Lecture Notes in Computer Science

Edited by G. Goos and J. Hartmanis

496

H.-P. Schwefel R. Männer (Eds.)

Parallel Problem Solving from Nature

1st Workshop, PPSN I
Dortmund, FRG, October 1–3, 1990
Proceedings

Springer-Verlag
Berlin Heidelberg New York London Paris
Tokyo Hong Kong Barcelona Budapest

Contents

Introduction 1

Genetic Algorithms 3

Genetic Algorithm Theory

A. E. Eiben, E. H. L. Aarts, K. M. Van Hee (Eindhoven, The Netherlands)
Global Convergence of Genetic Algorithms: A Markov Chain Analysis 4

D. E. Goldberg (Urbana-Champaign, IL)
The Theory of Virtual Alphabets 13

J. Hesser, R. Männer (Heidelberg, Germany)
Towards an Optimal Mutation Probability for Genetic Algorithms 23

J. Hesser, R. Männer (Heidelberg, Germany)
An Alternative Genetic Algorithm 33

K. A. De Jong (Fairfax, VA), W. M. Spears (Washington, D. C.)
An Analysis of the Interacting Roles of Population Size and Crossover in Genetic Algorithms 38

Applications of Genetic Algorithms

C. Blume (Karlsruhe, Germany)
GLEAM — A System for Simulated 'Intuitive Learning' 48

A. Colorni, M. Dorigo, V. Maniezzo (Milan, Italy)
Genetic Algorithms and Highly Constrained Problems: The Time-Table Case 55

Y. Davidor, Y. Goldberg (Rehovot, Israel)
An Evolution Standing on the Design of Redundant Manipulators 60

M. Gerrits, P. Hogeweg (Utrecht, The Netherlands)
Redundant Coding of an NP-Complete Problem Allows Effective Genetic Algorithm Search 70

M. Hulin (Munich, Germany)
Circuit Partitioning with Genetic Algorithms Using a Coding Scheme to Preserve the Structure of a Circuit 75

P. Husbands (Brighton, UK), F. Mill (Edinburgh, UK),
S. Warrington (Glasgow, UK)
Genetic Algorithms, Production Plan Optimisation, and Scheduling 80

T. Johnson, P. Husbands (Brighton, UK)
System Identification Using Genetic Algorithms 85

GENETIC ALGORITHMS AND HIGHLY CONSTRAINED PROBLEMS: THE TIME-TABLE CASE

Alberto Colorni, Marco Dorigo, Vittorio Maniezzo
Politecnico di Milano, Dipartimento di Elettronica
via Ponzio 34/5, 20133 Milano, Italy

e-mail: dorigo%ipmel1.polimi.it@iboinfn.bitnet
maniezzo%ipmel1.infn.it@iboinfn.bitnet

Introduction

Genetic Algorithms (GAs) are a stochastic computational device that allows an effective search in very large search spaces. They have been applied to various kind of optimization problems, including NP-complete problems [Garey-Johnson,1979], e.g. Travelling Salesman (TSP) [Mühlenbein,1989] and Satisfiability (SAT) [DeJong-Spears,1989], with quite good results. The main goal of our research is to understand GAs' limits and potentialities in addressing highly constrained problems, that is optimization problems where a minimal change to a feasible solution is very likely to generate an unfeasible one. As a test-problem [Colorni-Dorigo-Maniezzo, 1990], we have chosen the timetable problem (TTP), a problem that is known to be NP-hard [Even-Itai-Shamir,1976], but which has been intensively investigated, given its great practical relevance [de Werra,1985], [Davis-Ritter,1987], [Hertz-de Werra,1989].

The problem instance we faced consists in the construction of the lesson timetable for an (Italian) high school. This problem may be decomposed in the formulation of several interrelated timetables, one for each pair of sections of the school considered. A pair of sections can be in fact processed as an "atomic unit", not further decomposable given its high internal dependencies, but relatively isolated from the other pairs of sections.

Given this premises, the problem is described by:

- a list of the teachers of the pair of sections (20 in our case),
- a list of the classes involved (10 for the pair of sections),
- a list of the weekly teaching hours for each class (30 in our case),
- the *curriculum* of each class, that is the list of the frequencies of the teachers working in that class,
- some external conditions, for example the hours that the teachers use to teach in other sections.

The formal representation of the TTP may be the following:
given the 5-tuple $< H, T, A, R, f >$, where
T is a finite set $\{T_1, .. , T_i, .. , T_m\}$ of resources (teachers),

H is a finite set $\{H_1, .. , H_j, .. , H_n\}$ of time-intervals (hours),
A is a set of jobs to be accomplished (lessons to be taught),
R is a m x n matrix of $r_{ij} \in A$ (a timetable),
f is an objective function to be maximized, f: $R \Rightarrow \mathbb{R}$,
we want to compute
$\max f(\sigma, \Delta, \Omega, \Pi)$, where

- σ is the number of superimpositions, that is situations where more than one teacher is present in the same class at the same time (this number, to be minimized, is controlled by a parameter that may be different from zero for reasons of efficiency),
- Δ is the set of didactic goals (e.g., having the hours of the same subject spread over the whole week),
- Ω is a set of organizational goals (e.g., for each hour of the week having two teachers available for possible temporary teaching posts),
- Π is a set of personal goals (like, i.e., having a specific free-day).

Every solution (timetable) generated by our algorithm is feasible if it satisfies the following constraints:

- every teacher and every class must be present in the timetable in a predefined number of hours;
- there may not be more than one teacher in the same class in the same hour;
- no teacher can be in two classes in the same hour;
- there can be no "*uncovered hours*" (that is, hours when no teacher has been assigned to a class).

The algorithm

In order to apply the GA - we suppose the reader is comfortable with the model (see [Dorigo,1989] for an introduction) - we had to define an alphabet, some genetic operators and a fitness function.

Alphabet: is the set A of the jobs that the teachers have to perform; its elements include the lessons to be taught and other activities. We indicate:

- with the characters 1,2,3, .. ,0 the ten classes (i.e. $1^aA, 2^aA, ..., 4^aB, 5^aB$) where the lessons have to be taught;
- with the character D the hours at disposal for temporary teaching posts;
- with the character A the hours for the teacher updating;
- with the character S the hours during which lessons are taught in classes of sections different from the two considered.

Our alphabet is therefore A = {1,2,3,4,5,6,7,8,9,0,D,A,S}.

This alphabet allows us to represent the problem as a matrix **R** (an m x n matrix of $r_{ij} \in A$) where each row corresponds to a teacher and each column to a hour. Every element r_{ij} of the matrix **R** is a *gene*: its allelic value may vary on the subset of A specific to the teacher corresponding to the row containing the gene.

The constraints are managed as follows:

i) by *genetic operators*, so that the set of hours to be taught by each teacher, allocated in the inizialization phase, cannot be changed by the application of a genetic operator (specifically defined to meet this requirement);

ii) by a *filtering algorithm*, so that the superimpositions caused by the application of genetic operators be — totally or partially —eliminated by filtering;

iii) by the *objective function*, so that selective pressure is used to limit the number of individuals with superimpositions (superimpositions are explicitly considered in the objective function, with high penalties).

We decided to manage the superimpositions by means of both filtering (the so-called *genetic repair* [Mühlenbein,1989]) and fitness function penalties because in this way the algorithm has a greater degree of freedom in moving through the search space. This choice has been caused by the difficulty of the problem: in our application, in fact, every teacher represents a TSP-like problem (consisting in the analysis of the permutations of a predefined symbol set), which is *context sensitive* and has a search space of dimension

$$k = \frac{\left(\sum_h n_h\right)!}{\prod_h (n_h!)} = \frac{N!}{\prod_h (n_h!)}$$

where n_h is the number of repetitions of the h-th character in the row representing the teacher, and N is the total number of weekly hours (30 in our case).

It is possible to distinguish between two kinds of constraints: rows and columns. *Row constraints* are incorporated in the genetic operators and are therefore always satisfied; *column constraints* (unfeasibilities due to superimpositions and to uncovered hours) are managed by means of a combination of fitness function (f.f.) and genetic repair. Single-teacher solutions (i.e. solutions that satisfy a single teacher) are each other related through column constraints.

Fitness function: the objective function (o.f.) is the basis for the computation of the f.f., which provides the GA with feedback from the environment, feedback used to direct the population towards areas of the search space characterized by better solutions.

The o.f. is structured hierarchically and is built so as to measure a generalized cost, which represents the distance existing between the current timetable and the needs of the school. The o.f. is defined through a set of weights interactively chosen by the user.

The *hierarchical structure* has been chosen in order to acknowledge the different relevance of the several groups of problem conditions. This difference is reflected in the weights, which have different orders of magnitudes.

Genetic operators: the operators that we have defined are the following.

Reproduction. This is the classical reproduction operator that promotes individuals with an above average value of the fitness function.

Mutation of order k. This operator takes k contiguous genes and swaps them with other k contiguous non-overlapping ones belonging to the same row. Its application is subject to some restrictions: it cannot be applied when, among the genes to be mutated, there are some special characters, like A or S (these hours have in fact been allocated, during the initialization phase, to unconvertible activities).

Day mutation: it takes one day and swaps it with another one belonging to the same row. It is a special case of mutation of order k: it has been introduced for efficiency reasons (with special reference to free-day allocation).

Crossover. The objective of the definition of this operator is that of recombining efficiently building blocks (defined in the following for our case), so that, given two parents, it is possible to generate two sons having better f.f. values (or at least having one of them with a significant increase of f.f. value).

We call *local fitness function* (l.f.f.) the part of the fitness function due only to characteristics specific to each teacher. Given two individuals (timetables) of the population, R_1 and R_2, the rows of R_1 are sorted in order of decreasing l.f.f. and the best k_1 rows are taken as *building block*. Then the remaining $m-k_1$ (where m is the number of teachers) rows are taken from R_2 to generate the first son. The second son is obtained from the unutilized rows of R_1 and R_2. The value of k_1 is determined by the program on the basis of the l.f.f. of both the parents.

Filtering. The filtering operator takes as input an unfeasible solution and returns as output a possibly feasible one. It has been presented in [Colorni-Dorigo-Maniezzo,1990]. Row feasibility is always maintained, while this is not the case for column feasibility: the goal of filtering will, therefore, be that of recovering column — hence global — feasibility for any given timetable.

Some computational results

This model has been implemented using the C language on an IBM-PC with standard configuration, and it has been tested in defining the timetable for a large high school in Milan. We have co-operated with the teachers that usually define the timetable; this collaboration has been continuative, from the design phase (for requirement definition) until the validation, which is still going on with promising results.

In Fig.1 we present two evolutions of our system applied to this workbench. The first one regards a sample run, stopped after 3250 generations. It is evident how the algorithm converges to a positive value of cost. This non-zero asymptote derives from the conflicting nature of the teachers' needs. The second one concerns a run performed with very low crossover probability. It is manifest the difference of the trend and the significant worsening of the performance.

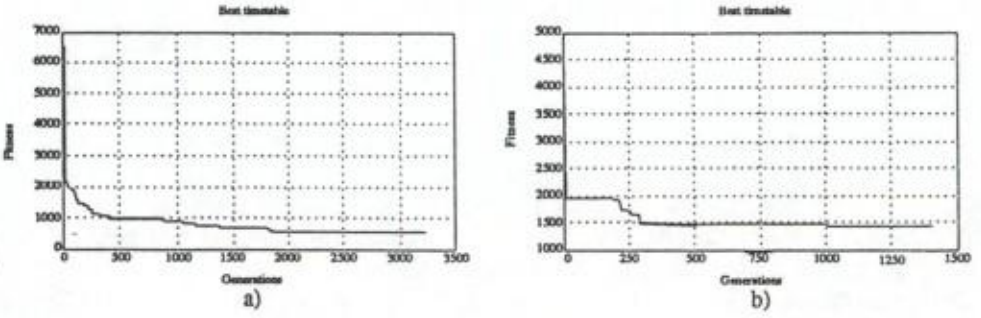

Fig.1- Two sample runs:
a) run with crossover probability set to 0.5
b) run with crossover probability set to 0.1

Conclusions and future work

In this paper we have presented a model, a class of algorithms and a computing program, regarding the solution of the timetable problem, with special reference to a real-world application (the timetable of a large high school). In so doing, we have developed a general methodology to apply GAs to highly constrained combinatorial optimization problems.

The features of our model that can be generalized are:

- the definition of genetic operators that minimize generalized cost functions with a penalty function for the possible unfeasibilities of the generated solutions;
- the distribution over genetic operators, f.f. and genetic repair of the management of the unfeasibilities;
- the hierarchical structuring of the o.f., in order to allow an easy and effective definition of the relevance of the different criteria and objectives used;
- the run-time adaptation of the f.f. weights and probabilities.

Possible future developments include:

- a theoretical assessment of the complexity of the operators utilized;
- an extensive experimentation of the parallel versions of the algorithm;
- the generalization to a broader class of constrained combinatorial optimization problems;
- the study of the convergence of GAs applied to this class of problems.

References

[Colorni-Dorigo-Maniezzo,1990] Colorni,A., Dorigo,M.,Maniezzo,V. "Genetic Algorithms: A New Approach to the Time Table Problem", NATO-ASI School on Combinatorial Optimization, Ankara, Turkey, 1990.

[Davis-Ritter,1987] Davis, L., Ritter, F. "Schedule Optimization with Probabilistic Search", Proc. 3rd IEEE Conference on Artificial Intelligence Applications, February 1987, Orlando, Florida.

[DeJong-Spears,1989] De Jong, K.A., Spears, W.M. "Using Genetic Algorithms to Solve NP-Complete Problems", Proc. 3rd Int. Conf. on Genetic Algorithms and their Application, June 1989, George Mason University.

[de Werra,1985] de Werra, D. "An Introduction to Timetabling", European Journal of Operational Research, Vol. 19, pag.151-162.

[Dorigo,1989] Dorigo, M. "Genetic Algorithms: The State of the Art and Some Research Proposal", Technical Report No. 89-058, Politecnico di Milano, Italy.

[Even-Itai-Shamir,1976] Even,S., Itai,A., Shamir,A. "On the Complexity of Timetable and Multicommodity Flow Problems", SIAM Journal of Computing, Vol.5, No.4.

[Garey-Johnson,1979] Garey, M.R., Johnson, D.S. "Computers and Intractability", W.H.Freeman & Company.

[Hertz-de Werra,1989] Hertz A., de Werra D. "Informatique et horaires scolaires", Output, Vol.12, pag.53-56.

[Mühlenbein,1989] Mühlenbein, H. "Parallel Genetic Algorithms, Population Genetics and Combinatorial Optimization", Proc. 3rd Int. Conf. on Genetic Algorithms and their Application, June 1989, George Mason University.