Solution to Vehicle Routing Problem with Genetic Algorithm

Solución al Problema de Ruteo de Vehículos Empleando Algoritmo Genético

Enrique De la Hoz Domínguez*
Karen Peña Segura**
Adel Mendoza Mendoza****

RESUMEN

El presente artículo compara dos métodos para solucionar el problema clásico de rutas de vehículos (VRP), conocido así por sus siglas en inglés (*Vehicle Routing Problem*), introducido por Dantzig y Ramser en el año de 1959, el cual consiste en minimizar el costo de repartir la mercancía desde un almacén a un conjunto de clientes, donde se utiliza un método exacto de programación lineal y una meta heurística basada en algoritmos genéticos.

El objeto de comparación será el problema de benchmark desarrollado por Christofides (1976).

En la comparación se tendrán en cuenta los mejores resultados obtenidos históricamente hasta la fecha y los obtenidos en el desarrollo de este artículo.

Palabras clave: Algoritmo genérico, VRP, Programación lineal, Heurística.

ABSTRACT

This paper compares two methods to solve the classic problem of vehicle routing (VRP), well known for its acronym in English (Vehicle Routing Problem), introduced by Dantzig and Ramser in 1959, which is to minimize the cost to distribute the goods from one warehouse to a set of clients, which uses an accurate method of linear programming and goal heuristic based on genetic algorithms.

The object of comparison is the benchmark problem developed by Christofides (1976).

In the comparison will be considered historically the best results to date and those obtained in the development of this paper.

Key words: Genetic algorithm, VRP, Linear programming, Heuristics.

** Centro de Desarrollo Tecnológico "Punto Estratégico". edelahoz@puntoestrategico.com.co*
*** Universidad del Norte. ksegura@uninorte.edu.co*
**** Universidad del Atlántico. adelmendoza@mail.uniatlantico.edu.co*

Fecha de recepción: 31 de julio de 2013 • Fecha de aceptación: 2 de septiembre de 2013

1. INTRODUCTION

The Vehicle Routing Problem (VRP), which has been introduced by Dantzig and Ramser in 1959 [1], is based on minimize the cost of distributing the goods from a store to a set of clients. The Vehicle Routing Problem (VRP) can be understood as the relationship or intersection of two combinatorial optimization problems well known. The first, the Traveling Salesman Problem (TSP), which considers the capacity of each car as infinite, and the Bin Packing Problem (BPP) [2].

The interest of this problem comes from two main causes. For a case, the VRP is a problem of high academic interest because of its difficulty (is a NP-hard problem, which tells us that the resolution of this issue exhaustively search all possible combinations since the best solution is not possible to be achieved in a reasonable time useful for the problem) [3], which includes restrictions, and the multitude of existing variants. On the other side, it has a direct application to reality, being applicable to large logistics companies and distribution companies (newspapers, food) [4].

The “GA is a stochastic global search method that mimics the metaphor of natural biological evolution. GA operates on a population of potential solutions applying the principle of survival of the fittest to produce (hopefully) better and better approximations to a solution” [5].

The general idea of this paper is based on finding solutions to the classical VRP problem instances E-n33-k4, B-n41-k6, F-n45-k4, which have been developed by Christofides [6], as the problem of Benchmarking for instance.

Given the above, it is clear to note that genetic algorithms are helpful in optimizing the VRP problems for this specific case, since they provide optimal solutions closest to that offered by simple heuristics. However, this will be the case study.

These solutions must be found using linear programming and the development of a genetic algorithm with mutation and selection by Ranking.

With this paper researchers pretend to find out which of the solutions is most appropriate for the problem, which minimizes the distance traveled by vehicles, where the percentage of improvement is greater and the execution times are minimized.

2. LITERARY REVIEW

Before embarking on the implementation of any article or scientific research, it is essential to better understand the contributions that have been previously on the subject, either guided or not to make mistakes. The step of the review of literature or documentation is critical to the development of science,

as it will benefit from the works of other authors; it will serve as a foundation for future research experiences, as in this case.

Within the environment of the VRP problem there is the emergence of combinatorial optimization and complexity.

The study combines optimization modeling and algorithmic solution of problems which seeks to maximize or minimize a function of several variables defined on a discrete set.

2.1. VRP (Vehicle Routing Problem)

The Vehicle Routing Problem (VRP), which has been introduced by Dantzig and Ramser in 1959 [1], is based on minimize the cost of distributing the goods from a store a set of clients (see Fig. 1).

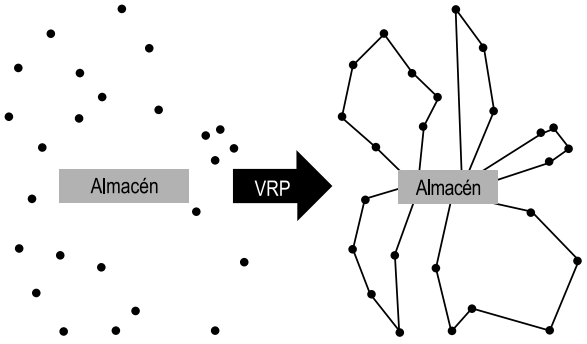

Figure 1. The Vehicle Routing Problem (VRP)
Source: Dorronsoro et al. [4]

In the case involves minimizing the cost to reduce the number of vehicles used for such distribution and the length of the routes to follow, so that the vehicle go more than once per customer.

The routing problem, is located in a wide range of variants: [7]

- CVRP (Capacitated VRP) (Ralphs, Hartman y Galati, 2001)
- MDVRP (Multi-depot VRP) (Hjorring, 1995)
- PVRP (Periodic VRP) (Baptista, Oliveira y Zuquete, 2002)
- SDVRP (Split Delivery VRP) (Dror, Laporte y Trudeau, 1994; Archetti, Mansini y Speranza, 2001)
- SVRP (Stochastic VRP) (Laporte y Loveaux, 1998)
- VRPB (VRP with Backhauls) (Ralphs, Hartman y Galati, 2001)
- VRPPD (VRP with Pick-up and Delivering) (Righini, 2000)
- VRPSF (VRP with Satellite Facilities) (Bard *et al.*, 1997)
- VRPTW (VRP with Time Windows) (Cordeau *et al.*, 2002)

The area of application of such models is found in industry, logistics and science, engineering and bu-

siness administration, routing and vehicle loading in distribution networks, telecommunications network design, production planning, crew assignment lines airline planning, the generation of the electricity [7-8].

In the recent history of VRP exists a constant evolution in the quality of the methodologies used to solve the problem. These techniques include both exact algorithms and heuristic methods. Anyway, because of the difficulty of the problem, there is no accurate method capable of solving instances of more than 50 clients [9].

2.2. Methods of solution

Currently, seeking methods for solving optimization problems is increasingly important and attention has focused more on the use of combinatorial optimization methods due to their complexity in obtaining solutions. These techniques are classified as conventional local optimization techniques (heuristics) and intelligent local optimization techniques (metaheuristics).

Heuristic is derived from Greek Eureka, which was the term used by Archimedes when he discovered his principle of density, so that its meaning is found term oriented, "I did!". Thus "a heuristic technique is a method that looks good solutions close to optimal, at a reasonable computational cost cannot guarantee optimality" [7].

Heuristic techniques for the routing problem described in this document can be classified into four categories:

- Construction and the method of Clarke and Wright Savings [10], whose bases are the savings generated by inserting new customers in each vehicle to complete a final solution.
- Methods of grouping first, then routing, grouping customers in several subsets, each subset assigned to a vehicle and then solve each corresponding TSP.
- Heuristic methods for routing first, then group are those who begin to solve the TSP have defined customers and then start the route found to assign a section to each vehicle.
- The methods of improvement, such as Or-Opt exchanges. Metaheuristics have emerged over the past two decades and is based on taking a feasible solution and find its optimal by using improved heuristics embedded in a larger framework [7].

Best known metaheuristics techniques used to solve the Vehicle Routing Problem (VRP) are the ant colony, scatter search, genetic algorithms and tabu search in this paper we analyze the results obtained from the solution to the VRP by applying different types of metaheuristics.

There is also the algorithm proposed by Christofides, Mingozzi and Toth [11], which operates in two phases. It is determined in the first phase the number of routes to be used, along with a client to initialize each of the routes. Then in the second phase, these routes are created and inserted the other customers there.

In the first phase of the algorithm it applies a sequential insertion algorithm for compact route. No special attention is given to the location of customers within each route, as in this stage only initial clients are kept of each route and the number of routes to the final solution.

To start the *K-th* route, a V_k costumer is selected, who is part of the non-visited ones. We define the cost of inserting the customer *w* in the route who has a V_k as δw, $V_k = C_{0w} + \lambda_{Cw}$, V_k (if the client can not be inserted, the function takes the value of ∞) and customers are assigned to the route starting with the lowest values *δ* since there is no feasible insertions, in which case a new route is created or the algorithm is terminated.

2.2.1. Christofides, Mingozzi y Toth algorithm. Phase 1

Step 1. (new route). Do k : = 1.
Step 2. (first costumer). Select a costumer not visited V_k to insert in the route. For each unvisited customer w, calculate δw, V_k.
Step 3. (insertion). Calculate w * = arg minw δw, V_k about the unvisited customers w. Insert w in the route and apply the 3-opt algorithm. If it stays unvisited, it can be inserted in the route, go to 3.
Step 4. (next route). If all the costumers stay in some route, end. If they don't do k := k + 1 and go to step 2.

In the phase two of the algorithm, it makes k routes and it starts with the selected costumers in the step 2 of the phase 1.

For each unvisited customers associated with the route, the insertion cost have to be minimized. After that, they select some route in which each customer are inserted.

To decide the customer's order they are inserted to a route, it must be calculated to each customer the diference between the cost of making the insertion in that route and the second best choice for him.

2.2.2. Christofides, Mingozzi y Toth algorithm. Phase 2

Step 5. (initialization). Make k routes $r_t = (0, v_t, 0)$ para t = 1, . . . , k, being k the number of routes obtained in the phase 1. Be J = $\{r_1, \ldots, r_k\}$.
Step 6. (association). For each w customer that has not been visited yet.
Calculate t_w = arg mint|rtϵJ δw, v_t
Step 7. (emergency). Select $r_t \epsilon$ J and do J := J \ $\{r_t\}$. For each customer
w so that tw = t, calculate T'_w = arg min t|$r_t \epsilon$ Jδw, v_t y $T_w = T'_w - T_w$.
Step 8. (insertion). Calculate w* = arg max w|T_w=t T_w. Insert w* in the route r_t and aplicate the 3-opt algorithm. If there are associated customers to r_t that have been inserted, go to 8.
Step 9. (ending). If J =6 φ, go to 6. If all the customers have been visited, end. If not, apply the algorithm again (including the phase 1) on the unvisited customers.

2.3. Evolutionary Algorithm Multi-objective

Multiobjective Evolutionary Algorithms (MOEAS) combine evolutionary computation techniques to multiobjective optimization theory, and therefore offer the possibility of unlimited searching and complex spaces.

Also, they can keep an entire population of optimal solutions, so that they have been raised as a good tool to solve various algorithmic problems of multi-objective optimization. Some of these are known evolutionary algorithms NSGA, SPEA, NSGA-II, SPEA II and PAES-II, among others.

A Multi-objective Optimization Problem (MOP) is one that includes a set of n decision variables, a set of k objective functions, and a set of m inequality constraints and p equality constraints, where the objective functions and constraints are functions of the n decision variables.

In its general form, a MOP can be expressed mathematically as:

> "x" represents the decision vector, "X" denotes the decision space, and is the target vector and the target space is denoted by Y. The set of decisions x that satisfy the m + p constraints are called feasible solutions set is denoted by Xf.
>
> Each decision vector is of the form x = (x1, x2, ..., Xn) and its corresponding objective vector is of the form y = (f1 (x), f2 (x), ..., Fn (x)) . Therefore, the problem is to find the x that provides the best value for f (x). multiobjective optimization, however the best solution is usually not the only one, but there is a set of best solutions known as non-dominated solutions [13].

2.4. SPEA II Algorithm

Powered by Zitzler, Laumanns and Thiele in 2001 [14], in order to overcome weaknesses identified in the allocation scheme of the SPEA adaptation.

In this algorithm, the fitness mapping function is improved taking into account for each individual the number of individuals it dominates and the number of individuals for which it is mastered.

This scheme also adds an estimate of population density. NE max-size-external population EP (used to elitism) is fixed, unlike the SPEA, in which the size of PE is variable but limited. PE is made only for non-dominated individuals as long as the number of these is greater than or equal to NE max.

The technique of grouping (clustering), responsible for maintaining diversity in the population of SPEA, is replaced by a truncation method, which prevents solutions eliminate extreme non-dominated set of solutions [14].

The selection is done by binary tournament, taking as a criterion of comparison the fitness of each individual. SPEA II assumes minimization of fitness, so that individual wins the tournament with a lower fitness value.

Pareto Evolutionary Algorithm Robust called in English or Strength Pareto Evolutionary Algorithm (SPEA) was proposed by Zitzler and Thiele in 2001.

3. VRP AND GA

This part describes the process to generate a solution for VRP problem using a genetic algorithm, first the detailed description of genetic algorithm applied to VRP are going to be shown: how do it solve the VRP in theory. After that, the program is going to be described. At the end thes results are going to be shown, and compared to Linear programming results for VRP and Best Known solutions from specialized literature.

3.1. Describing the specified VRP problem

The VRP to be solved is going to be described. This problem is a specified VRP, the Capacitated Vehicle Routing Problem. The first customer is always the depot where the trucks are starting from. Each truck has got a specified (and the same) capacity.

3.1.1. The customers

Each customer has got a number, which is not so important. They have got a specified position (X and Y coordinates), so the distance could be calculated. These coordinates are given in simple integers: the unit is ‘1’. All customers have got a demand, which couldn’t be more than a truck can maximal carry. The demand is also specified with integers, where the unit is also ‘1’.

3.1.2. Other restrictions

Each car is leaving from the depot, and going there back, if it has got not enough packages to deliver, or there is no customer. The capacity of the trucks and the number of trucks are also defined in the dataset. The measurement of the packages and the truck capacity is also one unit, represented on integers: ‘1’. Each package has the same size and dimension, exactly occupying a space ‘1’.

3.1.3. The aim

The goal of this project is to implement a code, which solves the problem as good as possible, under the given circumstances.

Example Dataset

The problem instances belong to different groups of known problems. Problems E-n33-k4, E-n76-k7 and E-n101-k8 belong to the Christofides and Eilon problems. The given properties are formatted in the following way.

E-n33-k4
VEHICLE

NUMBER	CAPACITY		
4	8000		
CUSTOMER			
CUST NO.	X	Y	DEMAND
0	92	495	0
1	298	427	700
2	309	445	400
…			
33	319	433	1100

3.2. Representation of the problem

The solution uses indexed array to store the chromosomes and each chromosome contains a set of customers, to be visited by an associated vehicle (The associated vehicle is the number of the vehicle which will visit these cities from the vehicle set).

The depot is not included in these chromosomes. The fitness evaluated then for each chromosome, what is a very easy function in this case: the sum of the totally needed distance. To make further calculations easier the fitness of the chromosome and the associated vehicle number is also stored in the chromosome. So a chromosome looks like this:
Customer#1, Customer#2, … , Customer#N, FITNESS, ASSOCIATED VEHICLE.

3.2.1. Initial population

This is a repeatable step of the program, until it does not reach a maximum value of failed initialization, which is defined in the user interface. It chooses a random, nonrepeating permutation of the customers, which of course might not be valid. Then a new chromosome has to be generated until the loop does not reach the allowed tries to generate chromosomes. Once a chromosome is valid (validity is checked after each generated chromosome), it is returned with its calculated fitness. It is also allowed to return invalid chromosomes if it is enabled on the User Interface. These chromosomes might fail later in the selection, but enables us to run the program faster.

3.2.2. Genetic cycle

The used structure of the program is very simple. First, the program generates the initial population and then it goes to a loop: this loop ends, when the maximum number of the generated population (value defined in the User Interface) is reached. Within this loop, all genetic methods (described later) are executed:

- Selection of survivals (value defined in User Interface),
- Creating children with crossover (value defined in User Interface),

- Applying mutation on previously crossover children (value defined in the User Interface),
- Adding children to the population.
- Fitness calculation (The fitness is the total length of the chromosome, so as it becomes larger, the solution becomes worse. One chromosome is "visited" by one vehicle).

3.2.3. Survival selection

The survivals are selected by the biased roulette wheel algorithm. The percentage of the survivals can be inputted from the User Interface. The selection ends, when the given number of individuals is reached.

3.2.4. Validation

The program loops with all vehicles and for each vehicle it tries to satisfy all constrains on the chromosome elements (customers) it is associated with. The loop finishes after a given number of attempts are reached: if the vehicle can not visit one of the "next customers", the code tries to find another vehicle for that customer. If it was unsuccessful (there are no more vehicles able to reach that customer), the code shows an unsuccessful result for the chromosome. If the validation was successful, it is marked as valid and the fitness is calculated. All vehicles are leaving from the depot.

The vehicle travels to the next customer in the chromosome (the first time it is the first customer from the depot, then second customer, etc.) and its metrics are being modified:

- The vehicles' travel time is increased by the distance.
  If the vehicle fails at the position, the code tries to find another vehicle for the city, until there are not vehicles left. Each vehicle also has a number (defined on the User Interface), which shows how many times a vehicle should fail, satisfying the constraints for a given city.

A chromosome is accepted as valid if, and only if:

- The current load of the vehicle is at least as many as the demand of the given customer, and this is true for each customer.
- All vehicles must return to the depot. The fitness is given by the total amount of time each vehicle was used.

3.2.5. Crossover

It is tough to create new individuals with crossover, which are not invalid, so do not duplicate customers appearance. How and when can we correct the errors occurring within the population? This is especially true in the case when we use GA for some kind of track or route optimization, when the genes are not separated, but representing a tour for cities or sites to visit, each one once:

Parent 1 Genes	1	9	2	8	6	7	3	4
Parent 2 Genes	5	4	3	7	6	8	2	1
Children 1 Genes	1	9	2	8	6	8	2	1
Children 2 Genes	5	4	3	7	6	7	3	4

Figure 2. Error in crossovers

Source: By authors

To solve this inconvenient we can make a little mutation, where these invalid genes are mutated with limitation to another one which is nearby and that is removed from the other part of the list (or swapped). This is quite easy, the mutation is not too much and not time consuming.

The child inherits the value, which is less. After the inheritance, the gene which the children inherited have to be found and replaced by the non inherited one in the parent from the children does not inherited. It then avoids us from having duplicates, because it is not going to be selected later again. Crossovers are applied on a given number of parents, defined in the User Interface.

3.2.6. Mutation

Mutation selects a random parent (even crossover parent and non-crossover parent), and swaps one of its genes, and recomposes and revalidates the fitness value of it. This might be then a new child.

3.2.7. Test of Genetic Algorithm

Tests have used the same settings, so they can be easily compared to each other. All tests were executed with:

100 populations,
- 100 chromosomes in each population,
- 30 % of survivors, where the best 5 individuals always survived,
- 35 % of both crossover and mutation.

4. THE BENCHMARK PROBLEMS

The test is performed on 3 small problems, which are listed in the tables below. It would have been desirable to perform the testing using more large problems but large problems with known optimal or best values were difficult to find and execute in Linear Programming solver software.

The problem instances belong to the Christofides and Eilon problems. Problems E-n33-k4, E-n76-k7 and E-n101-k8 are used for testing.

5. RESULTS

In this section the results are presented. The first subsection involves the testing for the linear programming procedure to solve problems. The next sections illustrate the results for the genetic algorithm proposed in this paper. In the final subsection, the results of the comparison with proposed genetic algorithm, the Linear Programming procedure and best known results from specialized literature are shown.

5.1. Linear Programming results

The problems were solved using the "VRP SOLVER" software developed by the Department of Industrial Engineering of Lehigh University.

Problem	Results
E-n33-k4	1020
E-n76-k7	890
E-n101-k8	1120

Figure 3. Linear Programming results

Source: By authors

5.2. Genetic Algorithm results

The genetic algorithm results obtained for the proposed problem were developed in a Matlab code.

Problem	Results
E-n33-k4	945
E-n76-k7	760
E-n101-k8	998

Figure 4. Genetic Algorithm results

Source: By authors

5.3. Results Comparison

Genetic Algorithm developed in this paper differs from the best known value in the specialized literature in an average of 12 % for the three proposed problems.

Table 1. GA vs Best Known Values

Problem	GA	Best Known	Diff from opt (Best Known)
E-n33-k4	945	835	13 %
E-n76-k7	760	682	11 %
E-n101-k8	998	817	22 %

Source: By authors

Linear Programming results differs from the best known value in the specialized literature in an average of 30 % for the three proposed problems.

Table 2. LP vs Best Known Values

Problem	LP	Best Known	Diff from opt (Best Known)
E-n33-k4	1020	835	22 %
E-n76-k7	890	682	30 %
E-n101-k8	1120	817	37 %

Source: By authors

Comparing results from Linear Programming procedure and Genetic Algorithm is clear how GA overcome the results obtained by LP.

Table 3. Ga vs Lp Results

Problem	LP	GA	Diff from opt (Best Known)
E-n33-k4	945	1020	-7 %
E-n76-k7	760	890	-15 %
E-n101-k8	998	1120	-11 %

Source: By authors

6. CONCLUSIONS

In this work, two rather effective methods were described to solve the capacitated vehicle routing problem. It seems like these methods can be effectively used for these kinds of problems.

Linear Programming is effective when is used in small size problems, but is not possible to obtain quick and quality results for large problems like Transportation Companies face in real life.

Genetic Algorithm proposed in this paper is capable of provide results that differ from the best known value in 12 % in average for the three instances tested, this percentage indicates that the algorithm could be improved by streamlining mutation process or applying some greedy techniques as Djisktras and Bellman-Ford's algorithm to select next customer visit. The GA could be used for both small an large space of solution due to the ability for generate random offspring in every single run.

ACKNOWLEDGEMENTS

The authors are grateful to Miguel Rojas Santiago for his contribution to provide knowledge bases for its development. Also, to God for allowing the realization of it in the best way. This paper is based on the legacy provided by Petrica C. Pop, Imdat Kara, Andrei Horvat Marc, in their paper "New mathematical models of the generalized vehicle routing problem", it was a guide to us for make a new solution to the problem, to supporting in Genetic Algorithm.

7. REFERENCES

[1] G. B. Dantzig y J. H. Ramser, "The Truck Dispatching Problem", *Management Science*, vol. 6, n. 1, pp. 80-91, October 1959.

[2] S. Martello y P. Toth, *Bin-packing problem. Knapsack Problems: Algorithms and Computer Implementations*. Wiley, capítulo 8, 1990, pp. 221-245.

[3] M. Garey and D. Johnson, *Computers and Intractability: A guide to the theory of NP-Completeness*. New York, NY, USA: W. H. Freeman & Co., 1979.

[4] B. Dorronsoro, A. J. Nebro, D. Arias y E. Alba, *Un algoritmo genético híbrido paralelo para instancias complejas del problema VRP*. s.f.

[5] A. Chipperfield, P. Fleming, H. Pohleim y C. Fonseca, "Genetic Algorithm Toolbox User's Guide". *ACSE Research Report* No. 512, University of Sheffield, 1994.

[6] N. Christofides, Worst-case analysis of a new heuristic for the travelling salesman problem, *Report 388*, Graduate School of Industrial Administration, CMU, 1976.

[7] J. M. Daza, J. R. Montoya, F. Narducci, Resolución del problema de enrutamiento de vehículos con limitaciones de capacidad utilizando un procedimiento metaheurístico de dos fases, 2009.

[8] O. Díaz Parra, M. A. Cruz Chavez, El problema del transporte. Centro de Investigación en Ingeniería y Ciencias Aplicadas, Cuernavaca. Morelos. /Papadimitriou, C.H., Steiglitz, K. *Combinatorial optimization, algorithms and complexity*. Mineola, New York, USA: Dover Publications, Inc., 1998.

[9] P. Toth and D. Vigo, The Vehicle Routing Problem, Monographs on Discrete Mathematics and Applications. Philadelphia: SIAM, 2001.

[10] G. Clarke and J. Wright, "Scheduling of vehicles from a central depot to a number of delivery points", *Operations Research*, 12 n. 4, 568-581, 1964.

[11] N. Christofides, A. Mingozzi, P. Toth, The Vehicle Routing Problem. In: *Combinatorial Optimization*. Wiley, Chichester, 1979, 315-338.

[12] A. Olivera, *Heurísticas para problemas de Ruteo de Vehículos*. Montevideo, Uruguay, 2004.

[13] A. Diaz, J. De Vasconcelos, "Multiobjective genetic Algorithms Applied to solve optimization problems" IEE, *Transaction of magnetic*, vol. 38, n. 2, March 2002.

[14] E. Zitzler, M. Laumanns and L. Thiele. "SPEA II: Improving the Strength Pareto Evolutionary Algorithm". *TIK-Report 103*. May 2001.