Using Travelling Salesman Problem Algorithms to Plan the Most Time-efficient Route for Robots
Zhaoxin Xue
The High School Affiliated to Renmin University of China, 14-0646

Using Travelling Salesman Problem Algorithms to Plan the Most Time-efficient Route for Robots

1.Abstract

Robots are used as substitutes for humans to perform a wide range of tasks. They can precisely complete every single movement but sometimes not efficiently, especially when performing a large amount of tasks with a great number of movements. This paper will discuss and evaluate the method to improve the efficiency by using the Travelling Salesman Problem (TSP) algorithms.

2.Introduction

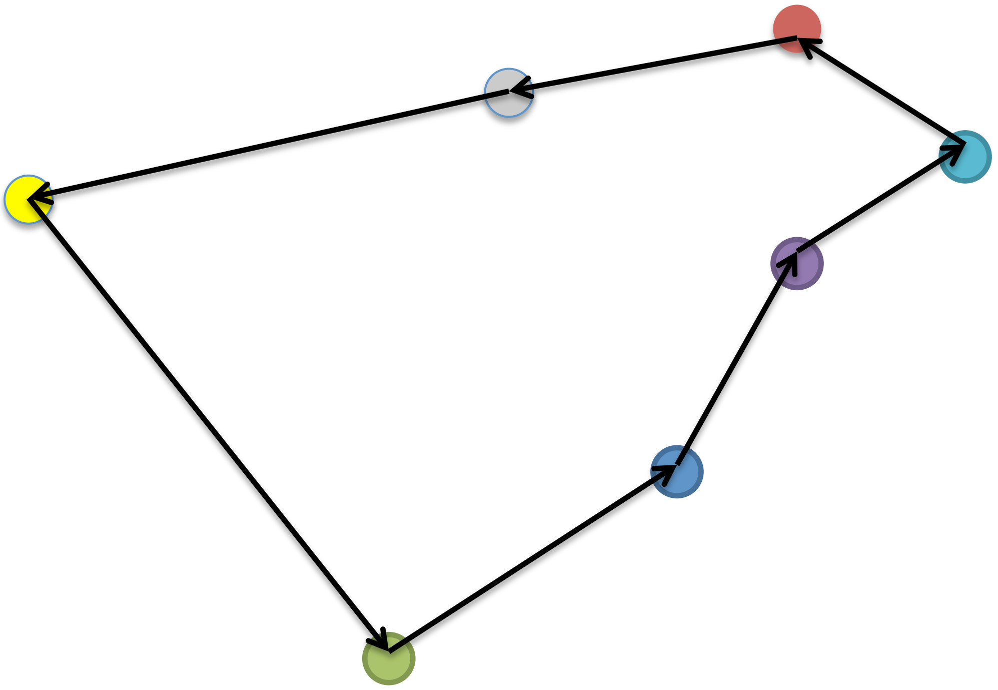
The TSP problem is to find out the shortest route possible of a journey with multiple destinations. When robots are performing a sequence of tasks that need to be carried out one after another under time-constrained conditions, finding out the routes with least most efficient would improve the efficiency. For example, in BotBall we have a 120-second time limit to complete all of our tasks on the game board to get as higher scores as we can. Carrying out those tasks involves in travelling between different locations on the game board. If a team can carefully plan the route for its robots, it can easily get more points than other teams. A typical example would be when a robot needs to collect thousands of small glass balls that are spread on a flat surface. The robot has to move to the first ball, extend the hand, catch the ball and store it. Then it will move to the next ball to do the same thing. However, there are thousands of small glass balls so it is hard to decide the order of picking them up. If the robot just randomly picks a ball to catch every time, it may expend a huge amount of time to finish all the tasks since the robot might choose the one that is the farthest from the current location. It is also impossible to enumerate all the possible routes and compare them to find the fastest one. The issue here becomes more challenging when also considering the different positions of the robot when reaching different locations would consume different amount of time to catch the balls. This is a typical Travelling Salesman Problem, which already has had plenty of algorithms to work out the optimized routes.

3. Applications of TSP related algorithms in robotics

3.1 Overview

Since the total amount of time used by robots is very short and the computing abilities of the controllers are limited, this paper will only discuss the algorithms that do not require strong computational power, but can still get relatively optimized solutions. Furthermore, the algorithms can be easily implemented so it is possible to apply them in the real world situations.

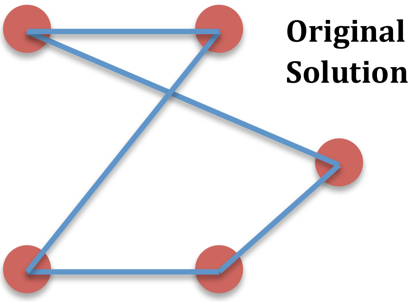
3.2 The Nearest-neighbor Algorithm

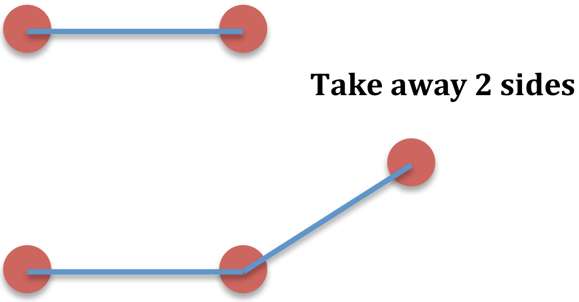


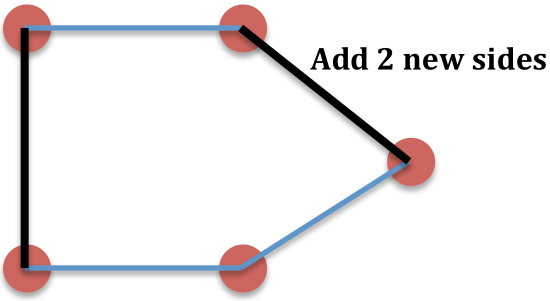
As the name stated, the nearest-neighbor algorithm focuses on the shortest route that is currently available. From the starting point, which is the green point in the figure above, the algorithm will pick the closest location and drive to this location to start operating. After finishing the task, it will pick another location that is closest and go through the same procedure mentioned above every time. In the figure, the robot will complete all the tasks following the route labeled by black line with arrows. The travelling cost will be no greater than $1+\log_2(n)/2$ than the best possible solution (Cook 2013). The implementation of the nearest-neighbor algorithm is relatively easy and it is also easy to be understood by elementary level programmers, but it is not recommended for a large amount of tasks.

3.3 The Lin-Kernighan-Helsgaun Algorithm

The Lin-Kernighan-Helsgaun (LKH) algorithm, developed by Keld Helsgaun, is an improved version of the Lin-Kernighan algorithm (LK). The LKH algorithm changed the 2-opt method, which is using two shorter sides to substitute two longer sides, used in LK algorithm to 5-opt (Cook 2013). The LKH algorithm works by improving the solutions from other algorithms and it can work out the best solution for 100 locations in 1 second (Cook 2013). Viewing from the quality of the solutions and the time taken for calculations, the LKH algorithm is highly applicable to robots. However, it is not easy to implement since it is a bit harder to imagine in mind. Here we use 2-opt method to show how the core idea of the algorithm.







From the figure we can see the original non-optimized route for travelling to five locations. Using the 2-opt method, we take away two longest sides and connect the points again using two shorter sides, which are shown using black lines. The 5-opt method is the same but takes away five sides at a time.

3.4 The Chained Lin-Kernighan Algorithm

The LKH algorithm described above is extended from the LK algorithm. The Chained LK algorithm uses k-opt method to see which value of k will give the best solution for the points, not all the locations on the map, that the algorithm is currently working on. The results from Chained LK algorithm are not as optimized as that of the LHK algorithm but the Chained LK algorithm can be used in a much larger scale than the LKH algorithm. It can obtain the routes that are less than 1% longer than the best possible solutions (Cook 2013). So the Chained LK algorithm is extremely useful for robots executing tasks that involves in thousands of movements, like the production of the circuit board. The implementation of the Chained LK algorithm is more straightforward than that of the LKH algorithm so it is more realistic for the programming of the specific types of robots.

3.5 Comparison

	Speed	**Complexity**	**Quality of Solution**	**Applicability**
Nearest-neighbor	**Slow**	**Simple**	**Medium**	**Low**
LKH	**Fast**	**Very complex**	**Very Good**	**High**
Chained LK	**Fast**	**Complex**	**Good**	**Very High**

The three algorithms mentioned above are all considered suitable to apply to robots. However, the Chained LK algorithm is considered as the most suitable one to use. The Chained LK algorithm is easier to implement and less complex than the similar

LKH algorithm but can still give extremely optimized results. The nearest-neighbor algorithm is the easiest to implement, but it is not applicable to large-scale conditions. The speed of the nearest-neighbor algorithm is another point that needs to be considered carefully since its calculations are very time consuming.

4. Conclusion

In this paper, three effective and easily understandable algorithms were described with the analysis of the usages and the implementation difficulties. Some TSP researchers have already foreseen the great potential that the TSP algorithms have in many industries and areas. But there are still only a few people in the robotics industry that are paying attention to the benefits those algorithms could bring. The applications of the TSP algorithms could lead the whole industry to the next level. Effective and applicable algorithms should be introduced to more people with clear interpretations so everyone can easily start to look for the best route for his or her robots as soon as possible.

5.Reference

Cook, W.J., *In Pursuit of the Travelling Salesman: Mathematics at the Limits of Computation* (C. Sui trans.), Princeton University Press.