propose a new fitness function which idea is very simple: to avoid to convergence certain satisfiable solutions, satisfiable solutions which are already discovered by GAs receive discounted fitness value. Suppose F and $\hat{F}$ denote a fitness function to solve CSPs and proposed fitness function, respectively.

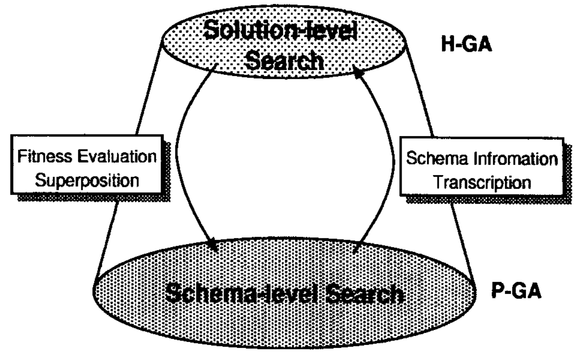


Figure 3: Process of Coevolutionary Genetic Algorithm

$$\hat{F} = \begin{cases} F - \beta E_{max} & \text{if chromosome indicates satisfiable solution and is already discovered,} \\ F & \text{otherwise,} \end{cases}$$

where, β and E_{max} denote denote the coefficient fixed in advance, and maximum effect when certain constraint is violated. By penalizing fitness value for satisfiable solutions which are already discovered, GAs can effectively search for neighborhood of satisfiable solution discovered lately. CSPs are formalized as the combinations of labels among variables, and constraints in CSPs are defined on compound-labels. Therefore, if certain satisfiable solution is found, neighbor solutions of the satisfiable solution may often be satisfiable solutions. That is the reason why we introduce such fitness function.

4 Computational Simulations

Proposed fitness function has the dynamic property such that fitness landscape is changed whenever GAs are finding novel satisfiable solutions. Thus, we adopt Thus, we adopt our Coevolutionary GA proposed in [handa97, handa98, handa99], which indicate the excellent performance for such environment. Following section describle the concept of Coevolutionary GA.

4.1 Coevolutionary GAs

As depicted in Figure 3, 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, 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

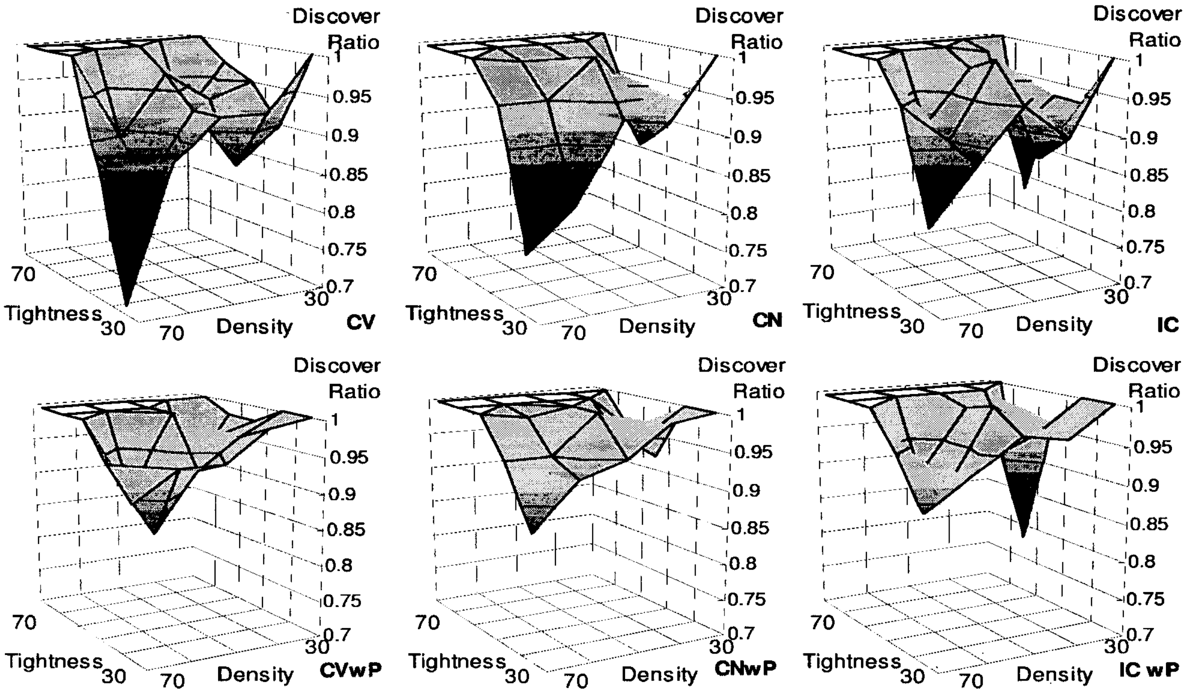


Figure 4: Experimental Results – Discovering-ratio: traditional fitness functions (UPPER; Constraint Violation **CV**, Constraint Network **CN**, and Information Content **IC**); Proposed fitness functions (LOWER, **CVwP**, Network **CNwP**, Information Content **ICwP**)

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.

4.2 Experimental Results

In this section, several experimental results on General Constraint Satisfaction Problems described in section 2.2 are examined. Especially, we investigate the effectiveness of proposed fitness function in the term of two indices: discovering-ratio and the number of fitness evaluations until finding all satisfiable solutions. The discovering-ratio means the ratio of the number of satisfiable solutions that GAs can find in certain problem instance at single execution, to the number of satisfiable solutions in the problem instance. We examine a variety of pairs of tightness and density. Both of them separately vary from 30 to 70 and adopt the GCSPs with 10 variables and five labels for each variable as the benchmark problem. The number of fitness evaluations on every execution is limited to 600,000. The maximum number of satisfiable solutions in the GCSPs with 10 variables and five labels is about 50,000. We consider it is sufficient to find all satisfiable solutions. Also, the vertical axis of all graphs in Figure 4 and 5 indicates a logarithm axis. Experimental results for discovering-ratio are delineated in Figure 4. In the figure, upper graphs indicate the traditional fitness functions explained in section 2.3: Constraint Violation (RIGHT), Constraint Network (MIDDLE), and Information Content (LEFT), respectively. Lower graphs indicate proposed fitness functions for each upper one. Every proposed fitness function improves the discovering-ratio of traditional's dramatically. For pairs of tightness and density such that tightness + density = 100, the surface curves of discovering-ratio declines. As mentioned in paper [handa98], it is difficult to solve the instances of GCSPs for such pairs. Such instances have almost 10 or 100 satisfiable solutions as depicted in Figure 2. Therefore, the landscape of such instances may be multimodal.

Figure 5 shows the results for the number of fitness evaluations until finding all satisfiable solutions. Note that the orientation and position of axis in this figure are distinct from the graphs in Figure 4. However, the location of graphs in this figure is the same as Figure 4. For all proposed fitness functions, the number of fit-

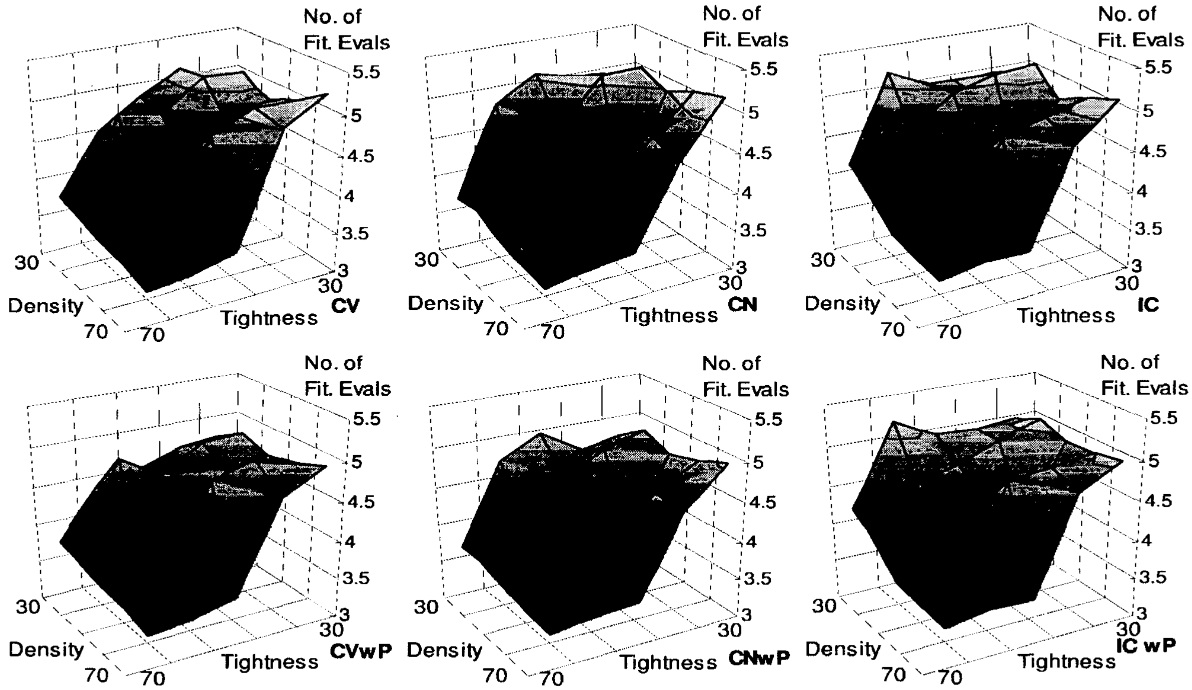


Figure 5: Experimental Results – the number of fitness evalutaions until finding all satisfiable solutions: traditional fitness functions (UPPER; Constraint Violation **CV**, Constraint Network **CN**, and Information Content **IC**); Proposed fitness functions (LOWER, **CVwP**, Network **CNwP**, Information Content **ICwP**)

ness evaluations until all satsifiable solutions decreases. The difference between traditional and prposed fitness functions are quit significant since the vertical axis of these graphs indicates a logarithm axis as we mentioned above.

5 Conclusions

We investigated how many satisfiable solutions Genetic Algorithms can find in the problem instance of Constraint Satisfaction Problems at single execution. Hence, we proposed a framework of a new fitness function which can apply to traditional fitness functions. The mechanism of proposed fintess function was quite simple. In spite of such simple definition, several experimental results on a variety of instances of General Constraint Satisfaction Problems demonstrate the effectiveness of proposed fitness function in the term of discovering-ratio and the number of fitness evaluations until finding all satisfiable solutions.

Acknowledgments

We would like thank Mr. H. Kimura for his kindful support.

Bibliography

[eiben94] Eiben, A. E., Raué and Ruttkay, Zs. (1994) "Solving Constraint Satisfaction Problems Using Genetic Algorithms," Proceedings of the 1st International Conference on Evolutionary Computation, Vol.II, 542-547.

[eiben95] Eiben, A. E., Raué and Ruttkay, Zs. (1995) "How to Apply Genetic Algorithms to Constrained Problems," Practical Handbook of Genetic Algorithms, Chapter 10, 307-365, Lance Chambers Editors, CRC Press.

[fogel95] Fogel, D. B. (1995) "Evolutionary Computation," IEEE Press.

[gold89] Goldberg, D. E. (1989) "Genetic Algorithm in Search, Optimization, and Machine Learning," Addison-Wesley.

[handa97] Handa, H., Baba, N., Katai, O., Sawaragi, T. and Horiuchi, T. (1997) "Genetic Algorithm involving Coevolution Mechanism to Search for Effective Genetic Information", Proceedings of the 4th International Conference on Evolutionary Computation,. 709-714.

[handa98] Handa, H., Katai, O., Sawaragi, T. and Baba, N. (1998) "Solving Constraint Satisfaction Problems by Using Coevolutionary Genetic Algorithms", Proceedings of the 5th Interantional Conference on Evolutionary Computation, 21-26.

[handa99] Handa, H., Watanabe, K., Katai, O., Konishi, T. and Baba, M. (1999) "Coevolutionary Genetic Algorithm for Constraint Satisfaction with a Genetic Repair Operator for Effective Schemata Formation", Proceedings of the 1999 IEEE System Man and Cybernetics Conference, Vol. III, 617-621.

[hills92] Hills, W. D. (1992) "Co-Evolving Parasites Improve Simulated Evolution as an Optimization Procedure", Artificial Life II, 313-324.

[march97] Marchiori, E. (1997), "Combining Constraint Processing and Genetic Algorithms for Constraint Satisfaction Problems", Proceedings of the Seventh International Conference of Genetic Algorithm, 330-337.

[mint94] Minton, S., Johnston, M. D., Philips, A. B. and Larid, P. (1994) "Minimizing conflicts: a heuristic repair method for constraint satisfaction and scheduling problems", Constraint-Based Reasoning, 161-206, Freuder, E. C. and Mackworth, A. K. Editors, The MIT Press.

[pare93] Paredis, J. (1993), "Genetic State-Space Search for Constraint Optimization Problems", Proceedings of the 13th International Joint Conference on Artificial Intelligence, Vol. II, 967-972.

[pare94] Paredis, J. (1994), "Co-evolutionary Constraint Satisfaction", Proceedings of Parallel Problem Solving from Nature III, 46-55.

[riff96] Riff M.-C. (1996), "Using the knowledge of the Constraint Network to design an evolutionary algorithm that solves CSP" , Proceedings of the 3rd International Conference on Evolutionary Computation, 279-284.

[riff97] Riff M.-C. (1997), "Evolutionary Search guided by the Constraint Network to solve CSP", Proceedings of the 4th International Conference on Evolutionary Computation, 337-342.

[tsang93] Tsang, E. (1993) "Foundation of Constraint Satisfaction," Academic Press.