ATG Press Appetizer

Volume 1, Number 4, December 2003

Important Note: The information, which is provided by this document, is devoted to the entire world for improving creativity, and therefore developing new job opportunities around the globe, in the field of computer science and engineering. So, no part of the proposed information is protected by any intellectual property right. However, the document itself, is copyrighted by Aria Technology Group. No part of this document could be published, in any form of media, without written permission of Aria Technology Group. This is a delayed publication. The actual release date is January 2005.

Globe Optimizer

Farnad Laleh
Aria Technology Group

ABSTRACT

The field of combinatorial optimization and integer programming has a rich literature but a poor progress. It is due to the limited number of fundamental concepts which are applicable in the development of the field, so the major research efforts go through the different combinations of these concepts to reach for a slightly better result. Since it has been proved that most problems in this field are in NP-C class, almost all of the efforts are diverted to finding good approximating methods to the solution of these problems. A few heuristic methods from the older greedy approaches to the newer genetic algorithms are the engines of these approximations. There are also a few exact methods like the dynamic programming and the branch and bound, which have a quite narrow applicability. In this paper, we introduce a new original method to the solution of combinatorial optimization problems, which is called *coordinated brute force* method. Basically, It is an exact method to the solution of NP-C problems, but it can be converted into a record breaking approximating method by a few slight changes. We apply this method to three famous NP-C problems: the traveling salesman as the hardest NP-C, the facility location as a strong NP-C, and the knapsack as a non-strong NP-C.

1. INTRODUCTION

Computer science is an interesting scientific field because of its unique nature: it highly depends on the mathematics and logics so that many scientists believe it as a branch of the mathematics while its progress engine is the intuition (like the arts) not the math. The algorithm as the core concept of computer science is based on the intuition as its *constructor* while the math is used as its *analyzer*.

Among different computer science fields, the filed of combinatorial optimization has a distinctive specification: there are a few basic algorithms to the solution of the problems in the filed while there are many sophisticated analyses around the problems. These analyses include a range of different stuffs from the classification of the problems to the *reduction theory* and *approximability*. Although these stuffs open a wider perspective to the problems but they could not originate new solutions to them.

Addressing the above issue, we introduced *coordinated brute force* method in 1997 [1] but not under this label. Due to the alternative nature of the proposed method, we only introduced its concept intuitively and without any name to avoid any possible misleading. However, the lack of labeling made it hard to categorize the method and the feedbacks we received from researchers during the past eight years prove this claim. Therefore, we decided to explain the concept of the coordinated brute force method in a more complete form. We also give some comparative studies between the proposed method and other classic methods (like the greedy approach) to make it easy for researchers to categorize it.

Since the tradition of this field directs us to analyze the proposed method, we open a way to do this. The unique feature of this method is its hybrid nature: this method is an exact solution to the NP-hard problems (so it is not in polynomial time) while by eliminating a single process in the method, it appears as an perfect approximating method in polynomial time. As this method is a universal one (can be applied to the most of NP-hard problems), a general *approximation threshold* [2] cannot be considered for it and it depends on the case of each problem. In this paper, when we use the term “approximating method” it does not imply the existence of an *approximation scheme* [2].

In section 2 of this paper, we introduce the concept of the coordinated brute force method by introducing it as an intuitive solution to the traveling salesman problem. In section 3, we apply the proposed method to the facility location and the knapsack problems and analyze the performance of the proposed solutions. Finally, in section 4, we discuss how it can be applied to other NP-hard problems and how its performance can be increased when we consider it as an approximating method.

2. COORDINATED BRUTE FORCE METHOD

The brute force algorithm has a very simple concept: try all possible ways and check the results. However, it comes to failure in the case of NP-hard problems as the number of possible ways grows exponentially. The interesting point in the brute force algorithm is that the technique we should use to select all possible ways is an open choice. This is exactly what the coordinated brute force method wants to address. In this method, we select the best choice at first, the first runner up at second and so on. It sounds like a joke: if we know what is the best choice, why should we solve the problem? The answer is we do not know what is the best choice, but we can know it sooner than every other possible choice! The following intuitive approach to the TSP clarifies this concept.

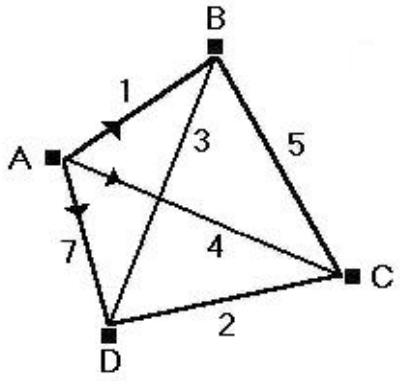

Fig.1. A 4-city non-Euclidean TSP sample

Suppose we are required to find the shortest tour starting from city A in figure 1. Our approach is to send a laser beam from city A to all other cities (e.g. through a fiber optic) simultaneously. As the sent beam from city A is received by any other city, it is modulated in order to indicate the traversed path by the beam. We suppose that the modulation time is negligible and equal to zero. The modulated beam is transmitted again to the other cities. The first beam that has gone through all the cities and reaches the home, i.e. city A, indicates the shortest tour.

As all possible tours are tested in the above approach, it is clearly a brute force approach but a *coordinated* one, i.e., we receive all possible choices in an ascending form. It sounds like a magical solution but it is still an approach and not a complete solution, i.e., we have not proposed any method for the implementation of this approach on a conventional computer (Turing machine). Fortunately, the projected implementation is as simple as the concept itself, using a new powerful concept which is called *space tape* [1]. In fact, space tape is a sequenced view to the proposed approach: if you cannot work in parallel to send beams simultaneously, you can send them sequentially but in their *corresponding* dispatch time and your memory can be used to register the proposed dispatch times.

To implement this concept, we consider a two-dimensional (width × length) memory array as a space tape along with a pointer, which can point at any given time to a position in the length of the tape. This pointer is somehow similar to the pointer in Turing machine but it can move only to forward. The space tape is used to register (store) the beam traveled path and the beam arrival time in each city (dispatching time from that city) in its width and length, respectively. Now we can implement the proposed approach on a conventional computer for the sample in figure 1, as the followings.

Consider city A as the starting city of the shortest tour and the pointer position as zero. The distances from city A to each of the other cities are determined and the corresponding beam path, which we call it sub-path, is registered in the horizontal position that corresponds to the traveled distance of the sub-path and the vertical position corresponds to its arriving city. For example, AB is registered in length 1 and width B, and AC in length 4 and width C. This *allocation* is shown in figure 2. After the initial allocation, the pointer moves one step to forward and comes to position 1. We check the width of the tape in length 1 to see if any sub-path is registered or not. For any registered sub-path, which is only AB in this example, the sub-paths which are started from city B and are not dispatched to the previously traveled cities, are registered in their corresponding position as explained before. Afterward, the pointer moves to the next position. Figure 2 shows the tape situation when the pointer arrives in position 2.

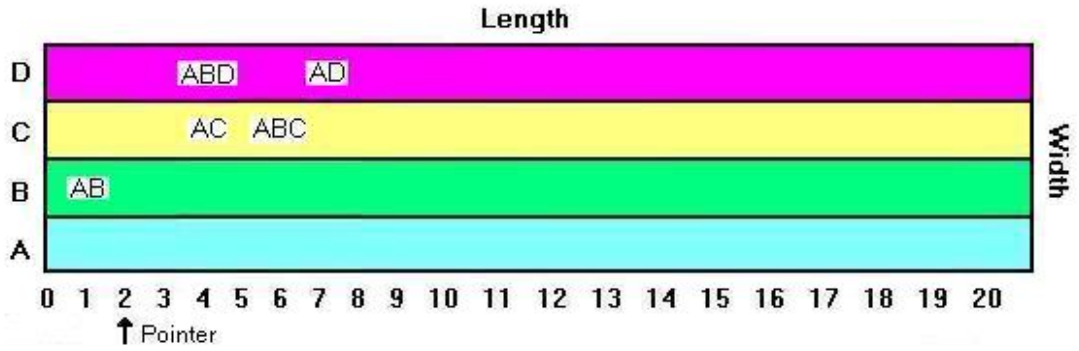

Fig. 2. Space tape situation for the sample in fig. 1 when the pointer arrives in position 2

The mentioned allocation is repeated until the pointer arrives in a position in which a complete tour can be detected. This tour is our required answer and the length in which it has been registered, represents the length of the optimal tour. For the proposed example, it happens in length 10 as is shown in figure 3. As can be seen, the optimal tour is ACDBA or ABDCA in its reverse form.

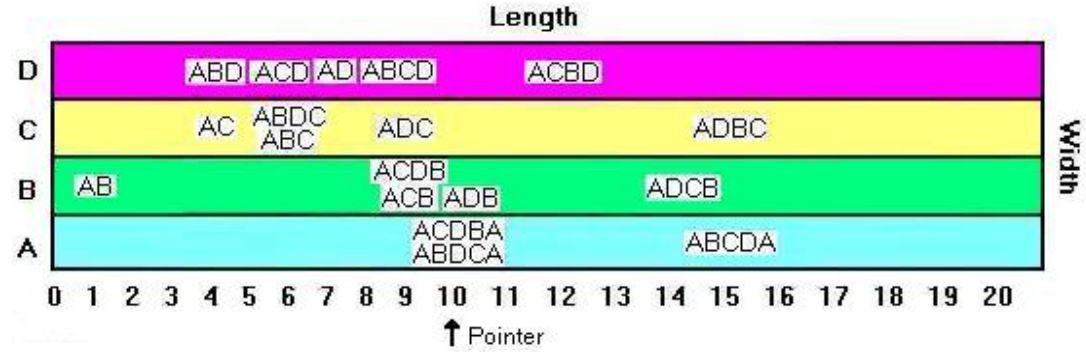

Fig. 3. Space tape situation for the sample in fig. 1 when the pointer arrives in position 10

The proposed algorithm solves the TSP in time $O(nWL)$, where n is the number of cities, W and L are the values of the width and the length of the tape, respectively. It may look like a *pseudo-polynomial* algorithm, but it is not because of the maximum required value of W. To keep this algorithm as an exact problem solver, the width of the tape should have enough space to register all of arriving sub-paths in any position of the length. As the volume of arriving sub-paths in any position does not exceed 2^n (in any position, you can register only one of the sub-paths whose starting, arriving and traveling cities are the same and disregard the others), the proposed algorithm is in time $O(n2^n L)$ which is not apparently a pseudo-polynomial algorithm. If it was, then P = NP [2].

The interesting point is that the dynamic programming can offer a solution to the TSP in time $O(n^2 2^n)$ [2] which is very similar to the time of the space tape method. In addition, if we put a limitation on the allocation in space tape, we reach the well-known Dijkstra's greedy algorithm [3] for the shortest path problem: once a sub-path arrives in a city, do not dispatch other sub-paths to that city. The result is all shortest paths from the starting city to all other cities. In this case, the algorithm is in time $O(n^2 L)$.

You may be surprised by the flexible nature of the space tape method: it looks like the dynamic programming, the greedy approach, and as we explain later, a perfect non-heuristic approximating algorithm! The secret behind this is the strong intuition in the coordinated brute force method. In the next section, we explain how more features of this interesting method could be exploited.

3. ESTABLISHMENT OF GLOBE OPTIMIZER

The space tape method can exactly solve different NP-hard optimization problems. However, the width of space tape grows exponentially in NP-hard problems, so we should find a way to reach for a feasible solution if we cannot reach the exact one. Fortunately, it can be achieved easily by limiting the growth of the width.

In the TSP, the width maximum value, W, can be limited by order of n^2 (sub-paths are categorized by their arriving city and in each arriving city by their number of traveled cities) to obtain an $O(n^3 L)$ approximating algorithm. In this case, if two sub-paths compete for one place in space tape, we should arbitrary eliminate one of the two. If two sub-paths have the same traveled-city histories, the elimination has no effect on the exact solution and so it is called *white elimination*, if not, it is called *gray elimination* because of its potential to disrupt the exact solution. For a given sample of the problem, if we apply the approximating algorithm and a solution without any gray elimination is generated, then we can be sure that it is the exact solution! It is another interesting feature of the space tape method.

Since the TSP is NP-hard, all other NP problems can be *reduced* to it and get a feasible or exact solution by the space tape method. This reduction has a polynomial time overhead and it is more interesting to solve other problems directly by the space tape. Fortunately, the flexible nature of this method makes it possible to apply it directly to the most of combinatorial optimization problems. This is why we call it *globe optimizer*.

The second target problem is the *uncapacitated facility location problem* (UFLP). In this problem, we are given a set of n cities (clients) and a set of m facilities (service points). Every facility has a *setup cost* (opening cost) and between every city and every facility, a *connection cost* is considered. All costs are nonnegative integer values. We are requested to find a subset of the facilities for which the total cost (setup costs plus the costs of connecting every city to an open facility) is minimized. It is apparent that each city should be connected to its nearest open facility, so the problem is to find the best subset of the facilities.

At the first look, the problem seems to have nothing to deal with the space tape. The key point to the solution of any problem using the space tape is the ability to define the structure of sub-paths for that problem. In the UFLP, a sub-path is a string of the names of opened facilities and each position in the string corresponds to a city. Therefore, the maximum length of a sub-path is n (the number of cities). For example, if we have 10 cities marked 1 to 10 and 6 facilities marked A to F, then a complete sub-path would be BAACFACFBC. The proposed sub-path denotes that cities 2, 3, and 6 are connected to facility A; cities 1 and 9 to B; cities 4, 7 and 10 to C; cities 5 and 8 to F; also facilities D and E are not opened.

In the TSP, we cannot meet each city more than once and therefore the name of a city is not repeated in a sub-path (except the starting city). Contrary, we can do this in the UFLP because every facility can serve all cities. This is why the problem is named *uncapacitated* facility location. Each sub-path is registered in the length that corresponds to its total cost. The first complete sub-path in the tape is the solution.

The proposed algorithm is in time $O(mWL)$, where m is the number of facilities, W and L are the values of the width and the length of the tape, respectively. If we limit W by order of n (register every sub-path in a width corresponds to the last connected city), an approximating algorithm in time $O(mnL)$ is obtained. The fact that every city should be connected to its nearest open facility can be applied to this algorithm as a white elimination rule. In this case, a more powerful approximating algorithm in time $O(mn^2L)$ is achieved.

The third target problem is the *knapsack*. We select this problem as the last example for the space tape method because of two interesting features:

1- It is not a *strong* NP-hard problem [2] and therefore a pseudo-polynomial algorithm exists for it. We want to see how the space tape deals with this problem and what is the time order of the resulting algorithm.
2- In contrast to the two previous examples, it is a maximization problem.

In the knapsack, we are given a set of n objects and a knapsack of capacity C. Every object has a *size* and a *profit*. All of the values are positive integers. We are requested to find a subset of the objects so that they can be fit in the knapsack (their total size should not exceed C) and their total profit is maximized.

The structure of a sub-path for this problem is very simple. It is a string of the names of objects which are located inside the knapsack. For example, if there are 6 objects marked A to F, then ADF denotes objects A, D, and F are located inside the knapsack. The permutation of the objects does not matter, e.g., ADF and DFA are equal and can be replaced with each other in the tape. The length in which a sub-path is registered corresponds to the total profit of the sub-path. In addition, a sub-path is dispatched if its total size does not exceed C by adding the new object. Since this is a maximization problem, the solution is the last sub-path in the tape that cannot be dispatched anymore.

This algorithm is in time $O(nWL)$, where n is the number of objects, W and L are the values of the width and the length of the tape, respectively. If we limit W to 1, the resulting algorithm is not only an exact solution to the problem, but also a pseudo-polynomial algorithm in time $O(nL)$. This interesting limitation of W is done by applying a perfect white elimination rule: if two sub-paths compete for one place in the tape, select the one with smaller size. In this paper, we do not prove why this is not a gray elimination rule, but it is easy for the reader to do it.

The interesting point is that the dynamic programming offers a solution to the knapsack in the same time order [2]. More interesting, the dynamic programming cannot be applied to the solution of the more complex *multidimensional knapsack problem* (due to its exponential time order) [4], but the space tape method can do this (although the resulting algorithm is not a pseudo-polynomial one) as it did for the other strong NP-hard problems.

4. ROADMAP FOR GLOBE OPTIMIZER RESEARCHERS

The coordinated brute force method, or its Turing compatible version, the space tape, can be directly applied to many hard combinatorial optimization problems in a similar way to what presented in the previous sections. Among different problems, we can point to the Hamiltonian path, the bin packing, the job scheduling, the graph coloring, and the subset sum problems. We develop the globe optimizer for some of the mentioned problems in ATG Persian Gulf Project. Since we believe in $P \neq NP$ conjecture, we do not recommend researchers to use the space tape for bringing NP into P. Instead, we encourage researchers to develop new approximation schemes for the NP-hard problems using this method.

In contrast to the knapsack problem that has a *fully polynomial* approximation scheme, the TSP cannot have any approximation scheme unless P = NP [2]. However, the UFLP and many other NP-hard problems can have good approximation schemes. Please note that when we use the space tape as an approximating approach, it does not entail that we can determine an approximation threshold for it.

Having the coordinated brute force method around, the use of heuristic methods (like the *genetic algorithms*, the *neural networks*, the *phase transition*, and many others) to the solution of combinatorial optimization problems is somehow unjustifiable. The accuracy of the space tape method can be easily improved by increasing the tape width in order to reduce the gray eliminations. In addition, new white elimination rules may be found in some cases and this is the most wanted research activity in this field.

Finally, we recommend two simple methods to reduce the tape length (a common complexity factor for all samples) in memory. Firstly, the tape can be a circular memory array whose length is the maximum distance between two cities in the problem dataset plus one. Secondly, the numbers in a dataset can be normalized to reduce the length of the tape without any effect on the solution. For example, we can reduce the distance between all of the cities in the TSP by subtracting the minimum distance between two cities in a dataset minus one from all of the distances in the dataset. Anyway, if you apply the space tape method to your datasets, you will find that the tape length is not a challenge.

REFERENCES

[1] M. Kamali, F. Laleh, and A.R. Mirzai, "A New Parallel Approach to the Solution of the Traveling Salesman Problem," in *Proc. IASTED & AAAI Conf. on AI & Soft Computing*, Banff, 1997, pp. 308-310.

[2] C. H. Papadimitriou, "*Computational Complexity*," First Ed. Addison Wesley, 1994.

[3] N. Deo, "*Graph Theory with Applications to Engineering and Computer Science*," First Ed. Prentice-Hall, 1974.

[4] D. Pisinger, "An Exact Algorithm for Large Multiple Knapsack Problems," *European Journal of Operational Research*, 1999, pp. 528-541.