A Parallel Genetic Algorithm for Solving the Probabilistic Minimum Spanning Tree Problem

Zhurong Wang, Changqing Yu, Xinhong Hei, Bin Zhang
School of Computer Science and Engineering
Xi'an University of Technology
Xi'an, China
wangzhurong@xaut.edu.cn

The probabilistic minimum spanning tree (PMST) problem is NP-complete and is hard to solve. However, it has important theoretical significance and wide application prospect. A parallel genetic algorithm based on coarse-grained model is proposed to solve PMST problem in this paper. Firstly, we discuss several problems of determinant factorization encoding, and develop repairing method for illegal individuals. Secondly, a coarse-grained parallel genetic algorithm, which combines message passing interface (MPI) and genetic algorithm, is designed to solve probabilistic minimum spanning tree problems. Finally, the proposed algorithm is used to test several probabilistic minimum spanning tree problems which are generated by the method introduced in the literature. The statistical data of the test results show that the expectation best solution and average best solution obtained by the proposed algorithm are better than those provided in the literature.

Keywords- Parallel genetic algorithm; Probabilistic minimum spanning tree; Determinant factorization encoding; Message passing interface (MPI)

I. INTRODUCTION

The traditional minimum spanning tree problem is a kind of combinatorial optimization problem, and it can be used in many practical problems, such as transportation problem, telecommunications network design problem, and so on. In practical applications, additional constraints are often added to form constrained minimum spanning tree problem. They can be divided into many types depending on the constraints, such as probabilistic minimum spanning tree (PMST) problem [1], quadratic minimum spanning tree problem, degree-constrained minimum spanning tree problem, multicriteria minimum spanning tree problem [2], etc. Theoretical studies have proved that most of the constrained minimums spanning tree problems are NP-hard problems. In this paper, we mainly discuss the PMST problem. For PMST, not all the vertexes are deterministically present in a generated minimum spanning tree, but each vertex is present with certain probability. From a practical viewpoint, it has important applications in transportation, communications, distributed systems, etc.

It is well known that genetic algorithm [3] provides one kind of basic general frameworks for the solution of complicated optimization problems. It is considered an effective approach for solving these kinds of problems [4-6]. Due to its own characteristics of global search, inherent parallelism, robustness, high efficiency, not relying on the problems, genetic algorithm has been demonstrated the good scalability characteristic. However, genetic algorithm has poor capability of local search. Parallel genetic algorithms are proposed to solve the above problems to some extent [7]. It combines high-speed parallelism of parallel computer and the inner parallelism of genetic algorithm together, speeds up the search process of genetic algorithm, keeps and enriches the diversity of population, and reduces the likelihood of the precocious phenomena. Commonly parallel genetic algorithms are divided into three types, namely master-slave parallel genetic algorithm (MSPGA), coarse-grained parallel genetic algorithm (CGPGA) and fine-grained parallel genetic algorithm (FGPGA). Among them, the most common type is CGPGA. Compared with other two parallel genetic algorithms, it has some advantages such as easy realization, good robustness, and high efficiency, etc.

In the paper we mainly discuss the encoding problem and improvement method for probabilistic minimum spanning tree problem, and design a CGPGA for solving PMST problem.

The rest of the paper is organized as follows: Section 2 describes the probabilistic minimum spanning tree problem. Section 3 focuses on determinant factorization encoding method and its improvement. Section 4 discusses problems of coarse-grained parallel genetic algorithm. Experimental studies on the PMST problems are conducted in Section 5. Section 6 concludes the study and discusses possible future works.

II. THE PROBABILISTIC MINIMUM SPANNING TREE PROBLEM

The probabilistic minimum spanning tree problem was proposed by Bertsimas in 1990 [4]. It can be defined as follows: Given a connected undirected graph $G=(V,E)$, a cost function c: $E \to \Re^+$, and a probability function P_i is in between [0, 1], the objective is to find a priori spanning tree, say T, which minimizes the following function:

$$E[L_T] = \sum_{e \in T} c(e)\{1-\prod_{i \in K_e}(1-p_i)\}\{1-\prod_{i \in V-K_e}(1-p_i)\} \quad (1)$$

where K_e, V- K_e, are the subsets of nodes contained in the two sub-trees obtained from T by removing the edge e.

This work is partially supported by National Natural Science Foundation of China (No. 61100173), and the fund of ShaanXi Province education department (No. 2013JK1185, 11JK1038).

978-1-4799-2549-0/13 $31.00 © 2013 IEEE
DOI 10.1109/CIS.2013.20

The expression includes all edges of T. After removed an edge e, T is divided into two sub-trees. The vertices set of one sub-tree is expressed with K_e, and the vertices set of the other one is expressed with V-K_e.

When existence probability P_i of all vertices in the probabilistic minimum spanning tree problem tends to 1, the solution of the probabilistic minimum spanning tree problem will tend to the traditional minimum spanning tree problem; and when P_i tends to 0, the solution will tend to optimal solution of network design problem. Because PMSA problem has the boundary conditions, it may become more complicated and intractable.

III. DETERMINANT FACTORIZATION ENCODING

There are several encoding methods for minimum spanning tree problems, such as the proper vector method [8], the prefix vector representation method, Prüfer encoding, determinant factorization encoding [9].

Among the above encoding methods, determinant factorization encoding is proposed by Faris N. Abuali in 1995. It is a competitive encoding method, and can be effective to many kinds of minimum spanning tree problems. However, individuals are not feasible with this kind of encoding method in many cases. Thus we need to check its validity and repair those infeasible individuals in order to make the algorithm efficient. In the following we first introduce general steps of determinant factorization encoding method, and then discuss and revise those infeasible encoding individuals.

A. Encoding Steps

The realization of the encoding is as follows:

(1) Given a connected undirected weighted graph G, determine its degree matrix. In the degree matrix, edges between two connected vertices are expressed with -1, edges which are not connected are expressed with 0. For instance, in figure 1 the left is a connected undirected weighted graph G constituted by seven vertices, and the right is its degree matrix.

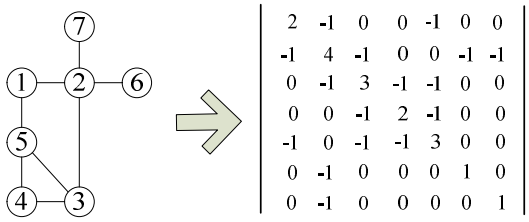

Figure 1 Degree matrix of the connected undirected weighted graph G

(2) Determine the position of -1 from the second column in the degree matrix, and record row number of the position of -1, as shown in figure 2.

Column:	2	3	4	5	6	7
Row Number:	1	2	3	1	2	2
	3	4	5	3		
	6	5		4		
	7					

Figure 2 Number of Rows of -1 in the degree matrix

(3) g, perform mutation and combination for the row number of the position of -1, then can generate $4\times3\times2\times3\times1\times1=72$ kinds determinant factorization encoding strings. For example, 123122 and 123322 are two cases.

B. Checking legitimacy of the encoding

The determinant factorization encoding obtained above may contain illegal encodings. The method for judging whether the determinant factorization encoding string is legal or not can be divided into three steps:

First, if the string of encoding has not number 1, which means that the vertex 1 is isolated, thus this encoding string is illegal. For example, the encoding string 342422 corresponds to a non-tree which is shown in Figure 3.

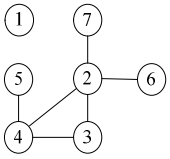

Figure 3 An example tree with encoding 342422

Next, if take the digit of position i (i is from 1 to n-1, n expresses vertex number of the connected undirected weighted graph G) and the position i+1 respectively as the number which are two vertices of the determinant factorization encoding, and take the two vertices combination for a vertex group, there are total n-1 groups. If the two groups of vertex number of vertex are exactly the same, it implies that this encoding of the solution is illegal. For example, the encoding string 342422 is an illegal string. The judgment method is shown in figure 4, where n=7, so i is from 1 to 6, then there are six vertices groups. For the group 1 and the group 2 in the left part, the vertex numbers of the two vertices are the same, i.e., both are vertex 2 and vertex 3. It explains that there are two edges linked between vertex 2 and 3, and one is redundant. So it is illegal. This encoding string which generates illegal solution is shown at right part in figure 4.

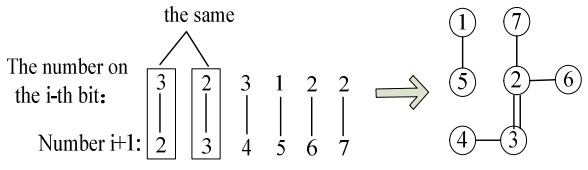

Figure 4 An example to demonstrate the infeasible solution of encoding 323122

Finally, if the methods above cannot determine the encoding string legal or not, then Floyd algorithm can be used to find the shortest paths in the weighted topology. It can accurately check whether a tree is connected or not

C. Revising Eencoding

In this section, we propose some revision methods to the existing problem for the encoding. It is described as follows:

(1) Scan the determinant code, to check that number 1 is in the code. If number 1 is already in the code, then perform the repair described in step (2); otherwise replace a randomly selected allele with number 1.

(2) Identify the set of components in the graph resulting from the determinant code in step (1). If the number of connected components is equal to one, then the determinant code represents a tree, and no repair is required. Otherwise go to step (3).

(3) Let *NC* be the number of components in the graph represented by the determinant code. Suppose that the encoding is illegal, the determinant encoding scheme produces graphs which are not spanning trees, but are *NC* graph. There are two cases for these graphs. One is that these graphs contain spanning trees and the graphs with cycle; the other is that these graphs may only contain spanning trees.

In this paper, we focus on discussing repair method for the second case in step (3). Repair process for the first case can be referred in [9].

It can be seen that, through decoding, an encoding string should include *n*-1 different edges. However, if two edges are the same, we need revise one edge to two different edges. Suppose that it is the *i-th* edge, and two vertex numbers connecting the edge are *i*+1 and *j*, where *j* is a gene position of illegal encoding strings, we replace the vertex number *j* with a number belonging to 1 to *n* except number *i*+1 and number *j*. Thus the illegal encoding is revised. For example, the encoding revision scheme of figure 4 is shown in figure 5. It suppose that we choose the second group, *i*=2, and replace the vertex number 2 with 1. The encoding string is revised as legal encoding string 313122, and its corresponding spanning tree also is shown in figure 5.

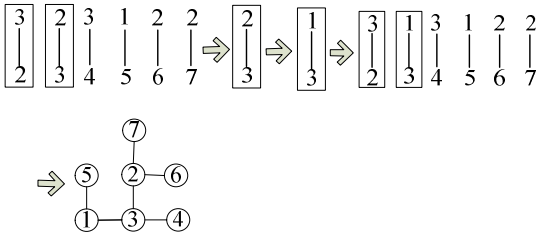

Figure 5 The process that revise the illegal encoding 323122 to legal encoding 313122

If use the first step and the second step of encoding judgment method cannot judge whether a encoding is legal or not, we can use Floyd algorithm. It is judged that the illegal encoding generates *NC* parts, and the *NC* graphs are spanning trees and these graphs are with cycle. In this case, revision method of the illegal encoding described in [9] is as follows:

While the number of connected components *NC* is greater than one, repeat the following steps:

(1) Select randomly a vertex *i* from *T* to become the parent.

(2) Select randomly a component, C_k.

(3) Select randomly a vertex, *j*, from the vertices forming the cycle in component C_k to become the child of parent *i*.

(4) Let *DC* (*j*-1) = *i*.

(5) Let $T=T \cup V(C_k)$, where $V(C_k)$ return the vertices in C_k.

(6) Let *NC*=*NC*-1.

IV. DESIGN THE PARALLEL GENETIC ALGORITHM

The parallel algorithm introduced in this paper is a hybrid parallel genetic algorithm model [10]. This model typically assumes that a global population is divided into subpopulations, which occasionally communicate with each other using message passing interface (MPI). Processes of parallel genetic algorithm are as follows.

(1) Set and broadcast the initial parameters.

Initial parameters of the algorithm include population size, probabilities of crossover and mutation operators, select strategy, max generations, max runs, number of processors, migration operation condition, and population pool initialization, etc.

(2) Generate the initial population for each processor with stochastic method.

(3) Execute genetic operators and generate next population for each processor.

Each processor contains a sub-population. Genetic operators are performed to generate child individuals for each processor independently, and next generation population is generated by some select strategy.

(4) Change individuals of population pool and broadcast the best individual to each processor.

There are three population pools, namely *Mainbuffer*, *Poolbuffer* and *Bestbuffer*. The *Mainbuffer* is used to store the best individuals from main process and each child process, compare the best individuals from each sub-population to find a relatively optimal individual, and then send the best individual to the *Poolbuffer*. The *Poolbuffer* is used to receive the relatively optimal individuals from *Mainbuffer* and store them, then broadcast data to sub-populations of each process. The *Bestbuffer* is used to receive and output the best individual.

(5) Set stop condition.

If the algorithm computing generations reaches the given max generation, stops computing and outputs the best individual to the *Bestbuffer*; else, go to step (3), and continue the computation.

Here three population pools in the above steps are designed so as to exchange data information among subpopulations with smaller communication costs.

V. EXPERIMENTAL STUDIES

A. Test cases

Here the test cases are generated by random method which is introduced according to literature [9]. It takes 100 vertices which are randomly distributed in a 100×100 grid, and the length of each grid is the unit 1. From the 100 vertices randomly select 20, 30 or 40 vertices. Then use these 20, 30 or 40 vertices to generate complete graphs, which are used as the test cases in the paper. For each scale problem, 10 instances are generated to test the algorithm validity.

B. Parameters Setting

Population size is set to 800, processor number is 4, sub-population size is 200, max generation is 200, crossover probability p_c = 0.7, mutation probability p_m = 0.3, fitness function is 1/E [LT]. 1/E [LT] is the expected value of the probabilistic minimum spanning tree problem. Selection strategy is based on roulette wheel selection. Crossover operator is uniform crossover. Mutation operator is single point mutation, and transport operator is generated via every 10 generations computation. Takes out four optimal individuals from four sub-populations respectively, and choose best individual among them. Then use the selected best individual to replace the worst individual for each sub-population.

C. Experiment results and Analysis

The parallel genetic algorithm, which is designed in this paper, is used to solve the instances of the minimum expectations of the probabilistic minimum spanning tree problem generated by the above method, and we compare test data with those in the literature [9].

Table I, table II, and table III include six columns. The first column is test cases with nodes scale being 20, 30, and 40, respectively. The columns from the second to the sixth express the expected active cost obtained by Greedy, Prufer (GA), LNB, Determinant (GA), CGPGA, respectively. The bottom line at each table is average value of the expected active cost among the test cases.

First we compare the expected active cost by different algorithms. It can be seen that, for 10 instances of PMST with 20 nodes, the best expected active cost is 46.93 by the proposed algorithm CGPGA, while its corresponding value is 54.09 by Greedy, 52.5 by Prufer (GA), 50.26 by LNB and Determinant (GA). For the cases with 30 nodes, the best expected active cost is 81.97 by CGPGA, while its corresponding value is 92.82 by Greedy, 95.76 by Prufer (GA), 83.02 by LNB, and 83.81 by Determinant (GA). For the cases with 40 nodes, the best expected active cost is 114.53 by CGPGA, while its corresponding value is 128.33 by Greedy, 135.59 by Prufer (GA), 116.70 by LNB, and 117.36 by Determinant (GA).

It is obvious that the proposed algorithm CGPGA has the best performance, while LNB has better performance than other compared algorithms. At the same time, we also find that though Prufer encoding is suitable to express many kinds of trees, its effect is poor in the solution quality. This may be due to less information could be applied to this encoding individual.

In addition, in terms of the average performance index, CGPGA also shows good characteristics. Because it is hard to obtain all test data obtained from different algorithms at the same platform, it may be more reasonable to compare average value of the expected active cost for all test cases among the compared algorithms. It can be seen that from average value index, CGPGA still outperformed the compared algorithms.

In brief, when taking into account the proposed algorithm's effect in solving the PMST problems, it can be attributed to the following two reasons: 1) the repair process for those infeasible encoding forms makes that CGPGA has higher efficiency; and 2) the parallel mechanism through using three population pools can increase communication efficiency.

VI. CONCLUSIONS

The probabilistic minimum spanning tree problem is a typical kind of constraint minimum spanning tree problem, and it is also a kind of NP-hard problem. It is well known that there are rare effective methods for solving this kind of problems. The coarse-grained parallel genetic algorithm is proposed in this paper, in which determinant factorization encoding is improved for undirected graph. Furthermore, we use MPI method and design the parallel genetic algorithm to solve the probabilistic minimum spanning tree problem. The test data show that the method proposed in this paper has some advantages. The next research will focus on the multi-core parallel computing characteristics, and expand the application areas of the algorithm.

REFERENCES

[1] D.J.Bertsimas. "The probabilistic minimum spanning tree problem". Networks, vol.20, pp. 245-275,1990.

[2] Zhou G., M Gen. "The genetic approach to the multicriteria minimum spanning tree problem". Proceedings of the First Asia-Pacific Conference on Simulated Evolution and Learning.1996, pp.387-394.

[3] [3] J.Holland. "Adaptation in natural and artificial systems". Ann. Arbor: The University of Michigan Press, 1975.

[4] [4] Palmer, C.C., A.Kershenbaum. "An approach to a problem in network design using genetic algorithms". Networks,vol.26, pp.151-163,1995.

[5] [5] Fei Tang, Hongfei Teng. "An improved genetic algorithm and its application in layout optimization". Journal of Software,vol.10, pp.1096-1102,October,1999. (in chinese)

[6] [6] Mistuo Gen, Runchuan Cheng. "Genetic algorithm and the engineering optimization". Beijing: tsinghua university press.2006. (in chinese)

[7] [7] Hongyan Wang,Jingan Yang. "Parallel genetic algorithm research progress". computer science,vol.26,pp.48-53, June,1999. (in chinese)

[8] [8] C.C.Palmer, A.Kershenbaum. "Representing trees in genetic algorithms". In D.Schaffer, H.-P.Schwefel, D.B.Fogel, editors. Proceedings of the First IEEE Conference on Evolutionary Computation. IEEE Press, 1994,pp.379-384.

[9] [9] F. N. Abuali, R. L. Wainwright, D. Schoenefeld. "A new encoding scheme for spanning trees applied to the probabilistic minimum spanning tree problem". Proceedings of the sixth International Conference on Genetic Algorithms.1995,pp.470-477.

[10] Wang Zhu-rong, Ju Tao, Cui Du-wu, Hei Xin-hong. “A Study of Hybrid Parallel Genetic Algorithm Model”. 2011 Seventh International Conference on Natural Computation. 2011,pp.1055-1059.

Table I. Comparison of the expected active cost between the CGPGA and the algorithms in [9] for 20 nodes cases (Node probability is 0.1)

test cases	Greedy	Prufer (GA)	LNB	Determinant (GA)	CGPGA
1	54.09	52.50	50.26	50.26	52.16
2	67.61	61.11	58.08	59.86	48.33
3	62.57	56.97	55.69	55.58	48.54
4	87.09	69.53	62.44	64.75	50.69
5	66.94	60.96	58.26	58.26	49.07
6	57.78	53.05	52.25	52.25	46.93
7	82.95	62.47	62.47	62.89	48.80
8	69.38	64.57	57.92	57.92	49.51
9	60.58	58.10	54.46	55.20	52.75
10	65.77	60.96	59.32	59.45	48.63
average value	67.48	60.02	57.12	57.642	49.541

Table II. Comparison of the expected active cost between the CGPGA and the algorithms in [9] for 30 nodes cases (Node probability is 0.1)

test cases	Greedy	Prufer (GA)	LNB	Determinant (GA)	CGPGA
1	94.29	95.77	87.34	88.1	87.17
2	110.56	109.17	95.98	97.34	90.08
3	109.29	100.69	94.23	96.11	83.61
4	122.97	99.95	90.23	90.83	83.86
5	108.58	95.76	90.11	90.01	81.97
6	92.82	99.44	83.02	83.81	86.74
7	115.58	107.78	98.83	99.55	85.22
8	100.81	99.21	92.41	92.67	83.29
9	106.28	103.26	91.26	93.25	82.46
10	114.95	120.37	101.51	101.24	87.11
average value	107.61	103.14	92.49	93.29	85.15

Table III. Comparison of the expected active cost between the CGPGA and the algorithms in [9] for 40 nodes cases (Node probability is 0.1)

test cases	Greedy	Prufer (GA)	LNB	Determinant (GA)	CGPGA
1	141.64	160.10	116.70	128.73	116.10
2	166.40	150.22	136.86	138.84	117.44
3	156.18	155.11	129.32	128.52	114.53
4	148.52	135.59	119.14	121.74	116.32
5	159.45	159.45	130.84	134.12	117.72
6	128.33	136.20	116.72	117.36	114.89
7	164.41	181.17	130.76	135.73	116.41
8	137.79	161.26	126.74	128.79	115.19
9	165.17	146.30	126.76	136.67	122.63
10	131.09	143.28	124.51	126.45	119.74
average value	149.90	152.87	125.34	129.70	117.10