Lagrangean Metaheuristic for the Travelling Salesman Problem

Rosa Herrero, Juan José Ramos, and Daniel Guimarans
Departament de Telecomunicació i Enginyeria de Sistemes
Universitat Autònoma de Barcelona Spain,
rosa.herrero,juanjose.ramos,daniel.guimarans@uab.cat

ABSTRACT

This paper presents a metaheuristic methodology based on the Lagrangean Relaxation technique, applied to the Travelling Salesman Problem. The presented approach combines the Subgradient Optimization algorithm with a heuristic to obtain a feasible primal solution from a dual solution. Moreover, a parameter has been introduced to improve algorithm convergence. The main advantage is based on the iterative evolution of both upper and lower bounds to the optimal cost, providing a feasible solution in a reasonable number of iterations with a tight gap between the primal and the optimal cost.

Keywords: Lagrangean relaxation, Metaheuristic, Travelling salesman, Subgradient optimization.

INTRODUCTION

The Travelling Salesman Problem (TSP) is probably the best known combinatorial problem: “A salesman is required to visit once and only once each of n different customers starting from a depot, and returning to the same depot. What path minimises the total distance travelled by the salesman?” (Bellman, 1962). The TSP belongs to NP-Hard optimization problems class (Savelsbergh, 1985).

Lagrangean Relaxation (LR) is a well-known method to solve large-scale combinatorial optimization problems. It works by moving hard-to-satisfy constraints into the objective function associating a penalty in case they are not satisfied. An excellent introduction to the whole topic of LR can be found in (Fisher, 1981).

A metaheuristic "refers to a master strategy that guides and modifies other heuristics to produce solutions beyond those that are normally generated in a quest for local optimality" (Glover and Laguna, 1997). Compared with classical heuristics, Lagrangean metaheuristics provides both an upper and a lower bounds (*UB* and *LB*), thus a posterior quality check of the solution obtained. Furthermore, it reduces search space, as dual information can be used to prune decision variables.

The proposed LR-based method uses the Subgradient optimization algorithm combined with a heuristic. Aiming to improve algorithm's convergence on the optimal solution, a heuristic is introduced in order to obtain a feasible solution from the dual variable. Indeed, this method tries to improve the upper bound with the values of these feasible solutions, so a better convergence is obtained.

The present paper is structured as follows: Next section introduces the notation and presents a formulation of the problem. The proposed LR-based method is described afterwards. The following section presents some results of tests on common benchmark instances. Finally, some conclusions and further research topics are presented.

PROBLEM FORMULATION

The symmetric TSP can be defined on a complete undirected graph $G=(I,E)$, connecting the customers set $I=\{1,2,\ldots,n\}$ through a set of undirected edges $E=\{(i,j)\,|\,i,j\in I\}$. The edge $e=(i,j)$ in E has associated a travel cost c_e, supposed to be the lowest cost route connecting node *i* to node *j*.

Solving the TSP consists on determining a route whose total travel cost is minimised and such that each customer is visited exactly once and the route starts and ends at the depot ($i=1$).

The classical formulation requires defining the binary variable x_e to denote that the edge $e=(i,j)\in E$ is used in the route. That is $x_e=1$ if customer *j* is visited immediately after *i*; otherwise $x_e=0$. Thus, TSP can be mathematically formulated as follows:

$$\min \sum_{e\in E} c_e x_e \tag{1}$$

subject to

$$\sum_{e \in \delta(i)} x_e = 2, \qquad \forall i \in I \qquad (2)$$

$$\sum_{e \in E(S)} x_e \le |S| - 1, \qquad \forall S \subset I, |S| \le \frac{1}{2}|I| \qquad (3)$$

where $\delta(i) = \{e \in E : \exists j \in I, e = (i, j) \text{ or } (j, i)\}$ represents the set of arcs whose starting or ending node is i; and $E(S) = \{e = (i, j) \in E : i, j \in S\}$ represents the set of arcs whose nodes are in the subset S of vertices.

Constraint (2) states that every node $i \in I$ must be visited once, that is, every customer must have two incident edges. Subtour elimination constraint (3) states that the route must be a Hamiltonian cycle, so it can not have any subcycle.

LAGRANGEAN RELAXATION METHOD

LR exploits the structure of the problem, so it reduces considerably the problem complexity. However, it is often a major issue to find optimal Lagrangean multipliers. The most commonly used algorithm is the Subgradient optimization (SO). The main difficulty of this algorithm lays on choosing a correct step-size λ_k in order to ensure algorithm's convergence (Reinelt, 1994).

In order to face this limitation, the proposed method combines the SO algorithm with a heuristic to obtain a feasible solution from a dual solution. It can get a better upper bound, so it improves the convergence on the optimal solution starting at an initial *UB* obtained with a Nearest Neighbour Heuristic. In spite of optimality can not be always reached, the proposed method is able to provide a feasible solution with a tight gap between the primal and the optimal cost in a reasonable number of iterations.

Lagrangean Dual Problem

The proposed LR relaxes the constraint set requiring that all customers must be served (2) weighting them with a multiplier vector *u*, since all subcycles can be avoided constructing the solution *x* as a 1-tree. Actually, a feasible solution of the TSP is a 1-tree having two incident edges at each node (Held and Karp, 1971). The advantage is that finding a minimum 1-tree is relatively easy.

The Lagrangean Dual problem obtained is $\max_{u \in \mathbb{R}^n} L(u)$ where $L(u) = \min_{x \text{1-tree}} \sum_{e \in E} c_e x_e + \sum_{i \in I} u_i (2 - \sum_{e \in \delta(i)} x_e)$, defining the subgradient $\gamma_i^k = 2 - \sum_{e \in \delta(i)} x_e$.

Lagrangean Metaheuristic

The proposed LR-based method can be considered a specification of the Lagrangean Metaheuristic presented on Boschetti and Maniezzo (2009). A heuristic obtains a feasible solution from the dual variable, so it tries to improve the *UB* and a better convergence is obtained. Eventually, this feasible solution may be provided as the best solution if the method is stopped. The stopping criterion is based on the maximum number of iterations ($k < max_{iterations}$) and also on a floating-point exception (step-size $\lambda_k < 10^{-15}$). The proposed LR-based method algorithm is shown in Table 1.

0	Initialization
1	Initialize parameters $u^0 = 0; \delta_0 = 2; \rho = 0.95; \alpha_L = 1/3$
2	Obtain an UB applying Nearest Neighbour Heuristic
3	Initialize $\overline{L} = L(u^0) + \alpha_L(UB - L(u^0))$
4	Iteration k
5	Solve the Lagrangean function $L(u^k)$
6	Check the subgradient $\gamma_i^k = 2 - \sum_{e \in \delta(i)} x_e$
7	if $\parallel \gamma^k \parallel^2 = 0$ then Optimal solution is found $\Rightarrow$ EXIT
8	if $\parallel \gamma^k \parallel^2 < \xi$ then apply a heuristic to improve the UB
9	Check the parameter $\overline{L}$
10	Calculate the step-size λ_k
11	Update the multiplier $u^{k+1} = u^k + \lambda_k \gamma^k$
12	$k \leftarrow k + 1$

Table 1: The Proposed LR-based Method Algorithm.

The proposed heuristic to improve the *UB* is applied when the 1-tree is nearly a Hamiltonian path. That is, if the subgradient satisfies $\| \gamma^k \|^2 < \xi$ (step 8). As any solution is a 1-tree, this criterion means that the solution has few vertices without two incident edges. This heuristic replaces an edge $e = (i, j)$ where *j* has some extra edges for an edge $e = (i, l)$ where vertex *l* has one single edge minimising the cost of the exchange. In the presented approach, two different moves have been defined (Figure 1): (a) unlimited move, and (b) limited move only replaces edges which both vertices *i* and *l* are connected to same vertex *j*. First, it iteratively applies the unlimited movement but being aware that it could produce an infinite loop. Then, it iteratively applies the limited movement until finding a feasible solution.

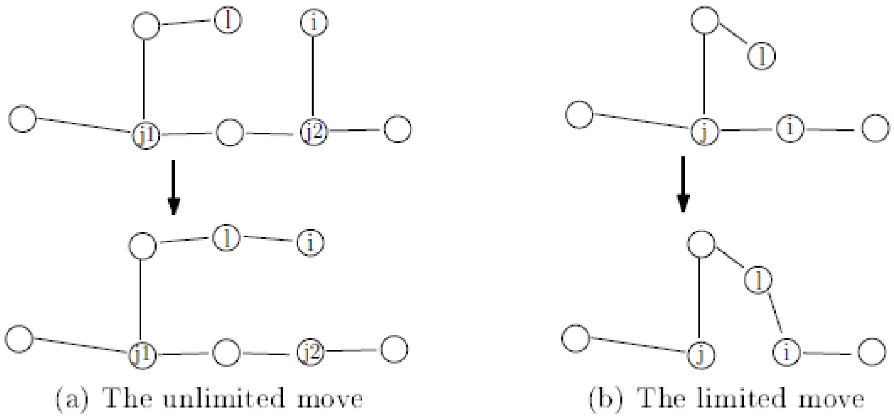

Figura 1: Movements of the heuristic.

A good estimation of the parameter ξ would avoid increasing the computation time. First, its value may be large, for instance the value of the first iteration $\xi = \|\gamma^1\|^2$, but it should be updated whenever a better feasible solution is found according to $\xi = \|\gamma^k\|^2$. If this parameter is not correctly updated, the heuristic becomes time expensive. Eventually, the heuristic could find the optimal solution without detecting it, so the method would continue iterating until $LB = UB$.

As mentioned, algorithm's convergence is critically influenced by the step-size λ_k . This value relies on either the *LB* or the *UB*, which are normally unknown or bad estimated. Therefore, convergence may not be assured for all cases. In order to overcome this limitations, the use of a parameter $\overline{L}$, such that $LB \le \overline{L} \le UB$, is proposed. By definition, this parameter corresponds to a better estimation of the optimal value *L** than those obtained for *LB* and *UB*. The calculation of the step-size turns into:

$$\lambda_k = \delta_k \frac{\overline{L} - L(u^k)}{\|\gamma_k\|^2}$$

Convergence is guaranteed if the term $\overline{L} - L(u^k)$ tends to zero. In turn, convergence efficiency can be improved as long as the new $\overline{L}$ parameter gets closer to the (unknown) optimal solution. The main idea is very simple: as the algorithm converges on the solution, new better lower bounds are known and new better upper bounds estimations can be obtained by using the heuristic designed to get feasible solutions. Therefore, the parameter $\overline{L}$ is updated according to the following conditions:

- It is initialized $\overline{L} = L(u^0) + \alpha_L (UB - L(u^0))$ with $0 < \alpha_L < 1$
- If $L(u^k) > \overline{L}$, it is updated $\overline{L} = L(u^k) + \alpha_L (UB - L(u^k))$
- If $\overline{L} > UB$, then $\overline{L} = UB$

Finally, the parameter δ_k is initialized to the value 2 and is updated as Zamani and Lau (2010) suggest. If the lower bound is not improved, δ_k is decreased, using the formula $\delta_{k+1} = \delta_k \rho$ with $0 < \rho < 1$. On the other hand, if the lower bound is improved, then its value is increased according to the formula $\delta_{k+1} = \delta_k \frac{3-\rho}{2}$ providing that $0 \le \delta_k \le 2$.

RESULTS

The methodology described in the present paper has been implemented in Java. All tests have been performed on a non-dedicated server with an Intel Xeon Quad-Core i5 processor at 2.66GHz and 16GB RAM. In general, different processes were launched to solve different problems, while external applications were active at the same time. For this reason, CPU time is provided just for giving a rough idea of the algorithm computational performance.

A total of 59 symmetric TSP instances have been used to test the efficiency of the proposed approach. They have been obtained from the library TSPLIB (http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/ updated 6 August 2008), a reference site with a large number of instances for the TSP, and related problems, from various sources and of various types.

The experiments have been conducted using the distance according to the specification included in the library, and the number of iterations (300) used by Reinelt (1994) without the setting of parameters $\rho = 0.95$ and $\alpha_L = 1/3$.

Table 2 presents the number of problems solved ordering by size, as well as the average gap of the obtained values *UB*, *LB*, and $\overline{L}$ from the best known value (*BKS*). The gap $\%\Delta\overline{L}$ was calculated as follows $100\left(\dfrac{\bar{L}-BKS}{BKS}\right)$. Therefore, if the gap is negative, the $\bar{L}$ is smaller than the best known solution. Thus, the parameter $\bar{L}$ is a good estimation with regard to an unknown lower bound or an initial upper bound obtained with a Nearest Neighbour Heuristic.

Size	Problems	$\%\Delta UB$	$\%\Delta LB$	$\%\Delta\bar{L}$
$n \leq 200$	27	3.03 %	1.96 %	-0.27 %
$200 < n \leq 500$	13	6.31 %	2.49 %	1.15 %
$500 < n \leq 1889$	19	13.02 %	1.99 %	3.48 %
	59	6.97 %	2.09 %	1.25 %

Table 2: Summary of results obtained.

Problem	BKS	UB	$\%\Delta UB$	t_{UB} (s)	$\%\Delta LB$	t_{Final} (s)
st70	675	691	2.37 %	0.21	0.61 %	0.43
eil76	538	545	1.30 %	0.20	0.19 %	0.63
kroA100	21282	21503	1.04 %	0.11	1.66 %	0.65
rd100	7910	8017	1.35 %	0.20	0.14 %	0.98
eil101	629	629	**0.00 %**	0.68	0.26 %	0.78
lin105	14379	14402	0.16 %	0.09	0.06 %	0.25
pr124	59030	60119	1.84 %	0.44	1.65 %	0.66
bier127	118282	122261	3.36 %	1.37	0.74 %	1.88
ch130	6110	6237	2.08 %	0.56	0.58 %	1.19
ch150	6528	6665	2.10 %	2.82	0.60 %	2.84
rd400	15281	16015	4.80 %	126.2	0.85 %	126.8
pr439	107217	112149	4.60 %	76.0	1.76 %	82.7
d493	35002	36943	5.55 %	371.5	0.72 %	385.0
d657	48912	52996	8.35 %	378.9	1.07 %	390.6
u724	41910	46766	11.59 %	976.6	0.71 %	996.3
rat783	8806	9715	10.32 %	1138.5	0.46 %	1673.5
pr1002	259045	283811	9.56 %	4853.4	1.06 %	4942.9
d1291	50801	56468	11.16 %	3654.2	1.83 %	4167.4
rl1304	252948	275982	9.11 %	5094.8	1.66 %	5554.3
vm1748	336556	384797	14.33 %	27375.5	1.59 %	40146.3

Table 3: Results obtained for some representative problems.

Table 3 presents representative results comparing obtained values to best known solution. The columns shows: *UB* best feasible value, $\%\Delta UB$ percentage distance from optimality of the *UB*, t_{UB} CPU time in seconds to find the best feasible solution, $\%\Delta LB$ percentage distance from optimality of the *LB*, and t_{Final} final CPU time in seconds.

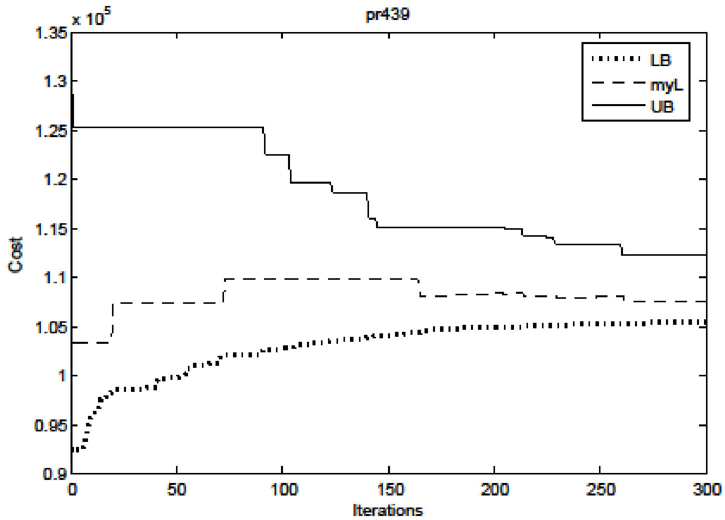

Figure 2: Convergence of LB, $\overline{L}$, and UB in problems pr439.

Figure 2 shows the evolution of upper and lower bounds on one run (problem pr439). As can be seen, the parameter $\overline{L}$ is updated according to the conditions explained. It shows *LB*, $\overline{L}$, and *UB* converge on *BKS* with theirs respective gaps 1.76 %, 0.36 %, and 4.60 %. The initial *UB* obtained with a Nearest Neighbour heuristic has a gap of 24 % then the gap was much reduced.

CONCLUSIONS AND FUTURE WORK

The present paper has presented a metaheuristic methodology based on the Lagrangean Relaxation technique. This scheme has been used to tackle the Travelling Salesman Problem.

The proposed LR-based method uses the SO algorithm combined with a heuristic. Aiming to improve algorithm's convergence on the optimal solution, the heuristic is introduced in order to obtain a feasible solution from the dual variable. On the one hand, the method provides both *UB* and *LB*, thus a posterior quality check of the solution obtained. On the other hand, it reduces search space, as dual information can be used to prune decision variables. Finally, In spite of optimality can not be always reached, the proposed method is able to provide a feasible solution with a tight gap between the primal and the optimal cost in a reasonable number of iterations.

It should be remarked that several lines for future research are open. First, the parameters ρ and α_L must be adjusted with fine-tuning processes. Second, since $\overline{L}$ is a good estimation, the heuristic to obtain a feasible solution from a dual one is to be improved in order to reduce its computation time. Finally, the presented algorithm has been included into a Variable

Neighbourhood Search framework to tackle the Capacitated Vehicle Routing Problem, showing very good results both in terms of the quality of the solution and in terms of computational efficiency (Guimarans et al 2010).

REFERENCES

Bellman, R. (1962), Dynamic programming treatment of the travelling salesman problem, *Journal of ACM* (1), 61-63.

Boschetti, M. and Maniezzo, V. (2009), Benders decomposition, lagrangean relaxation and metaheuristic design, *Journal of Heuristics* 15, 283{312.

Fisher, M. (1981), The lagrangean relaxation method for solving integer programming problems, *Management Science* 27, 1-18.

Glover, F. and Laguna, M. (1997), *Tabu Search*, Kluwer Academic Publishers: London

Guimarans, D., Herrero, R., Riera, D., Juan, A. and Ramos, J. (2010), Combining constraint programming, lagrangean relaxation and probabilistic algorithms to solve the vehicle routing problem, in *Proceedings of 17th RCRA workshop on Experimental Evaluation of Algorithms for Solving Problems with Combinatorial Explosion*, Bologna, Italy.

Held, M. and Karp, R. (1971), The travelling salesman problem and minimum spanning trees: Part II, *Mathematical Programming* 1, 6-25.

Reinelt, G. (1994), *The traveling salesman: computational solutions for TSP applications*, Lecture Notes in Computer Science 840.

Savelsbergh, M. (1985), Local search in routing problems with time windows, *Annals of Operations Research* 4, 285-305.

Zamani, R. and Lau, S. (2010), Embedding learning capability in lagrangean relaxation: An application to the travelling salesman problem, *European Journal of Operational Research* 201(1), 82-88.