Parallel local search algorithm for finding the cheapest hotel in a given time

Jordan Ivanov and Vladimir Zdraveski

December 2022

Abstract- **The goal of this research is finding the cheapest hotel among multiple options within a limited amount of time. The basic local search method, which looks for the best place to go from the current position by repeating the process of finding a local optimum from a small search space, is improved with parallel local search by starting multiple instances of the basic method from random positions, collecting all the local optima and finding the best one. The results of this algorithm will depend on factors such as the time needed to allocate starting places, visit hotels, collect offers and compare offers, as well as the available time to complete the entire process.**

Index Terms*— *Local search, Parallel local search, local optima

1 Introduction

Sometimes we are dealing with the problem to make the best decision about something, but we don't have enough time to research all the possible solution and we need to act fast.

The problem researched in this article is to find the cheapest hotel among a set of hotels, where the information about all the hotels is not given and only the prices of the adjacent hotels are available. We only have a limited amount of time to do this. So we can't check all the hotels.

In reality this problem can be seen in more variations, and at the most cases the searched value, in our case price of the hotels, is not randomly spread around the searched entities. This indicate that we can use some variation of Local search to solve this problem.

2 Related work

Local search algorithms constitute a class of approximation algorithms that are based on the exploration of neighborhoods of solutions. They have shown to be successfully applicable to a wide range of problems and they give good quality solutions. [10]

The most famous local search algorithms are hill climbing, simulating annealing, genetic search, beam search...

Hill climbing method is an optimization technique that is able to build a search trajectory in the search space until reaching the local optima.It only accepts the uphill movement which leads it to easily get stuck in local optima. [8]

Simulating annealing method is inspired by the annealing of the steel, this method does big steps at first in order to miss the local optima and how the time pass the steps are smaller and smaller so they can find global optima. [2]

Genetic search algorithm reflects the process of natural selection where the fittest individuals are selected for reproduction in order to produce better next generation.

Beam search is a heuristic search algorithm that explores a graph by expanding the most promising node in a limited set. Beam search is an optimization of best-first search that reduces its memory requirements.

Some popular problems that can be solved with help of local search are the vertex cover problem, the traveling salesman problem [4], the boolean satisfiability problem [6] , the scheduling problem [5] [7] [1] and others.

However, local search algorithms often require long running time for large instances. One way to reduce the running time is by using parallelism. [10]

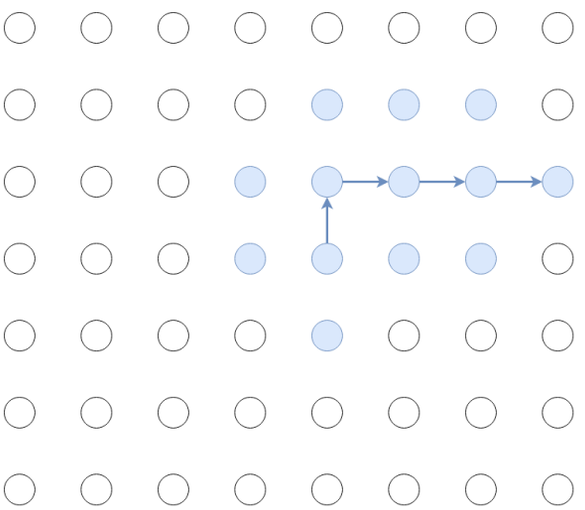
Parallel local search is a form of local search optimization technique, which is based on the concept of parallel computing. It is used to solve optimization problems by running multiple instances of local search algorithms in parallel. This technique can be used to improve the performance of local search algorithms by using multiple processors or computers to work on different parts of the problem simultaneously. The goal of parallel local search is to explore the search space more quickly and effectively than a single processor or computer can do alone. [3] [9]

3 Solution architecture

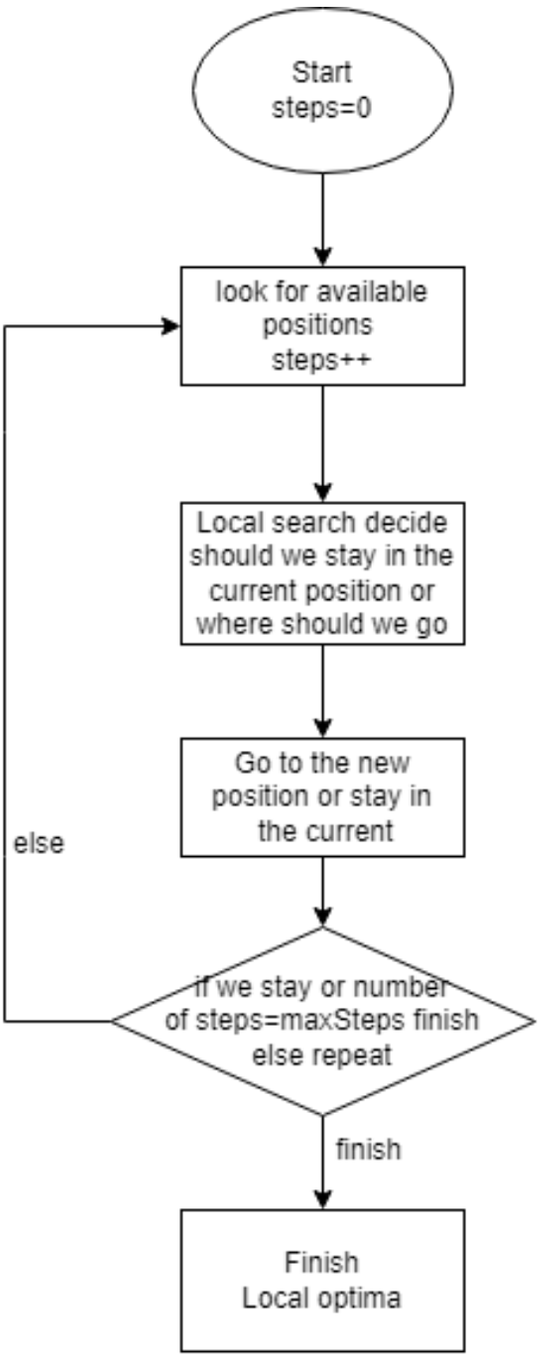
In the following section, we will see how the local search algorithm works.

First we need to start the algorithm from some random position. Then in each iteration, the algorithm chooses a new solution from the neighborhood of the current solution based on hotel price. If the new solution is better than the current solution, it becomes the new current solution and the number of steps is increased by 1. The process continues until no better solution can be found within the neighborhood or the number of maximum allowed steps is reached, at which point the algorithm terminates and the best solution found so far is returned as the result.

This algorithm will find a local optimum from a small searched space(Blue circles below).



Example of basic local search



Basic local search algorithm

We want to improve this method and make it return better result, faster and from a bigger search space, this could be achieved by parallel local search.

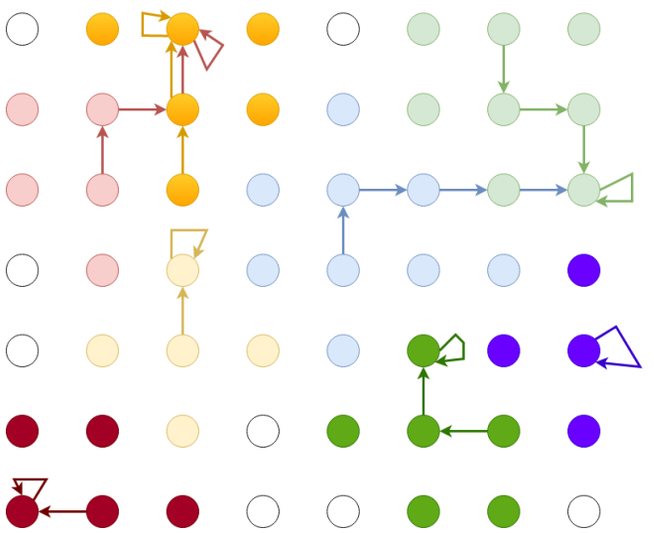
Parallel local search will start more instances of the previous method from random positions, they will still be doing the same thing, finding local op-

tima. Each of this instances will start on different processor or thread giving us the parallelism.

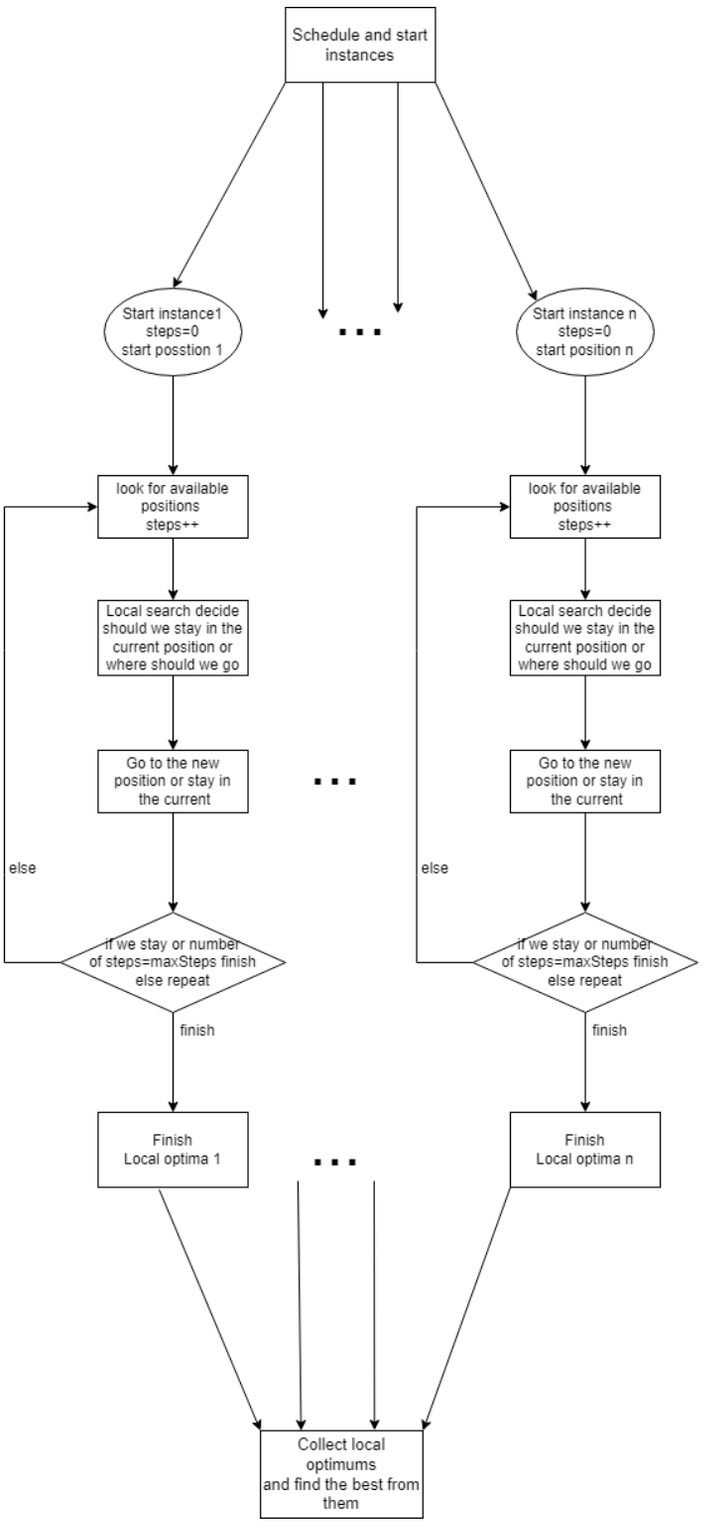
Then we will be collecting all the local optimums, comparing them and finding the best one. This will result in a bigger searched space(Colored circles below), and in the most cases better results.

The cost will be that we will need more time for scheduling the instances, collecting the optimums and finding the best one. Note that some instances can end in the same local optima and we can still not find the global optima.

There can be some modification of this algorithm, like giving the threads opportunities to communicate between them. However, all the instances will last as long as the longest one of them, because all of them need to be finished so the collecting of the local optimums can start and by my opinion the time addition for communication in most cases is not worth it. However if we change the algorithm not only to go to the best solution then to go to the best solution that is not searched maybe we can get some improvement.



Example of parallel local search



Parallel local search algorithm

4 Results

The goals in this research would be how much of the graph is explored and what is the value that will be returned, the results depend on the time needed to allocate starting places, from the time it will take to visit a hotel, the time needed to collect and find the best offer and the time for comparison between offers from the hotel in where we are at the moment and the neighboring hotels. The available time that we have to Finish the entire process is also important.

As an example of these values we will take the following values: 30 seconds * number of people (for number of people¿1) to allocate and collect people and their best offers. Visiting each new hotel would take 100 seconds (travel + visit). And the decision in which direction to go would take 20 seconds. The maximum number of steps will be 4

We will consider multiple options as groups of:

1 person going with Local search, here will be excluded the times to distribute and collect people and their best offers. (100+20)*n for time and n for the number of places visited (MAX n past points +3*n data for hotels for which we have info,this will happen if it goes only in one direction, This is geometric distribution) 120*n seconds Max:2+3*n explored hotels

AVG1= 5 + 3*4/5+ 4/5*(2*2/4+3*1/4) + 4/5*2/4*(2*2/4+3*1/4) +4/5*1/4*(2*2/4+3*1/4) = 9.85 =10 for 4 steps max

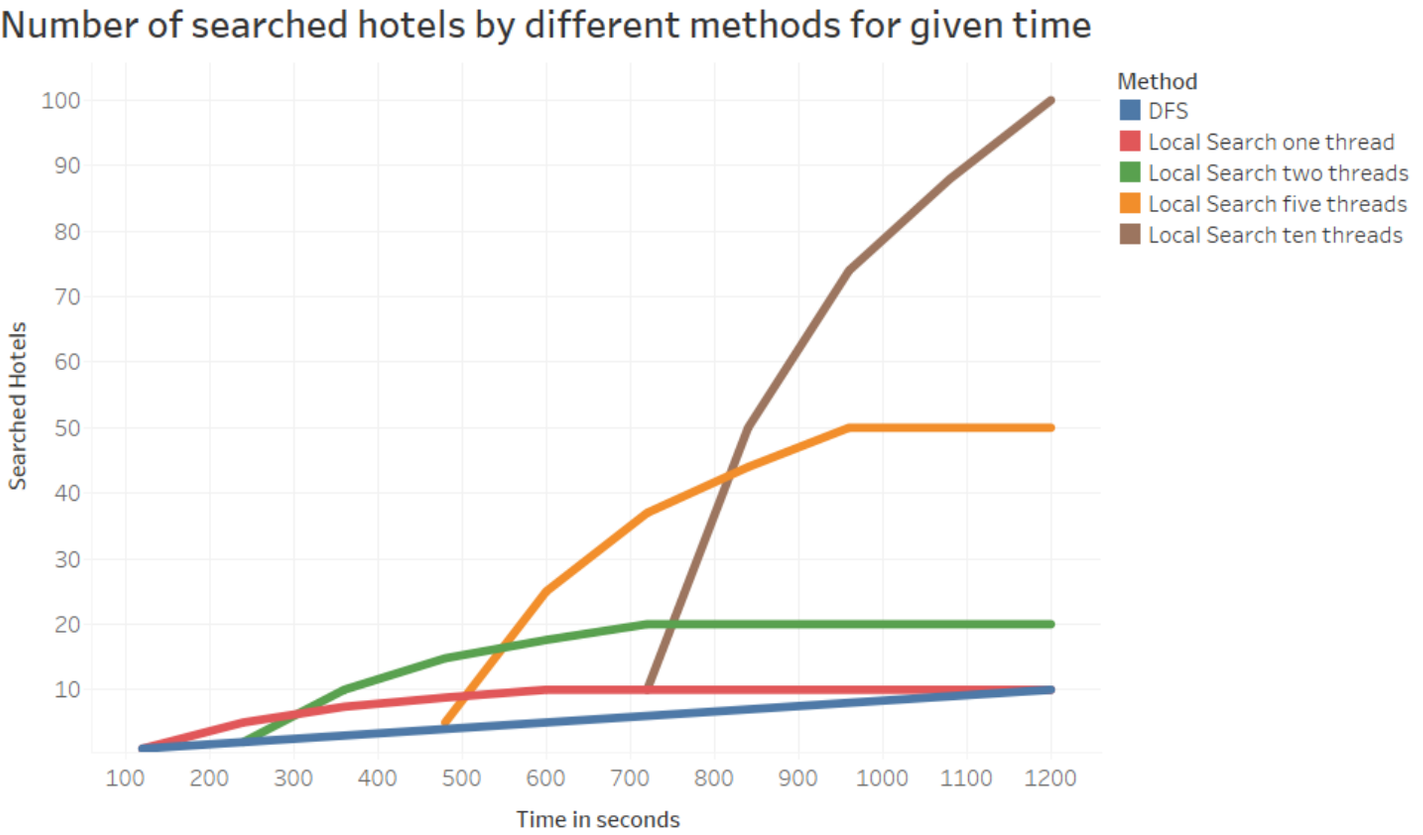
1 man going as DFS, this will exclude the times to allocate and collect people and their best offers and the decision of which direction to go, but additionally it will take time to return to the best offer ((number of passes/2) *30). 100*n+n*15+n*5 (here we will pass n Hotels for sure) 120*n seconds Max:n hotels explored

2 people 2*30+ (100+30)*n + 2*30 time (MAX:n*2 passed points if there is no overlap and all n steps have been passed + MAX:3*n*2 data for hotels for which we have info,this will happen if it goes only in one direction) 120+120*n seconds Max:2*(2+3*n) explored hotels AVG2=2*AVG1=20 assuming no overlap

5 people 5*30+ (100+30)*n + 5*30 (MAX:n*5 passed points if there is no overlap and all n steps have been passed + MAX:3*n*5 data for hotels for which we have info,this will happen if it goes only in one direction) 300+120*n seconds Max:5*(2+3*n) explored hotels AVG5=5*AVG1=50 assuming no overlap

10 people 10*30+ (100+30)*n + 10*30 (MAX:n*10 passed points if there is no overlap and all n steps have been passed + MAX:3*n*10 data for hotels for which we have info,this will happen if it goes only in one direction) 600+120*n seconds Max:10*(2+3*n) explored hotels AVG10=10*AVG1=100 assuming no overlap

Number of searched hotels by different methods for given time					
method seconds	DFS	Local Search one thread	Local Search two threads	Local Search five threads	Local Search ten threads
120	1	1	NA	NA	NA
240	2	5	2	NA	NA
360	3	7.4	10	NA	NA
480	4	8.8	14.8	5	NA
600	5	10	17.6	25	NA
720	6	10	20	37	10
840	7	10	20	44	50
960	8	10	20	50	74
1080	9	10	20	50	88
1200	10	10	20	50	100



5 Conclusion

In this paper, we define local search and gave the various subsets of it. We described how local search can be implemented in solving our problem and gave detailed information of how it works. Then we show how we can start multiple instances of this algorithm and with that achieve parallelism that improve our results.

At the end we gave example of how this algorithm is giving better results than standard search algorithms (as DFS) when we don't have enought time to search the whole graph (Visit all hotels) and show how we can chose the optimal number of instances for given period of time to achieve the best results .

References

[1] NUR NEESHA ALIMIN, NOR ALIZA ABD RAHMIN, GAFURJAN IBRAGIMOV, and NAZIHAH MOHAMED ALI. Multi-start local search for online scheduling in parallel operating theatre. *Advances in Mathematics: Scientific Journal*, 9(12):10915–10927, 2020.

[2] Franco Busetti. Simulated annealing overview. *World Wide Web URL www. geocities. com/francorbusetti/saweb. pdf*, 4, 2003.

[3] Philippe Codognet, Danny Munera, Daniel Diaz, and Salvador Abreu. Parallel local search. In *Handbook of parallel constraint reasoning*, pages 381–417. Springer, 2018.

[4] Gizem Ermiş. *Accelerating local search algorithms for travelling salesman problem using gpu effectively*. PhD thesis, 2015.

[5] Celia A Glass, CN Potts, and P Shade. Unrelated parallel machine scheduling using local search. *Mathematical and Computer Modelling*, 20(2):41–52, 1994.

[6] Cong Peng, Zhongwei Xu, and Meng Mei. Applying aspiration in local search for satisfiability. *PloS one*, 15(4):e0231702, 2020.

[7] Landir Saviniec, Maristela O Santos, and Alysson M Costa. Parallel local search algorithms for high school timetabling problems. *European Journal of Operational Research*, 265(1):81–98, 2018.

[8] Bart Selman and Carla P Gomes. Hill-climbing search. *Encyclopedia of cognitive science*, 81:82, 2006.

[9] Thé Van Luong, Nouredine Melab, and El-Ghazali Talbi. Gpu computing for parallel local search metaheuristic algorithms. *IEEE transactions on computers*, 62(1):173–185, 2011.

[10] Marcus Gerardus Aldegonda Verhoeven and Emile HL Aarts. Parallel local search. *Journal of heuristics*, 1(1):43–65, 1995.