Journal of the Applied Mathematics, Statistics and Informatics (JAMSI), **1** (2005), No. 1

Steiner Problem Solved with Genetic Algorithm

Dominik Ďurikovič

Abstract

This article introduces into the genetic algorithms(GA) and it's applications on the Steiner optimization problem. It is possible to create cost optimal plans for power and gas distribution with the Steiner problem in graphs. The article shows few new modified methods to solve Steiner problem with genetic algorithms. The modifications are based on the better start point of the GA algorithm, operators modifications and other. These methods are compared with known methods Takahashi Matsuyama algorithm and Kou, Markowsky and Berman heuristic.

Mathematics Subject Classification 2000: 68R10, 05C05, 05C35, 90C15
Additional Key Words and Phrases: genetic algorithms, crossover, mutation, Steiner problem, graph theory, optimization

1. GENETIC ALGORITHMS (GA) INTRODUCTION

Described genetic algorithms with few more comments can be found in the [6], [2]. The genetic algorithm (GA) is a search procedure that optimizes some objective function f by maintaining a population P of candidate solutions and employing operations. Operations are inspired by genetics (called crossover and mutation) to generate a new population from the previous one. The candidate solutions are encoded as a bit strings usually.

One point Crossover:. provides a method to combine two candidates from the population to form two new candidates. For example, take first k bits from two n bit strings A and B and switch them together to form two new candidates.

More point Crossover:. We expect two bit strings
B=$(b_1, b_2, \ldots, b_{pos}, b_{pos+1}, \ldots, b_k)$ and C=$(c_1, c_2, \ldots, c_{pos}, c_{pos+1}, \ldots, c_k)$
each with k bits length.
—Lets do the m-point crossover ($m < k$, m is an integer constant).
—Generate m random integer points x_i from interval $\langle 1, k-1 \rangle$. We will sort that points like :

$$1 \leq x_1 < x_2 < \ldots < x_{m-1} < x_m \leq k-1$$

—New pair of individuals after m -point crossover:
$(b_1, b_2, \ldots, b_{x_1}, c_{x_1+1}, \ldots, c_{x_2}, b_{x_2+1}, \ldots, \ldots, b_{x_m}, c_{x_m+1}, \ldots, c_k)$ and
$(c_1, c_2, \ldots, c_{x_1}, b_{x_1+1}, \ldots, b_{x_2}, c_{x_2+1}, \ldots, \ldots, c_{x_m}, b_{x_m+1}, \ldots, b_k)$

Uniform crossover:. This cross over is the same as other crossover but after crossover two worst candidates from parents and offsprings are deleted. We select two different parents for crossover .

Mutation:. provides a method to modify one candidate in the population. The random flipping of one bit in a candidate is one possible mutation.

Selection:. In this article we use selection of the new population called roulette selection.
—Evaluate *fitness* of each candidate $\mathbf{v_i}$ and note as $eval(\mathbf{v_i})$,

$$i = 1, \ldots, pop\_size$$

pop_size - size of the population
—Evaluate *fitness of the population* as $F = \sum_{i=1}^{pop\_size} eval(\mathbf{v_i})$
—Calculate candidate's selection probability p_i for each candidate $\mathbf{v_i}$

$$p_i = eval(\mathbf{v_i})/\mathbf{F}$$

—Calculate group probability q_i for each candidate $\mathbf{v_i}$

$$q_i = \sum_{j=1}^{i} p_j$$

—Choose random number r from the interval $\langle 0, 1\rangle$
—We select first canditate in to the new population if $r < q_1$, otherwise candidate $\mathbf{v_i}$ is selected in to the population if $q_{i-1} < r \leq q_i$.
The two last points repeat while new population is not filled.

The most important part of the GA is an evaluation of the fitness for each candidate. The fitness of candidate represents a probability to stay in a new population for candidate.

Genetic algorithms allows parents with good fitness to stay in next population usually.

2. THE STEINER PROBLEM IN GRAPHS

The NP hard Steiner problem in graphs (SPG) is to find connected subgraph for a given vertices from G with minimal cost.

In the SPG is given graph $G = (V, E)$ and subset $V' \subseteq V$ (V- set of vertices, E- set of edges).

Graph $G' = (V', E')$ is a connected subgraph of the graph G induced with V' , where

$$[v_i, v_j \in V' \wedge (v_i, v_j) \in E] \Rightarrow (v_i, v_j) \in E'$$

Complete graph is a graph with connected vertices to each other. Complete graph with cost of edge e_{vw} equals to the shortest path in the graph G ($sp(v, w)$) is called *range graph of graph G. Cost function* $c : E \to \mathbb{R}$ assigns real number to each edge in a graph. The sum of all edge costs from G is *the graph cost* ($c(G)$).

Steiner problem in graphs (SPG): We have a connected undirected graph $G = (V, E)$, a positive cost function $c : E \to \mathbb{R}_+$ and a subset $W \subseteq V$ (vertices from W are called *terminals*). The goal is to find connected subgraph $G' = (V', E')$, where $W \subseteq V'$ and cost $c(G')$ is minimal. Connected and acyclic subgraph G' of the graph G where $W \subseteq V'$ is called *steiner tree for W in graph G.* The result G'

with minimal cost is called *minimal steiner tree (MStT) for* W *in* G. We define *a steiner vertices* from G'as set $S \subseteq V \setminus W$, where $V' = W \bigcup S$.

The following notations will be in use here : $n = |V|$, $p = |W|$ a $r = n - p$.

3. GA ALGORITHM FOR THE SPG

Author of the following algorithm is Esbensen [3] and this article shows some modifications to get better results. Here we have individuals with best fitness (they will stay in the next population) and individuals with worst fitness (they will die) as it is usual in GA.

Algorithm 1: GAMStT
/*INPUT - graph and set of the terminals from the MStT definition */
/*OUTPUT - found optimal Steiner tree with minimal cost */
graphReduction();
RandomlyGenerate(Φ_c); Evaluate(Φ_c); /* fitness from section [1] */
λ=BestOf(Φ_c);
repeat until stopCriteria();
... $\Phi_n = \emptyset$;
... **repeat** $N/2$ **times:**
..... choose $\phi_1 \in \Phi_c$, $\phi_2 \in \Phi_c$; /*Select parents*/
..... $\{\psi_1, \psi_2\}$=Crossover(ϕ_1, ϕ_2); /*Uniform crossover [section 1]*/
..... $\Phi_n = \Phi_n \bigcup \{\psi_1, \psi_2\}$;
... **end;**
... Evaluate($\Phi_c \bigcup \Phi_n$); /* fitness from section [1] */
... Φ_c =reduce($\Phi_c \bigcup \Phi_n$);
... $\forall \phi \in \Phi_c$: mutateWithProbability(ϕ); /*Mutation*/
... $\forall \phi \in \Phi_c$: invertWithProbability(ϕ); /*Reordering of a genes*/
... Evaluate(Φ_c); /* fitness from section [1] */
... λ=BestOf($\Phi_c \bigcup \{\lambda\}$);
end;
optimize(λ); /*heuristic algorithm*/
output(λ); /*output the best one*/

Algorithm 1 works with population of an individuals as specific near optimal results of the MStT problem. The Crossover and mutation genetic operators are used in this algorithm. Individuals with highest fitness responds to the good optimal result of MStT problem, after few generations. Whole graph is reduced with the procedure *graphReduction()* described in [3] or [2]. The vertices and edges are reduced after *graphReduction()*. This reduction is usefull to make smaller CPU time consumption. Procedure *optimize()* runs *KMB heuristic algortihm* [4] on the best individual after the GA simulation.

4. GENOTYPE CODING IN GA

The main kernel for our GA is a well coded genotype. Coding represented here is based on the Kou, Markowsky and Berman heuristic [4]. The timecomplexity of this algorithm is $O(pn^2)$.

Genotype is a set of the steiner vertices paired with 1 or 0 in our problem. While steiner vertex is paired with 0 then this vertex is not in the result graph represented by individual otherwise it is. Decoder (KMB heuristic) calculates the fitness from genotype (cost of the MStT represented by the individual). The KMB heuristic as a decoder is called with input set W union with steiner vertices from current genotype.

Predict numbered vertices $V \setminus W$ Steiner problem and $\pi : \{0, 1, \dots r-1\} \to \{0, 1, \dots r-1\}$ is bijective function for current input . Genotype should be :

$$\left\{ \left(\pi(0), i_{\pi(0)}\right), \left(\pi(1), i_{\pi(1)}\right), \dots, \left(\pi(r-1), i_{\pi(r-1)}\right) \right\}$$

, where $i_k \in \{0, 1\}$, $k = 0, 1, \dots r-1$. Steiner vertices with the genotype are represented as $S = \{v_k \in V; i_k = 1\}$. The KMB heuristic was choosed due to that it returns right Steiner tree for any S set .
The following lines shows crossover example of the Steiner candidates.
We have parents

$$\alpha : \{(2,1), (0,1), (1,0), (4,0), (3,0)\}$$

$$\beta : \{(1,0), (2,1), (4,1), (3,1), (0,0)\}$$

Step 1: reorganisation of the β:

$$\gamma : \{(2,1), (0,0), (1,0), (4,1), (3,1)\}$$

Step 2: Crossover in point $x = 2$:

$$\phi : \{(2,1), (0,1), (1,0), (4,1), (3,1)\}$$

$$\psi : \{(2,1), (0,0), (1,0), (4,0), (3,0)\}$$

5. GA FOR MSTT PROBLEM MODIFICATIONS

The global timecomplexity of the 1 algorithm is $O(n^3 + Npn^2 + N \log N + (n-p)pn^2)$. Where N is the given number of the individuals in the population (*population size*).

As we can see procedure *Optimize()* in the algorithm 1 has high time consumption and its timecomplexity is $O((n-p)pn^2)$. We will replace KMB heuristic in procedure *Optimize()* with TM algorithm [9] (Takahashi Matsuyama algorithm). The new timecomplexity is $O(min\{p-2, n-p\}(pn^2 + |E| \log |E|))$ (number of the *Steiner vertices* multiplied by TM algorithm complexity) . Assume that $p - 2 \leq n - p$ (our set of test problems [1] meet this condition) then new *Optimize()* procedure has timecomplexity $O(p(pn^2 + |E| \log |E|))$.

—If $p = n$ then a new optimize procedure is worse than an old

—If $p \leq konst.$, or $p = \sqrt{n}$ then timecomplexity is better for the new optimize procedure

group/runs	opt.	precision$\leq 1\%$	$\leq$IKMB	$\leq$ITM	time $\leq$IKMB	time $\leq$ITM
b/900	94.3%	98%	93%	92.2%	13%	13%
c/1000	35.8%	64.5%	55.5%	45%	38.5%	4.5%
d/1000	35.5%	63%	37%	56.5%	58.5%	0%
e/160	35%	60%	65%	45%	45%	5%

Table I. Comparing Esbensen's GA with IKMB and ITM for B, C, D, E groups.

This modification globally speed up GA but the result is not with better accurate. The next modifications tries to get results with better accurate:

Procedure *RandomGenerate()* randomly generate zero population. We generate $3/5$ of individuals with TM heuristic and others are generated randomly in the zero population. This has timecomplexity $O(N(pn^2 + |E|\log|E|))$ (where N is a size of the population).

We have changed one point crossover genetic operator with three point uniform crossover. Two new individuals are created in the uniform crossover. We choose two best fitness individuals from this individuals and its parents as two new individuals.

6. EXPERIMENTAL RESULTS

This part describe comparision of the GA from Esbensen with our GA modification and with well known algorithms IKMB, ITM (iterated version of KMB and TM).

The large connected graph test problems for MStT are from [1]. They are divided into the four groups: B, C, D and E. These graphs were randomly generated with integer positive cost edges under 10.

The B graphs: $n = 100, p = 50$ with 200 edges.

The C graphs: $n = 500, p = 250$ with 12500 edges.

The D graphs: $n = 1000, p = 500$ with 25000 edges.

More complex are graphs in the E: $n = 2500, p = 1250$ with 62500 edges.

Used parameters for GA: population size $N = 40$, mutation probability $p_{mut} = 0.005$, inversion probability $p_{inv} = 0.1$. Second column in the tables describe optimum achievement in the percentage. Third column represents number of the results with the condition $100 * (1 - \frac{optimalOptimum}{foundOptimum}) \leq 1$. Number of better results than IKMB or ITM algorithms results are shown in two next columns. The last two columns represents number of the results found in shorter time than with IKMB or ITM.

These tests were run on a server with AMD 800MHz, 384MB RAM with system Mandrake Linux. The GA was running 50 times on each test problem from B, C, D group test problems. Due to the time consumption test problems from the E group were solved with GA eight times only. Exact results of the test problems were calculated by Beasly with algorithm „branch and cut“ on supercomputer Cray X-MP/48 [1].

Total GA for Steiner tree problem in graphs was runned on 3060 problems.

7. CONCLUSIONS

The tables I and II shows that our modification has better precision in the B, C and E groups than Esbensen's GA algorithm. The modification has better time-

group/runs	opt.	precision$\leq$ 1%	$\leq$IKMB	$\leq$ITM	time $\leq$IKMB	time $\leq$ITM
b/900	99.3%	99.4%	99.3%	99.8%	10.6%	10.6%
c/1000	56.9%	86.9%	74.7%	93.6%	82.5%	0%
d/1000	35.1%	72.5%	45.6%	99.3%	93.6%	6.9%
e/160	25.4%	62.1%	54.2%	100%	100%	35%

Table II. Comparing our modification GA with IKMB and ITM for B, C, D, E groups.

complexity on some examples as we can see. Due to the stochasticity the results may be little other from time to time but here published complexity supports these results generally.

REFERENCES

[1] J. E. Beasley, OR-Library: *Distributing test problems by electronic mail*, J. Oper. Res. Soc. 41 (1990).

[2] D. Ďurikovič: *Genetic's algorithms and Steiner problem (in Slovak)*. FMFI UK, Bratislava 2003 (rigid work).

[3] H. Esbensen: *Computing near-optimal solutions to the Steiner problem in a graph using a genetic Algorithm*, Networks 26(1995), 173-185.

[4] L. Kou, G. Markowsky and L.Berman: *A fast algorithm for Steiner trees*, Acta Informatica 15(1981), 141-145.

[5] E. L. Lawler, *Combinatorial Optimization: Networks and Matroids*. Holt, Rinehart and Wiston, NY (1976)

[6] Z. Michalewicz: *Genetic Algorithms+Data Structures=Evolution Programs*, Springer-Verlag, Budapest 1992.

[7] V. J. Rayward-Smith and A. Clare, *On finding Steiner vertices*, Networks 16 (1986), 283-294.

[8] I. Rechenberg: *Evolutionsstrategie,* Frommann-Holzboog, Stuttgart, 1973.

[9] H. Takahashi and A. Matsuyama: *An approximate solution for the Steiner problem in graphs*, Math. Japon. 24(6) (1980), 573-577.

[10] graph test problems `http://graph.ms.ic.ac.uk/info.html`, 2000.

Dominik Ďurikovič,
Department of Applied Mathematics, St. Cyril and Metod University,
Square J. Herda 2, 917 01 Trnava,
Slovak Republic
e-mail: d.dominik@zoznam.sk

Received September 2004;