

Parallel Computing for Modified Local Search

JAN PANUŠ

Institute of System Engineering and Informatics

Faculty of economics and administration

University of Pardubice

Studentska 84, Pardubice 53210, Czech republic

CZECH REPUBLIC

jan.panus@upce.cz, <http://www.upce.cz>

Abstract: - The paper deals with using modified type of local search algorithm for optimizing some testing functions and one type of graph methods – travelling salesman problem. The effect of modifying of chosen algorithm is shown on travelling salesman problem and on some testing functions. We will try to modify this algorithm by using principles of parallel computing.

Key-Words: - parallel computing, travelling salesman problem, optimization.

1 Introduction

Parallel computing is the computing and data management infrastructure that will provide the electronic underpinning for a global society in business, government, science, research and entertainment [2], [7]. Desktop machines, engineering workstations, and computer servers with more than one processor connected together are becoming common platforms for design applications. It is therefore extremely important, from the point of view of cost, performance, and application requirements, to understand the principles, tools, and techniques for programming the wide variety of parallel platforms currently available.

Development of parallel software has traditionally been thought of as time and effort intensive. This can be largely attributed to the inherent complexity of specifying and coordinating concurrent tasks, a lack of portable algorithms, standardized environments, and software development toolkits. There are some evident trends in hardware design which indicate that uniprocessor architecture is not possible to realize as sustainable performance in the future.

The speed of the processor is not important for the overall speed of computation also ability of the memory system to feed data to it. Parallel platforms typically yield better memory system performance because they provide larger aggregate caches and higher aggregate bandwidth to the memory system. Localities of data reference, as the principles that are at the heart of parallel algorithms, also lend themselves to cache-friendly serial algorithms.

The past few years have seen a revolution in high performance of many scientific computing applications. Utilization is possible in such science disciplines as computational physics or chemistry. Advances have

focused on understanding processes ranging in scale from quantum phenomena to macromolecular structures. It is possible to use for designing of new materials, understanding of chemical pathways etc. Weather modeling, mineral prospecting, floods predictions, etc., rely heavily on parallel computer and have very significant impact on day-to-day life.

Other utilization of parallel algorithm is in graph theory [10] that plays significant role in computer science because it provides systematic way to model many problems. We will focus on one type of problem from family of graph theory which is travelling salesman problem in next paper of this paper. We solved this problem with modified local search algorithm which we will introduce in next chapter.

2 Modified Local Search Algorithm

Basic parts of modified local search (MLS) algorithm will be show in this chapter. MLS algorithm is inspired by three well-known methods that are used in optimization and in analysis of networks. The first one is simulated annealing [1], second one is basic hill-climbing [4], [8] and the last one is tabu search [5],[6]. Fundamentals of MLS are shown below.

2.1 Solution Features

MLS uses features of primary objective function to characterise solutions. These features can be any property of founded solution that can satisfy the simple constraints that are not trivial. This means that some solutions have the property while others do not. Features of objective function that is solution are

depended on problem and serve as the interface between the algorithm and a particular application.

We used some information that we have on the beginning of local search procedure. Usually we have constraints on objective function features and we know the course of local search [10]. Information that involve to the problem is called the cost of features of objective function. The cost of objective function has the direct or indirect impact on corresponding solution and on solution properties and on the cost of the solution, of course. Information about the search process involve in the solutions visited by local search and in particular local minima. The property U_i represents whether solution has i (means if there is any value in process of local searching) or not. Formula is above

$$U_i = \begin{cases} 1, & \text{solution has } i \\ 0, & \text{otherwise} \end{cases}, s \in S. \quad (1)$$

2.2 Increased Cost Function

The formula increases the cost function g by including a set of penalties in the problem. We construct new cost function h that is defined as follows:

$$h = g + \lambda \times \sum_{i=1}^M p_i \times U_i \quad (2)$$

where M is the number of features defined over solutions, p_i is the penalty parameter corresponding to feature U_i and λ (lambda) is the parameter that regulates behaviour of algorithm. The penalty parameter p_i gives the degree up to which the solution feature U_i is constrained. The parameter λ represents the relative importance of penalties with respect to the solution cost and is of great significance because it provides a means to control the influence of the information on the search process. MLS iteratively uses local search and then modifies the vector of penalty. Every time when local search settles in a local minimum, algorithm takes a modification action on the increased objective function and increases this function by the vector of penalties. Initially, every penalty parameter is set to zero. This situation happens every time the algorithm settles in the local minima. The penalty parameter is always increment by value of one. Information inserted into these actions increases objective function.

2.3 Penalty Modifications

In a local minimum s^* , the penalty parameters are incremented by one for all features U_i that maximise the utility expression:

$$\text{using } f_i = U_i \times \frac{c_i}{1 + p_i}. \quad (3)$$

It means that penalty parameter of feature f_i is used when the actions with maximum value of eq. 3 is found, selected and then performed. We used penalty parameter p_i in eq. 3 to prevent the scheme from being totally biased towards penalising features of high cost. The role of the penalty parameter in eq. 3 is similar to a counter which counts how many times a feature has been penalised. If a feature is penalised many times another feature with the same value but with not so high penalty parameter is giving the chance to also be penalised. The policy implemented is that features are penalised with a frequency proportional to their cost. Features of high cost are penalised more frequently than those of low cost. The effort of local search procedure is to distribute feature costs and the already visited local minima, since only the features of local minima are penalised.

2.4 Regularisation Parameter

Important parameter for local search with penalization is regularisation parameter λ in the augmented cost function in eq. 3. This parameter determines the degree up to which constraints on features are going to affect local search. We test how this parameter is going to affect the moves performed by a local search method. A move alters the solution, adding new features and removing existing features, whilst leaving other features unchanged. If the parameter is larger then the selected moves it will solely remove the penalised features from the solution and the information will fully determine the course of local search. Respectively, if λ is zero then local search will not be able to escape from local minima.

2.5 Definition of an aspiration criterion

Aspiration criterion is an idea that comes from tabu search. In tabu search, an aspiration criterion is any condition under which the status of a tabu move or a tabu attribute may be overridden. The most commonly used form of aspiration criterion is called the improved best aspiration criterion. New improved solution can be obtained by a tabu move when the tabu status of that move is ignored and the move is executed anyway, thus obtaining a new best solution.

We have set of penalties imposed on solution features, rather than a list of tabu solution attributes or a list of tabu moves. So the improved move in MLS is defined as aspiration move to be such that a new best found solution that is generated by that move, and that move would not have otherwise been chosen by the

local search using the increased objective function. Pseudo code for local search with aspiration moves is given below this text.

```

Penalized_Local_Search_with_Aspiration
(x, h, g, N)
{
    Do
    {
        If (Aspiration_Move (x,h,g,N,z))
            x=z
        else
        {
            y = y in N(x) that h(y) is
            minimised
            Δh = h(y) - h (x)
            If (Δh <= 0) then x = y
            If (Δh > 0) then
            iteration = iteration + 1
            Else iteration = 0
        }
        If (g(x) < g(x*)) then x* = x
    }
    While (Δh <= 0) and (iteration < 2)
    Return x
}
    
```

```

Aspiration_Move (x,h,g,N,z)
{
    z = z v N(x) that g(z) is minimized
    if (g(z)<g(x*) and ((h(z)-h(x))>0))
        return true
    else
        return false
}
    
```

where:

- x, y a z are solutions,
- $g()$ returns the cost of a solution with regard to the original cost function,
- $h()$ returns the augmented cost of a solution,
- x^* is the solution of lowest (original) cost found so far by the algorithm,
- $N(x)$ is the neighbourhood function, giving neighbouring solutions of x

3 Testing

We tested the value of λ parameter at every instances of every problem. The value of this parameter varied as reader can see below:

$$\lambda = \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100\} \quad (4)$$

For testing functions was measured following values:

- number of iterations necessary for achievement of optimal solution (or at least sub-optimal),
- the best found solution,
- number of change of the best found solutions – it means how many times we measured change of the best found solution to next better solution in local search procedure. We used many test functions from [3] but for this paper we used one of these functions – Sine envelope sine wave function, function is in eq. 5.

$$-\sum_{i=1}^{n-1} \left(\frac{\sin^2 \left(\sqrt{x_{i+1}^2 + x_i^2} - 0.5 \right)}{\left(0.001 \left(x_{i+1}^2 + x_i^2 \right) + 1 \right)^2} + 0.5 \right), \quad -100 \leq x_i \leq 100 \quad (5)$$

We can see that MLS is pretty strong for finding global minimum for this function even for different value of aspiration criterion. When we connect the algorithm with this aspiration criterion we can see better results than without criterion aspiration. We can see this on found global minimum because the results are flatter and it is not so fluctuating such as MLS without aspiration criterion. One disadvantage is more time consuming of algorithm with aspiration criterion than without the criterion. This is because algorithm uses control element that is looking for whether is solution in aspiration criterion or not. Reward for this is more accurately result. The number of found new best solutions is better with algorithm connected with aspiration criterion.

Another problem that we focus on is to tune regularization parameter λ in that way to get better results in some short time. Really good results are for parameter within range of $\langle 0.4; 0.9 \rangle$ (see fig. 1 and fig. 4). Time consuming and flatness of found results are really pleasure. We can say that this range of parameter is acceptable for another calculation as a started point. The values of parameter within the range of $\langle 1; 100 \rangle$ provide satisfying results but only for time consuming. When we measure excess from the best known results, this is not really good results achieved. We can use this range just for those problems where we do not need so precious results.

We can see on fig. 3 that found solution for algorithm with aspiration criterion is better than without, especially, when λ parameter is within values of $\langle 0.1; 9 \rangle$. MLS with aspiration criterion has better results of the number of iterations than MLS without aspiration criterion – fig. 1 and fig. 2, that is obvious with values of λ parameter in the range of $\langle 0.1, 0.9 \rangle$. The results are not so fluctuating and it looks more flattened when we used MLS algorithm with aspiration criterion. This flatness is possible to explain on the low value of parameter λ . The less is the value the raising of penalization is smaller and the time to consume to local

searching is higher. Because the number is low, the tendency to escape from local minima is very weak and finding of good solution is really strong. The higher is the value of this parameter the weaker is tendency to escape from local minima. The lower is value of the parameter, the really good chance to find good solution.

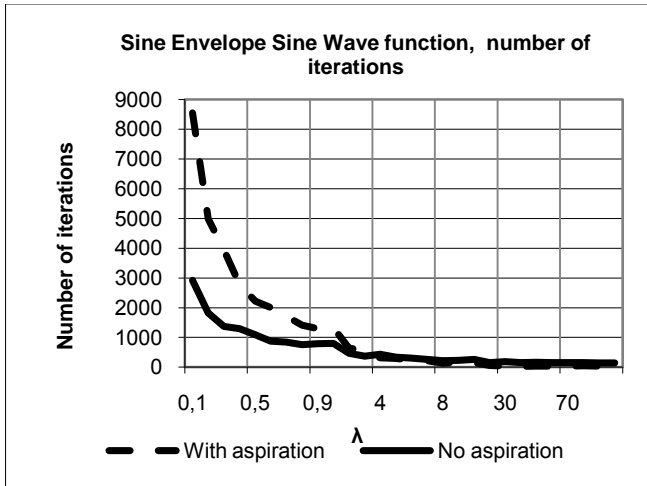


Fig. 1 - Sine Envelope Sine Wave function, number of iterations

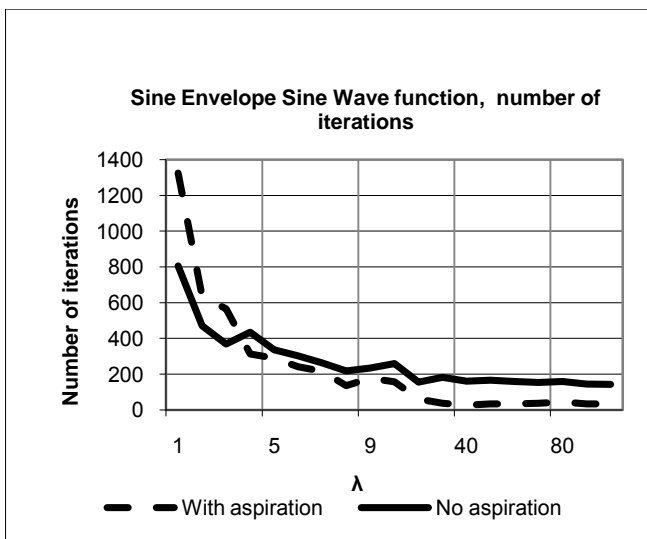


Fig. 2 - Number of iterations for Sine Envelope Sine Wave function - specific range

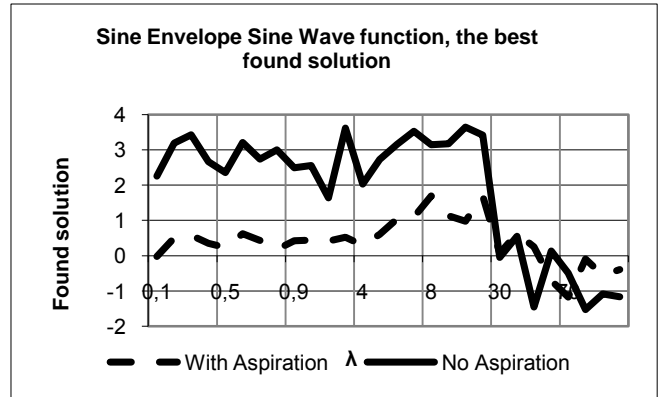


Fig. 3 - The best found solution for Sine Envelope Sine Wave function

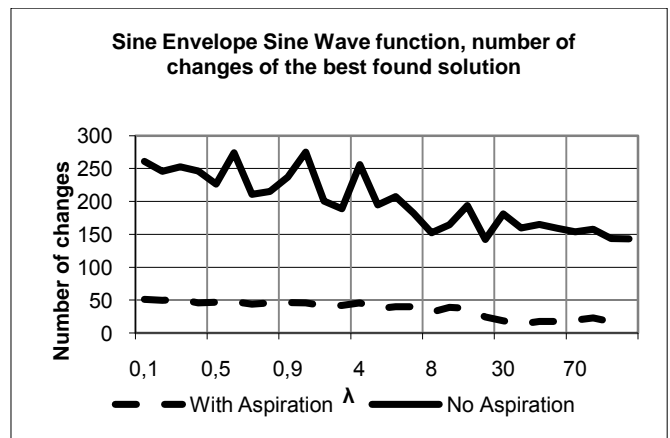


Fig. 4 - Number of changes of the best found solution for Sine Envelope Sine Wave function

4 Graph algorithm and parallel computing

Graph theory plays an important role in computer science because it provides an easy and systematic way to model many types of problems [12]. An undirected graph G is a pair (V,E) , where V is finite set of points, called *vertices* and E is a finite set of *edges*. Directed graph G is a pair (V,E) , where V is set of vertices as we just defined, but an edge $(u,v) \in E$ is an ordered pair. It means that it defines connection between u a v .

A *path* from a vertex v to vertex u is a sequence $(v_0, v_1, v_2, \dots, v_k)$ of vertices. The length of a path is defined as the number of edges in the path. The length can be representing as distance between the first vertices and the last one or it can be a time that should be reached to get from the beginning to the end of path.

The travelling salesman problem is one of the most famous problems in combinatorial optimization. We will examine how MLS with aspiration criterion can be applied to the problem and what results we will reach. We will examine classic symmetric travelling salesman

problem, that is defined by N cities and a symmetric matrix $D=[d_{ij}]$. This matrix gives the distance between any two cities i and j . The goal is to find a tour which visits each city exactly once and is of minimum length. A tour can be represented as a cyclic permutation π on N cities if we interpret $\pi(i)$ to be the city visited after city i , $i=1,\dots,N$. The cost of a permutation is defined as:

$$g(\pi) = \sum_{i=1}^N d_{i\pi(i)} \quad (6)$$

Procedure for paralleling of the problem can be described as follows: The initial configuration is a tour that is defined as a sequence of cities. The length of the tour is represent by value of $g(\pi)$. Each move at a given λ parameter involves the following steps: (1) Generate a new tour by permutations of visiting order of cities, (2) Calculate the length difference between the new and old permutations of cities, (3) Accept of new permutation if the difference less than zero. Parallelization of the problem was in dividing the problem to the groups of smaller problems and computing the problems for each of this problem. It means that we decide to use just a few small groups of permutations to compute. Simulation was set off for two sources each of them computed in the range of permutation given in the beginning. Results were comparing with basic simulated annealing method and with tabu search method.

MLS, simulated annealing and tabu search were tested on 8 instances from TSPLIB [11]. The results are shown in Table 1 where simulated annealing and tabu search are compared with MLS. MLS were set with aspiration criterion and with parameter λ on value of 0.4 because it is best value taken from testing functions.

Table 1 - MLS, Simulated Annealing, and Tabu Search performance on TSPLIB instances.

Problem	MLS with aspiration		Simulated annealing		Tabu search	
	Aver. excess (%)	Aver. time (s)	Aver. excess (%)	Aver. time (s)	Aver. excess (%)	Aver. time (s)
eil51	0	10.46	0.73	6.34	0.00	1.14
eil76	0	10.97	1.21	18.00	0.00	5.24
kroA100	0	12.37	0.42	37.36	0.00	21.34
kroC100	0	11.25	0.80	36.58	0.25	4.80
eil101	0	10.74	1.76	33.29	0.00	61.41
kroA150	0	17.03	1.86	103.32	0.03	413.06
kroA200	0	150.16	1.04	229.38	0.72	776.93
lin318	0.005	346.44	1.34	829.46	1.31	2672.8

As we can see, the superiority of MLS with aspiration criterion over the tabu search variant and simulated annealing is evident. The tabu search variant easily found the optimal solutions for small problems and it scaled pretty well for larger problems. However, it was much slower than PLP and moreover failed to reach the solution quality of PLP in the larger problems. Simulated annealing had a consistent behavior finding good solutions for all problems but sometimes it failed to reach the optimal solutions.

5 Conclusion

We introduced the concept of parallel local search procedures to help understand what effect the created algorithm is having on the search in a defined space. By doing this, we can better evaluate if an algorithm works well as we expect, or if something different is happening. This helps to remove the ad hoc trial and error testing of meta-heuristics, which has become common in the literature. By doing it this way, we have also gained some understanding in how this algorithm works for each problem type we tested.

We have shown some basic heuristic used in local search procedures that can help us understand how this is working. We have shown some methods based on random, population, local searching, and weights.

We also have shown how the algorithm MLS was made by using a basic local searching procedure, plus aspiration criterion from tabu search method, and with penalization. We show by some simple examination of the original objective function whether a new and better solution, than previously found, exists in the current neighborhood. We have shown how MLS algorithm works on some parameter settings and problems.

Finally we can conclude that aspiration criterion works better when it makes the overriding aspiration moves, namely when a new better-than-previous found solution exists in the local search neighborhood. It means that the algorithm could not miss these solutions. We have backed this up by performing a control experiment which performed local search procedure according to the primary objective function, which during the next steps minimizes the increased objective function.

References:

- [1] Aarts, E., Korst, J.: *Simulated Annealing and Boltzmann Machines*. Chichester: J. Wiley and Sons, ISBN 0-471-92146-7 (1989)
- [2] Čech, P., Bureš, V.: *Advanced Technologies in e-Tourism, The 9th WSEAS International Conference on Applied Computer Science*, Genova, WSEAS Press, pp.85-92, ISBN 978-960-474-127-4 (2009)

- [3] Hedar, A. R.: Test Functions for Unconstrained Global Optimization, http://www-optima.amp.i.kyotou.ac.jp/member/student/hedar/Hedar_files/TestGO_files/Page364.htm
- [4] Gass, S.I., Harris, C.M.: *Encyclopedia of Operations Research and Management Science* centennial edition, Dodrecht, Kluwer AP. ISBN 0-7923-7827-X (2004)
- [5] Glover, F.: Tabu search Part I. In: *Journal on Computing*, Vol. 1, Operations Research Society of America (ORSA), pp. 109-206. (1989)
- [6] Glover, F.: Tabu search Part II. In: *Journal on Computing*, Vol. 2, Operations Research Society of America (ORSA), pp. 4 - 32, (1990)
- [7] Jirava, P., Krupka, J. Information System Classification. In *Proceedings of the 8th WSEAS International Conference on Systems Theory and Scientific Computation (ISTAC'08)*. Rhodes, Greece, WSEAS Press, pp. 94-98. ISBN: 978-960-6766-96-1(2008).
- [8] Mladenovic, N., Hansen, P.: Variable Neighborhood Search. In: *Computers in Operations Research*, Vol. 24, No. 11, pp. 1097-1100 (1997).
- [9] Panuš, J., Šimonová, S.: Measurability of evaluative criteria used by optimization methods. In: *Proceedings of Conference on Computational, Intelligence, Man-Machine Systems and Cybernetics*. 1st edition. Dallas, Texas: Wseas Press. pp. 451-455. ISBN 960-8457-55-6. (2007)
- [10] Panuš, J., Šimonová, S.: Pre-Computation of Regional Data for Optimization Analysis. In: *Eurocon 2005*: IEEE Catalog Number 05EX1255C. 1st edition. Beograd, Serbia and Montenegro: School of Electrical Engineering, Beograd. pp. 1-4. ISBN 1-4224-0050-3. (2005)
- [11] Reinelt, G.: A Travelling Salesman Problem Library. In: *ORSA Journal on Computing*. pp. 376-384. (1991)
- [12] Trudeau, R.J. *An Introduction to Graph Theory*. New York: Dover Publications, Inc. ISBN – 9-486-67870-9 (1993)

Acknowledgement:

This paper was supported by grant P403/10/P334 of Czech science foundation.