Heuristic Approaches for the Probabilistic Traveling Salesman Problem

Christoph Weiler[1], Benjamin Biesinger[1], Bin Hu[2], and Günther R. Raidl[1]

[1] Institute of Computer Graphics and Algorithms, TU Wien
Favoritenstraße 9-11/1861, 1040 Vienna, Austria
christoph.weiler@tuwien.ac.at
{biesinger|raidl}@ac.tuwien.ac.at
[2] AIT Austrian Institute of Technology
Mobility Department - Dynamic Transportation Systems
Giefinggasse 2, 1210 Vienna
bin.hu@ait.ac.at

Abstract The Probabilistic Traveling Salesman Problem (PTSP) is a variant of the classical Traveling Salesman Problem (TSP) where each city has a given probability requiring a visit. We aim for an a-priori tour including every city that minimizes the expected length over all realizations. In this paper we consider different heuristic approaches for the PTSP. First we analyze various popular construction heuristics for the classical TSP applied on the PTSP: nearest neighbor, farthest insertion, nearest insertion, radial sorting and space filling curve. Then we investigate their extensions to the PTSP: almost nearest neighbor, probabilistic farthest insertion, probabilistic nearest insertion. To improve the constructed solutions we use existing 2-opt and 1-shift neighborhood structures for which exact delta evaluation formulations exist. These are embedded within a Variable Neighborhood Descent framework into a Variable Neighborhood Search. Computational results indicate that this approach is competitive to already existing heuristic algorithms and able to find good solutions in low runtime.

Keywords: probabilistic traveling salesman problem, variable neighborhood search, construction heuristics

1 Introduction

The Probabilistic Traveling Salesman Problem (PTSP) is an NP-hard problem [6] introduced by Jaillet [10]. It is a variant of the Traveling Salesman Problem (TSP), where each city has an assigned probability of requiring a visit. We search for an a-priori tour through all cities that minimizes the expected length of the real tour, where some cities might not have to be visited. The real tour, also called realization of the a-priori tour, then follows this a-priori tour and skips cities which do not have to be visited. A real world application for the PTSP would be, e.g., a postman who delivers mails on a fixed assigned route every day.

From historical data the probability of each address requiring a visit is known. After each delivery he checks which address he has to visit next and proceeds accordingly.

Formally we are given a complete graph $G = \langle V, E\rangle$ with V being a vertex set containing n nodes and E being an edge set containing m edges. Each edge $(i, j) \in E$ is assigned a cost d_{ij} and each node $v \in V$ has a probability p_v of requiring a visit. If the probabilities p_v are equal for every node, the problem is called homogeneous and otherwise it is heterogeneous [9]. A solution $T = \langle v_1, \ldots, v_n\rangle$ is an a-priori tour, represented by a permutation of the n nodes, where the first and last nodes are implicitly assumed to be connected. In this paper we concentrate on the homogeneous PTSP with symmetric distances due to comparability with other papers.

The expected costs of a tour T are defined as

$$c(T) = \sum_{i=1}^{2^n} p(R_i)L(R_i) \tag{1}$$

where R_i is a possible realization, i.e., one possible a-posteriori tour, $p(R_i)$ its occurrence probability, and $L(R_i)$ the resulting length of the tour. Since there are $O(2^n)$ different realizations, it is not convenient to compute the objective value in such a way. Therefore Jaillet [10] showed that the expected length can be calculated in $O(n^2)$ time using the following closed form expression:

$$\begin{aligned} c(T) = \sum_{i=1}^{n} \sum_{j=i+1}^{n} d_{v_i v_j} p_{v_i} p_{v_j} \prod_{k=i+1}^{j-1} (1 - p_{v_k}) + \\ \sum_{j=1}^{n} \sum_{i=1}^{j-1} d_{v_j v_i} p_{v_i} p_{v_j} \prod_{k=j+1}^{n} (1 - p_{v_k}) \prod_{k=1}^{i-1} (1 - p_{v_k}) \end{aligned} \tag{2}$$

This formula represents the most general form for heterogeneous PTSPs. As we concentrate on the homogeneous problem, we will use the following, simplified objective function:

$$c(T) = p^2 \sum_{r=1}^{n-1} (1-p)^{r-1} \sum_{i=1}^{n} d_{v_i v_{i+r}} \tag{3}$$

The PTSP is a well studied problem in the literature. Bertsimas et al. [4, 6] contributed theoretical properties of the PTSP such as bounds and asymptotic analyses. Bianchi et al. [7, 8] proposed metaheuristic approaches based on ant colony optimization and local search with exact delta evaluation. Balaprakash et al. [1, 2] analyzed sampling and estimation-based approaches. Weyland et al. [14, 15] considered new sampling and ad-hoc approximation methods for local search and ant colony system. Marinakis and Marinaki [11] proposed a hybrid swarm optimization approach. The most promising results were obtained by using not the exact objective function as stated in Eq. 3, but either a restricted depth approximation or a sampling approach. In contrast, our approach is based on an efficient exact evaluation.

In Section 2 we apply several TSP construction heuristics to the PTSP, consider Jaillet's Almost Nearest Neighbor Heuristic [10], and introduce two new construction heuristics derived for the PTSP: Probabilistic Farthest Insertion and Probabilistic Nearest Insertion. In Section 3 we propose a Variable Neighborhood Descent Framework embedded in a Variable Neighborhood Search to improve constructed solutions. Section 4 presents experimental results and Section 5 concludes this work.

2 Construction Heuristics

For generating an initial solution for the subsequent VND / VNS we consider several different construction heuristics. First we investigate construction heuristics having already been used for the TSP and then we improve upon them by taking also the probabilities into account.

2.1 TSP Construction Heuristics on PTSP

To construct a reasonable tour for PTSP we first investigate how well TSP construction heuristics perform. We evaluate the following construction heuristics:

Nearest Neighbor (NN): Starting from a first node $v_0 \in V$, we iteratively append an unvisited node that is nearest to the previously inserted one. The resulting computational complexity is in $O(n^2)$.

Nearest Insertion (NI): Starting with a simple tour T containing only one node $v_0 \in V$, we insert the nearest node to the previously inserted one at the best fitting position. Let v_j be the node to be inserted next, then we calculate the best fitting position by determining

$$\min_{v_i, v_{i+1} \in T} (d_{v_i v_j} + d_{v_j v_{i+1}} - d_{v_i v_{i+1}}) \tag{4}$$

The computational complexity is in $O(n^2)$.

Farthest Insertion (FI): This heuristic is similar to NI. The only difference is that we choose the farthest node to the previously inserted one is chosen for insertion instead of the nearest one.

Space Filling Curve (SFC): SFC was introduced by Bartholdi et al. [3]. The heuristic constructs a Sierpiński curve over all cities and visits them as they appear on this curve. The computational complexity is in $O(n \log n)$.

Radial Sorting (RS): We construct a new virtual city which can be seen as the center of mass of all cities. The cities are then sorted and visited by their angle relative to this center. For TSP this heuristic usually yields poor results, but for stochastic vehicle routing problems with low probabilities it can perform really well [5]. The computational complexity is dominated by the sorting, and thus, is in $O(n \log n)$.

2.2 Construction Heuristics for PTSP using probabilities

To take probabilities into account we adapt the first three heuristics from Section 2.1.

Almost Nearest Neighbor Heuristic (ANN): ANN was mentioned by Jaillet [10] in his dissertation, but it did not gain much attention until now. It appends the city with the lowest change of expected length from the last inserted city to the tour. The cost of inserting city v_j can be computed the following way for heterogeneous problems:

$$\min_{v_j \in (V-T)} \left(\sum_{i=1}^{|T|} p_{v_i} p_{v_j} d_{v_i v_j} \prod_{k=i+1}^{|T|} (1 - p_{v_k}) \right) \tag{5}$$

For the homogeneous problem we can simplify the formula:

$$\min_{v_j \in (V-T)} \left(\sum_{i=1}^{|T|} d_{v_i v_j} (1-p)^{(|T|-i)} \right) \tag{6}$$

Note that we omitted the p^2 in the homogeneous formula because we try to find a minimum and therefore scaling by p^2 does not matter. The resulting computational complexity is in $O(n^3)$ because we insert each city in the tour and evaluate equation 6 on each position in the tour.

Probabilistic Nearest Insertion (PNI): PNI is a new heuristic derived from NI where we insert the node nearest to the last inserted node into the tour and evaluate the objective function on each possible position. This naive approach results in a computational complexity of $O(n^4)$. We use Bianchi et al.'s delta 1-shift [8] local search procedure to solve this heuristic more efficiently: Bianchi showed that 1-shift local search is possible in time $O(n^2)$ using delta evaluation. Therefore we insert the node at the first position in the tour and then perform one iteration of the delta 1-shift procedure. Therefore we are able to reduce the complexity to $O(n^3)$.

Probabilistic Farthest Insertion (PFI): PFI is a new heuristic similar to PNI. The only difference is that the farthest node to the previously inserted one is chosen for insertion instead of the nearest one.

3 Variable Neighborhood Search

Variable Neighborhood Descent (VND) and Variable Neighborhood Search (VNS) were introduced by Mladenović and Hansen in 1997 [12]. VND uses the fact that if a solution is at a local optimum in some neighborhood it is not necessarily locally optimal in another neighborhood. We use delta 2-opt and delta 1-shift neighborhood structures by Bianchi et al. [8] to combine them within a VND framework because they showed that it is possible to exactly evaluate them using delta evaluation formulations. Therefore, we compute a solution that is locally optimal with respect to both neighborhoods.

Via a general VNS we repetitively randomize the tour by applying shaking, and locally improving it again with VND. In the i-th shaking neighborhood we perform $2i$ random shift moves. Our VNS terminates after 20 iterations without improvement.

4 Computational Results

The algorithms were implemented in C++99 and compiled with GNU GCC 4.6.4. As test environment we used an Intel Xeon E5649, 2.53 GHz Quad Core, running on Ubuntu 12.04.5 LTS (Precise Pangolin). For a comparison with the literature we use test instances from the TSPLIB [13]. Homogeneous visiting probabilities were assumed to be $p \in \{0.1, 0.2, \ldots, 0.5\}$.

Table 1. Results of construction heuristics.

Instance	p	RS	t[s]	SFC	t[s]	NN	t[s]	ANN	t[s]	FI	t[s]	PFI	t[s]	NI	t[s]	PNI	t[s]
eil101	0.1	199.3	<1	206.5	<1	243.9	<1	243.4	<1	202.7	<1	**197.4**	<1	235.3	<1	197.6	<1
	0.2	301.8	<1	301.8	<1	372.7	<1	367.3	<1	296.6	<1	286.9	<1	353.0	<1	**285.2**	<1
	0.3	406.7	<1	377.3	<1	465.4	<1	447.1	<1	373.8	<1	358.3	<1	435.1	<1	**350.6**	<1
	0.4	515.8	<1	442.2	<1	539.7	<1	506.6	<1	439.6	<1	**413.4**	<1	500.6	<1	417.1	<1
	0.5	627.6	<1	500.8	<1	602.4	<1	610.3	<1	496.9	<1	493.8	<1	556.4	<1	**471.6**	<1
d198	0.1	8580.7	<1	7971.6	<1	9010.2	<1	9128.4	<1	7677.0	<1	**7438.9**	3	8011.3	<1	7554.6	3
	0.2	12958.4	<1	10202.3	<1	11523.3	<1	10899.6	<1	9793.1	<1	**9503.8**	3	10259.2	<1	9637.6	3
	0.3	17238.0	<1	11805.9	<1	13037.9	<1	12784.2	<1	11191.6	<1	**10658.1**	3	11731.5	<1	11142.8	3
	0.4	21559.6	<1	13186.4	<1	14190.4	<1	13987.1	<1	12304.8	<1	12069.9	3	12902.4	<1	**12009.5**	3
	0.5	25900.2	<1	14424.3	<1	15141.8	<1	15372.7	<1	13252.8	<1	13001.1	2	13896.9	<1	**12875.3**	3
att532	0.1	54706.7	<1	42508.3	<1	51691.4	<1	45218.4	3	39271.3	<1	**33795.7**	66	43611.1	<1	33933.0	65
	0.2	99168.2	<1	56731.6	<1	67732.8	<1	58060.9	3	53879.0	<1	47245.8	64	59269.2	<1	**46913.2**	53
	0.3	143491.0	<1	67947.3	<1	77856.9	<1	68591.5	2	64473.3	<1	**56196.2**	60	69679.5	<1	57738.6	62
	0.4	187206.0	<1	77640.2	<1	85264.5	<1	80507.1	2	72778.1	<1	**65339.4**	60	77609.6	<1	66619.8	60
	0.5	230028.0	<1	86270.7	<1	91162.4	<1	83988.9	2	79606.1	<1	76768.1	47	84149.3	<1	**72540.6**	59
rat783	0.1	5844.9	<1	3521.5	<1	5038.8	<1	4628.9	9	4174.4	<1	**3339.5**	224	4424.9	<1	3561.6	219
	0.2	11380.1	<1	4978.1	<1	6571.1	<1	5921.0	9	5939.2	<1	**4816.7**	216	6027.2	<1	4985.6	205
	0.3	17005.6	<1	6123.0	<1	7577.0	<1	7294.5	8	7117.3	<1	6381.3	160	7092.5	<1	**6101.9**	206
	0.4	22648.4	<1	7103.9	<1	8360.0	<1	8227.8	8	8008.2	<1	**6782.0**	196	7903.5	<1	6973.7	187
	0.5	28283.8	<1	7978.2	<1	9010.7	<1	8789.8	8	8729.4	<1	**7736.5**	192	8572.2	<1	7795.2	134

Table 1 summarizes our evaluation of the construction heuristics. Generally the probabilistic versions generate better results than their deterministic counterparts

Table 2. Variable Neighborhood Descent/Search comparison.

Instances	pACS [7]		HPSO [11]		EACS [15]		FI+VND		FI+VNS			PFI+VNS	
p	obj*	t[s]	obj*	t[s]	$\overline{\text{obj}}$	t[s]	obj	t[s]	obj*	$\overline{\text{obj}}$	$\bar{t}[s]$	$\overline{\text{obj}}$	$\bar{t}[s]$
eil101													
0.1	199.7	102	200.0	2	213.8	<1	197.4	<1	197.3	**197.3**	<1	**197.3**	<1
0.2	286.7	102	284.9	2	288.8	8	285.3	<1	283.6	**283.6**	8	**283.6**	13
0.3	353.5	102	**283.7**	2	358.7	5	352.0	<1	349.2	349.7	5	350.8	4
0.4	410.9	102	405.4	2	413.1	12	409.2	<1	404.7	405.6	12	**404.7**	1
0.5	470.7	102	455.7	2	462.9	14	459.9	<1	455.5	**456.2**	14	458.8	13
d198													
0.1	7556.1	1011	7504.9	5	8026.1	5	7438.2	1	7436.9	**7436.9**	5	**7436.9**	34
0.2	9489.2	1011	9415.1	5	9372.7	89	9357.8	2	9312.1	**9312.1**	89	9313.1	78
0.3	10951.9	1011	N/M		10635.4	74	10651.8	2	10531.3	**10538.9**	74	10540.3	52
0.4	12047.9	1011	N/M		11621.3	126	11665.0	2	11538.7	11586.6	126	**11555.7**	95
0.5	12745.5	1011	12527.6	5	12556.1	90	12617.9	1	12426.5	**12489.1**	90	12507.9	11
att532													
0.1	35179.7	2830	N/M		**33663.2**	3600	34533.5	50	33665.0	33685.8	2263	33683.0	2426
0.2	47531.4	2830	N/M		**44653.4**	3600	45867.9	47	45011.0	45179.0	2184	45148.6	2426
0.3	55865.3	2830	N/M		54008.2	2330	56150.4	49	53943.9	54111.1	2330	**53846.0**	2567
0.4	63308.0	2830	N/M		61455.6	1790	63973.5	38	61145.7	61500.9	1790	**61175.8**	1777
0.5	69671.2	2830	N/M		**67538.2**	1954	70431.3	33	67600.9	68285.0	1954	68298.6	3000
rat783													
0.1	3368.9	6131	3616.4	70	**3235.6**	3600	3292.4	281	3243.1	3259.7	4752	3255.1	10043
0.2	4781.2	6131	4775.1	63	**4534.0**	3600	4694.4	197	4583.3	4595.5	4627	4590.4	8712
0.3	5794.0	6131	N/M		5591.1	4537	5770.1	169	5574.0	5596.2	4537	**5579.4**	6712
0.4	6643.6	6131	N/M		**6336.3**	4119	6611.1	153	6369.7	6402.7	4119	6352.3	5303
0.5	7334.1	6131	7094.9	64	**6941.2**	4130	7282.4	137	7022.2	7073.5	4130	7007.5	4944

N/M . . . not mentioned

but at the price of much higher runtimes. From the deterministic ones, FI performs best, but also SFC performs well. Overall, PFI performs best but PNI is close.

Table 2 shows our Variable Neighborhood Search results compared to results from the literature. Weyland et al. [15] only published results of their EACS for instances att532 and rat783 with $p = 0.1$ and $p = 0.2$, and therefore, we performed additional tests using their code and published parameters for the other instances. For the VNS we apply FI and PFI as construction heuristic because FI offers the best tradeoff between efficiency and runtime and PFI performs best. Therefore we only include FI+VND, FI+VNS, and PFI+VNS in our results. In many cases our VND yields good solutions in short time, but VND alone cannot keep up with Weyland et al.'s EACS [15] or Marinakis' HybPSO [11]. However, our VNS is able to achieve new best solutions in 11 cases. We further observe that EACS performs excellent especially on the larger instances but the VNS is typically better on smaller instances. When comparing FI and PFI in combination with the VNS we see that they find solutions of similar quality but for larger instances the runtime of PFI+VNS increases more than the runtime of FI+VNS.

5 Conclusions and Future Work

In this paper the PTSP was discussed and the most important properties were shown. We focused on five TSP construction heuristics to generate an initial solution, namely Radial Sorting, Farthest Insertion, Nearest Insertion, Space Filling Curve, Nearest Neighbor, and compared them on several TSPLIB instances. Overall, Farthest Insertion performed best, followed by Space Filling Curve. Then we derived new construction heuristics to take probabilities into account: Almost Nearest Neighbor, Probabilistic Nearest Insertion and Probabilistic Farthest Insertion. They all significantly improve on their deterministic counterparts but are much more time-consuming. To improve these solutions we introduced a VNS framework with embedded VND based on 1-shift and 2-opt neighborhood structures and exact delta evaluation. Results show that FI+VNS and PFI+VNS typically yield the best solutions out of our tested configurations and they are even able to find new best-known solutions for 11 instances.

The major reason why our VNS performs so well is the usage of efficient, exact delta evaluation approaches for 1-shift and 2-opt originally proposed by Bianchi et al. [8]. Future work should consider further neighborhood structures for the PTSP for which exact delta evaluations are possible. In particular we are currently considering Or-opt.

Acknowledgments The authors thank Dennis Weyland for providing the source code of his EACS for better comparison.

References

1. Balaprakash, P., Birattari, M., Stützle, T., Dorigo, M.: Adaptive sample size and importance sampling in estimation-based local search for the probabilistic traveling salesman problem. European Journal of Operational Research 199(1), 98 – 110 (2009)
2. Balaprakash, P., Birattari, M., Stützle, T., Yuan, Z., Dorigo, M.: Estimation-based ant colony optimization and local search for the probabilistic traveling salesman problem. Swarm Intelligence 3(3), 223–242 (2009)
3. Bartholdi III, J.J., Platzman, L.K.: An $O(n\ log\ n)$ planar travelling salesman heuristic based on spacefilling curves. Operations Research Letters 1(4), 121–125 (1982)
4. Bertsimas, D., Howell, L.H.: Further results on the probabilistic traveling salesman problem. European Journal of Operational Research 65(1), 68–95 (1993)
5. Bertsimas, D.J., Chervi, P., Peterson, M.: Computational approaches to stochastic vehicle routing problems. Transportation Science 29(4), 342–352 (1995)
6. Bertsimas, D.J., Jaillet, P., Odoni, A.R.: A priori optimization. Oper. Res. 38(6), 1019–1033 (1991)
7. Bianchi, L., Gambardella, L., Dorigo, M.: An ant colony optimization approach to the probabilistic traveling salesman problem. In: Guervós, J., Adamidis, P., Beyer, H.G., Schwefel, H.P., Fernández-Villacañas, J.L. (eds.) Parallel Problem Solving from Nature — PPSN VII, Lecture Notes in Computer Science, vol. 2439, pp. 883–892. Springer Berlin Heidelberg (2002)
8. Bianchi, L., Knowles, J., Bowler, N.: Local search for the probabilistic traveling salesman problem: Correction to the 2-p-opt and 1-shift algorithms. European Journal of Operational Research 162(1), 206 – 219 (2005)

9. Chervi, P.: A Computational Approach to Probabilistic Vehicle Routing Problems. Master's thesis, Massachusetts Institute of Technology, Department of Electrical Engineering and Computer Science (1988)
10. Jaillet, P.: Probabilistic Traveling Salesman Problems. Ph.D. thesis, Massachusetts Institute of Technology (1985)
11. Marinakis, Y., Marinaki, M.: A hybrid multi-swarm particle swarm optimization algorithm for the probabilistic traveling salesman problem. Computers & Operations Research 37(3), 432 – 442 (2010)
12. Mladenović, N., Hansen, P.: Variable neighborhood search. Comput. Oper. Res. 24(11), 1097–1100 (1997)
13. Reinelt, G.: TSPLIB. `http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/`, accessed: 2015-05-07
14. Weyland, D., Bianchi, L., Gambardella, L.: New approximation-based local search algorithms for the probabilistic traveling salesman problem. In: Moreno-Díaz, R., Pichler, F., Quesada-Arencibia, A. (eds.) Computer Aided Systems Theory - EUROCAST 2009, Lecture Notes in Computer Science, vol. 5717, pp. 681–688. Springer Berlin Heidelberg (2009)
15. Weyland, D., Montemanni, R., Gambardella, L.M.: An enhanced ant colony system for the probabilistic traveling salesman problem. In: Di Caro, G.A., Theraulaz, G. (eds.) Bio-Inspired Models of Network, Information, and Computing Systems, Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, vol. 134, pp. 237–249. Springer International Publishing (2014)