Estimation-based Local Search for the Probabilistic Traveling Salesman Problem

Mauro Birattari Prasanna Balaprakash Thomas Stützle Marco Dorigo

Université Libre de Bruxelles (ULB), CoDE, IRIDIA
CP 194/6, Av. F. Roosevelt 50, 1050 Bruxelles, Belgium
{mbiro,pbalapra,stuetzle,mdorigo}@ulb.ac.be

Abstract

In this paper, we propose an effective local search algorithm that makes use of *empirical estimation* techniques for a class of stochastic combinatorial optimization problems. We illustrate our approach and assess its performance on the PROBABILISTIC TRAVELING SALESMAN PROBLEM. Experimental results on a large set of instances show that our approach is very competitive.

1 Introduction

The PROBABILISTIC TRAVELING SALESMAN PROBLEM (PTSP) [1] is a paradigmatic example of a stochastic combinatorial optimization problem. It is similar to the TSP with the difference that each node has a probability of requiring a visit. The *a priori* optimization approach for the PTSP consists in finding an *a priori* solution that visits all the nodes such that the expected cost of *a posteriori* solutions is minimized: The *a priori* solution must be found prior to knowing which nodes actually require being visited and the associated *a posteriori* solution is computed *after* knowing which nodes need being visited. It is obtained by visiting the nodes in the order in which they appear in the *a priori* solution. This paper focuses on a the implementation and study of an iterative improvement algorithm, that is, an algorithm that starts from some initial solution and then iteratively moves to an improving neighboring one until a local optimum is found. Essential for designing and implementing an effective iterative improvement algorithm is that the cost differences among neighboring solutions can be computed efficiently. Currently, the state-of-the-art iterative improvement algorithms for the PTSP, namely, `2-p-opt` and `1-shift` use for this task closed-form expressions based on heavy mathematical derivations [2,3].

2 Estimation-based iterative improvement algorithm for the PTSP

We consider the PTSP as a stochastic combinatorial optimization problem that can be described as: Minimize $F(x) = E\big[f(x,\Omega)\big]$, subject to $x \in S$, where x is an *a priori* solution, S is the set of feasible solutions, the operator E denotes the mathematical expectation, and $f(x,\Omega)$ is the cost of the *a posteriori* solution that depends on a random variable Ω, which is an n-variate Bernoulli distribution and a realization ω of Ω prescribes which nodes need being visited. An unbiased estimator of $F(x)$ of a PTSP solution x can be computed on the basis of a sample of costs of *a posteriori* solutions obtained from M independent realizations of the random variable Ω.

In iterative improvement algorithms for the PTSP, we need to compare two neighboring solutions x and x' to select the one of lower cost. For x', an *unbiased* estimator of $F(x')$ can be estimated analogously $F(x)$ by using a different set of M' independent realizations of Ω. However, in order to increase the accuracy of this estimator, the well-known variance-reduction technique called *the method of common random numbers* can be adopted. In the context of PTSP, this technique consists in using the same set of realizations of Ω for estimating the costs $F(x')$ and $F(x)$. Consequently, we have $M' = M$ and the estimator $\hat{F}_M(x') - \hat{F}_M(x)$ of the cost difference is given by:

$$\hat{F}_M(x') - \hat{F}_M(x) = \frac{1}{M}\sum_{r=1}^{M}\Big(f(x', \omega_r) - f(x, \omega_r)\Big). \quad (1)$$

We implemented iterative improvement algorithms that uses this way of estimating cost differences exploiting a neighborhood structure that uses the *node-insertion neighborhood* on top of the *2-exchange neighborhood* structure, that is, the well-known *2.5-exchange neighborhood.* To make the computaqtion of the cost differences as efficient as possible, given two neighboring *a priori* solutions and a realization ω, the algorithm needs to identify the edges that are not common to the two *a posteriori* solutions. This is realized as follows: for every edge $\langle i, j\rangle$ that is deleted from x, one needs to find the corresponding edge $\langle i^*, j^*\rangle$ that is deleted in the *a posteriori* solution of x. We call this edge the *a posteriori edge* and it is obtained as follows: If node i requires visit, then $i^* = i$, otherwise, i^* is the first predecessor of i in x such that $\omega[i^*] = 1$, that is, the first predecessor for which the realization is one, indicating it requires visit. If node j requires visit, then $j^* = j$, otherwise, j^* is the first successor of j such that $\omega[j^*] = 1$. Recall that in a *2-exchange* move, the edges $\langle a, b\rangle$ and $\langle c, d\rangle$ are deleted from x and replaced by $\langle a, c\rangle$ and $\langle b, d\rangle$. For a given realization ω and the corresponding *a-posteriori-edges*, $\langle a^*, b^*\rangle$, $\langle c^*, d^*\rangle$, the cost difference between the two *a posteriori* solutions is given by $c_{a^*,c^*} + c_{b^*,d^*} - c_{a^*,b^*} - c_{c^*,d^*}$, where $c_{i,j}$ is the cost of edge $\langle i, j\rangle$. The procedure described can be directly extended to *node-insertion* moves. Furthermore, the proposed algorithm adopts classical neighborhood reduction techniques such as *fixed-radius search*, *candidate lists* and *don't look bits.* We denote the proposed algorithm `2.5-opt-EEs`. For further reference, we refer the reader to [4].

Here we report some example results obtained on *clustered homogeneous* PTSP instances of 1000 nodes, which are arranged in a $10^6 \times 10^6$ square and where each node has a same probability p of appearing in a realization. (For more results see [4]). We considered a probability range in $[0.1, 0.5]$ with a step size of 0.1; 100 instances were generated for each probability level. All algorithms were implemented in `C` and the source codes were compiled with `gcc`, version 3.3. Experiments were carried out on AMD Opteron$^{\text{TM}}$244 1.75GHz processors with 1 MB L2-Cache and 2 GB RAM, running under Debian GNU/Linux. The nearest-neighbor heuristic is used to generate initial solutions. The *candidate list* is set to size 40 and it is constructed with the *quadrant nearest-neighbor* strategy. Each iterative improvement algorithm is run until it reaches a local optimum. The number of realizations in `2.5-opt-EEs` is set to 1000; in order to compare the cost of the *a priori* solutions reached by each algorithms, we used the closed-form expression that computes the

Table 1: Mean and standard deviation (s.d.) of final solution cost and computation time in seconds.

	Algorithm	Solution Cost		Computation Time	
		mean	s.d.	mean	s.d.
$p = 0.1$	`2.5-opt-EEs`	5069560	424448	9.487	1.170
	`1-shift`	5178144	469977	635.757	84.010
	`2-p-opt`	5365486	449318	1464.535	341.993
$p = 0.2$	`2.5-opt-EEs`	6681179	475423	3.981	0.447
	`1-shift`	6744906	494658	547.263	71.878
	`2-p-opt`	6978843	477590	859.276	159.102
$p = 0.3$	`2.5-opt-EEs`	7875735	511413	2.658	0.306
	`1-shift`	7982498	531787	451.773	58.575
	`2-p-opt`	8175022	547812	552.554	95.447
$p = 0.4$	`2.5-opt-EEs`	8846373	594006	2.026	0.198
	`1-shift`	8995824	567472	374.877	50.938
	`2-p-opt`	9060142	551377	422.211	76.872
$p = 0.5$	`2.5-opt-EEs`	9591076	635788	1.689	0.149
	`1-shift`	9856073	579796	316.049	44.883
	`2-p-opt`	9799426	594452	338.203	63.679

exact cost [1]. The computational results, measured across the 100 instances, are shown in Table 1. Regarding the time required to reach local optima, irrespective of the value of p, `2.5-opt-EEs` is approximately 2.5 orders of magnitude faster than the state-of-the-art algorithms, `1-shift` and `2-p-opt`, respectively. Concerning the average cost of local optima, `2.5-opt-EEs` achieves an average cost which is approximately 2% lower than that of `1-shift` and `2-p-opt`; the observed differences are always statistically significant according to the paired Wilcoxon test (α=0.05)

The main novelty of our approach consists in using the *empirical estimation* techniques in the *delta evaluation* procedure. The proposed approach is conceptually simple, easy to implement, scalable to large instance sizes and can be applied to problems in which the cost difference cannot be expressed in a closed-form. Our future work will aim at developing adaptive sampling procedures that save computational time by selecting the most appropriate number of realizations with respect to the variance of the cost difference estimator.

References

[1] Bertsimas D. J, Jaillet P., and Odoni A. (1990): "A priori optimization". In: *Operations Research* **38**, 1019–1033.

[2] Bertsimas D. J. (1998): "Probabilistic Combinatorial Optimization Problems". Ph.D. thesis. Massachusetts Institute of Technology, Cambridge, MA.

[3] Bianchi L. (2006): "Ant Colony Optimization and Local Search for the Probabilistic Traveling Salesman Problem: A Case Study in Stochastic Combinatorial Optimization". Ph.D. thesis. Université Libre de Bruxelles, Brussels, Belgium.

[4] Birattari M., Balaprakash P., Stützle T., and Dorigo M. (2007): "Estimation-based local search for stochastic combinatorial optimization". Technical Report [TR/IRIDIA/2007-003]. IRIDIA, Université Libre de Bruxelles, Brussels, Belgium.