Euclidean Traveling Salesman Problem

Dominik Schultes

January 2004

1 Introduction

The *Traveling Salesman Problem (TSP)* is one of the most famous NP-complete problems. A weighted graph G with n vertices is given and we have to find a cycle of minimum cost that visits each of the vertices of G exactly once [Ski98].

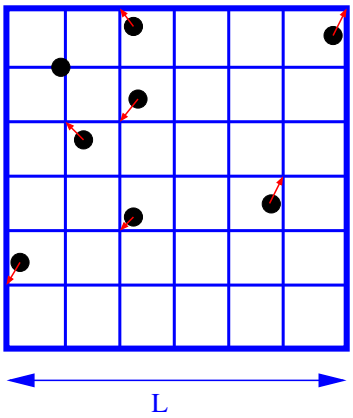
Since the computation of an exact solution is very expensive (under the assumption that $P \neq NP$), we are interested in approximate solutions – at least for special cases. Particularly, we look for an *Polynomial Time Approximation Scheme (PTAS)*, i.e., for any fixed error parameter $\varepsilon > 0$, the running time is bounded by a polynomial in n and the costs of the computed tour does not exceed $(1+\varepsilon)\cdot OPT$, where OPT stands for the costs of an optimal tour.

The *Metric TSP* is a special case of the Traveling Salesman Problem, where the edge costs satisfy the *triangle inequality*. There is a simple factor 2 algorithm, which bases on Minimum Spanning Trees and Eulerian tours [Vaz01] and runs in $O(m + n \log n)$ [Ski98]. In 1976 Christofides presented a factor 3/2 algorithm [Chr76], which runs in $O(n^3)$ [Ski98]. But, unfortunately, there is no PTAS for the Metric TSP [ALM+92].

The *Euclidean TSP* is a special case of the Metric TSP. For a fixed d, we consider n points in $\mathbb{R}^d$. The graph is complete and we use the Euclidean distance as cost function, i.e., $dist(x,y) = \left(\sum_{i=1}^{d}(x_i - y_i)^2\right)^{1/2}$. For the sake of simplicity, we concentrate on $d = 2$, i.e., we deal with n points in a plane. We now want to describe a PTAS for the Euclidean TSP [Vaz01] that bases on Dynamic Programming. In order to be able to apply Dynamic Programming, we first have to simplify the input and restrict the solution space.

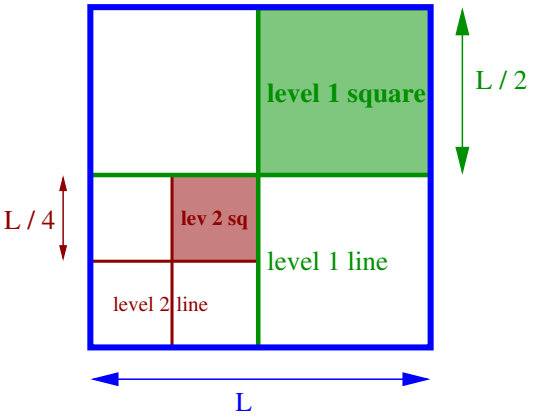
2 Basics

We define a *bounding box* as the smallest square that encloses all n points. We set the length of each edge of the square to $L = 4n^2$. This is possible by stretching all distances by an appropriate factor. Obviously, an optimal tour still stays an optimal tour (and vice versa). We assume w.l.o.g. that n is a power of two so that $L = 2^k$ is a power of two as well. Hence, $k = 2 + 2\log_2 n = O(\log_2 n)$. On the square, we define a unit grid.

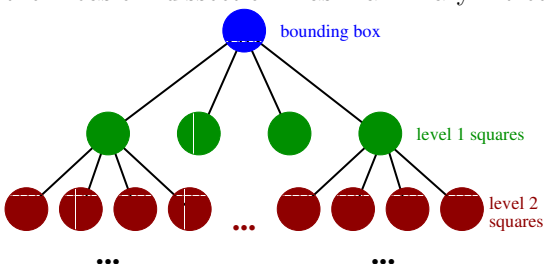


Furthermore, we relocate every node of G to the nearest gridpoint so that all nodes get integer coordinates. Due to the following facts, the effect of this perturbation is bounded. A lower bound for the length of the optimal tour OPT is $2L$ because at least two nodes are situated at opposite edges of the bounding box (otherwise, we would choose a smaller bounding box). The absolute error per node is bounded by $\sqrt{2}$ as the maximum distance from an arbitrary point to the nearest grid point is $1/\sqrt{2}$ and we cannot lose more than twice this distance. Hence, the total absolute error is bounded by $\sqrt{2}n$. Thus, we obtain a bound for the relative error $r \leq \frac{\sqrt{2}n}{OPT} \leq \frac{\sqrt{2}n}{2L} = \frac{\sqrt{2}n}{8n^2} = \frac{1}{4\sqrt{2}n}$. We can conclude that $\forall\varepsilon > 0\ \exists n_0\ \forall n \geq n_0 : r \leq \varepsilon$. Therefore, the relative error can be compensated by an appropriate adjustment of ε.

Now, we introduce a *basic dissection* of the bounding box: the bounding box is divided by two level 1 lines into four level 1 squares of size $L/2 \times L/2$. By four level 2 lines each level 1 square is divided into four level 2 squares of size $L/2^2 \times L/2^2$. This partition is continued recursively until unit squares are obtained.

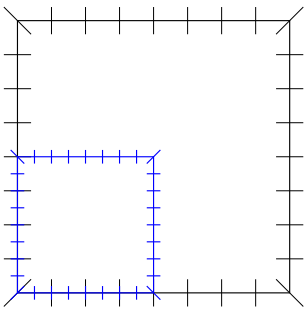


In general, a level i square has size $L/2^i \times L/2^i$. We can interpret the basic dissection as a 4-ary tree.



Obviously, the depth of the tree is k and the number of tree nodes amounts to $\frac{4^{k+1}-1}{4-1} = \frac{4^{(2+2\log_2 n)+1}-1}{3} = O(n^4)$.

The lines of the basic dissection may be crossed only at certain points, called *portals*. Each square has one portal for each corner and additional $m-1$ portals for each edge, i.e., each square has a total of $4m$ portals. The parameter m must lie in the interval $[k/\varepsilon, 2k/\varepsilon]$. Furthermore, m has to be a power of two. This guarantees that a portal of a level i square is a portal of a contained level $i+1$ square as well.



The distance between two portals of a level i square is $L/(2^i m)$.

As already said, we want a tour to cross the lines of the basic dissection only at portals. A tour with this property is called *well behaved*:

Definition 1 *A tour τ is* well behaved w.r.t. the basic dissection *if it is a tour on the n points and any subset of the portals.*

Our goal is to find an optimal tour τ of this type and show that $\|\tau\| \leq (1+\varepsilon)OPT$, where $\|\tau\|$ stands for the *length* of the tour τ. However, for the algorithm, it is necessary to introduce a further restriction on the tour:

Definition 2 *A tour τ that is well behaved w.r.t. the basic dissection* has limited crossings *if it visits each portal at most twice and is non-self-intersecting.*

We will only look for tours that are well behaved w.r.t. the basic dissection and have limited crossings. We can show that it is sufficient to look only for such tours.

Lemma 1 *Let the tour τ be well behaved w.r.t. the basic dissection. Then there is a tour τ' that is well behaved w.r.t. the basic dissection and has limited crossings so that $\|\tau'\| \leq \|\tau\|$.*

This lemma holds because of the fact that removing self-intersections by shortcutting cannot increase the tour length due to the triangle inequality. Hence, if we show that there is a well behaved tour τ that is short enough, i.e., $\|\tau\| \leq (1+\varepsilon)OPT$, we know due to Lemma 1 that there is a well behaved tour with limited crossings τ' that is short enough, too, because of $\|\tau'\| \leq \|\tau\|$. If we look for the shortest well behaved tour with limited crossings τ'', we can be sure that the tour that we find is short enough, too, because of $\|\tau''\| \leq \|\tau'\|$.

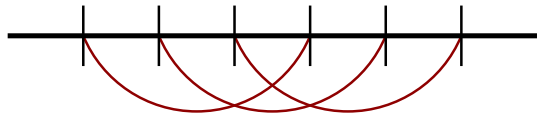
Therefore, we "only" have to

1. show that there is a well behaved tour that is short enough (Section 4) and
2. find an optimal well behaved tour with limited crossings (Section 3).

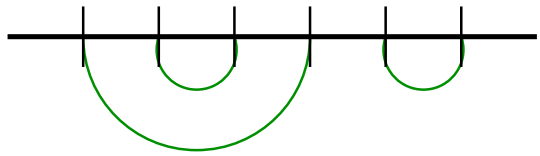
3 Dynamic Programming

Let us first deal with the visit of one square. As we look only for tours with limited crossings, each portal can be used at most

twice, i.e., it can be used 0, 1 or 2 times. Since a square has $4m$ portals, there are $3^{4m} = n^{O(1/\varepsilon)}$ possibilities. Furthermore, we claim that there is no self-intersection inside the square, i.e., "invalid pairings" are forbidden,



while "valid pairings" are allowed.



Each valid pairing corresponds to a balanced arrangement of parentheses; thus, the number of valid pairings is equal to the r-th Catalan number $C(r)$ when $2r$ portals are used. $C(r)$ can be bounded by 2^{2r} (as there are $2r$ parentheses and 2 types of parentheses (left and right)). Furthermore, we know that each of the $4m$ portals is visited at most twice, hence, $2r \leq 8m$. This leads to $C(r) \leq 2^{2r} \leq 2^{8m} = n^{O(1/\varepsilon)}$. Therefore, the number of valid pairings is bounded by $n^{O(1/\varepsilon)}$.

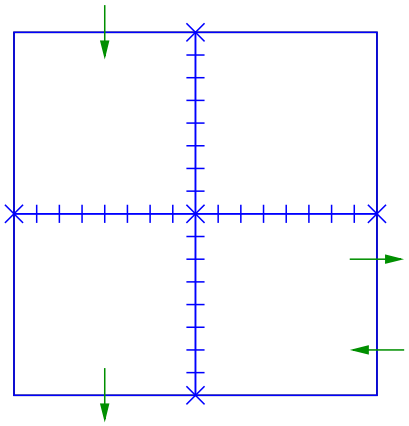
Now, we can combine both results: there are $n^{O(1/\varepsilon)}$ possibilities of portal usage and, for each portal usage, there are $n^{O(1/\varepsilon)}$ possibilities of valid pairings. This leads to a total of $n^{O(1/\varepsilon)} \cdot n^{O(1/\varepsilon)} = n^{O(1/\varepsilon)}$ possibilities. Of course, only the possibilities that cover all points inside the square are useful.

The algorithm that we want to present fills in the following 2-dimensional Dynamic Programming table:

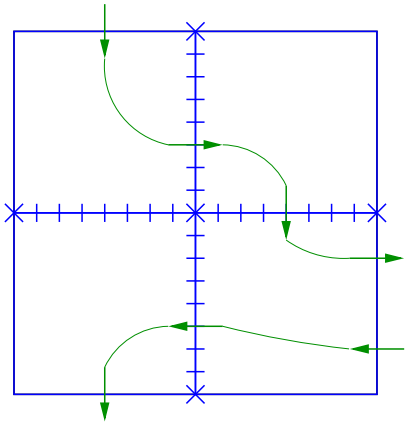
minimum costs	level 0	level 1			...
	square 1	sq. 1	...	sq. 4	...
valid visit 1					
valid visit 2					
valid visit 3					
.					
.					
.					

The number of columns is equal to the number of squares, which is equal to the number of nodes in the 4-ary tree, which is in $O(n^4)$. The number of rows is equal to the number of valid visits, which is bounded by $n^{O(1/\varepsilon)}$. Hence, the table size amounts to $O(n^4) \cdot n^{O(1/\varepsilon)} = n^{O(1/\varepsilon)}$.

The table is filled using a bottom-up approach, i.e., we start at the leaves of the 4-ary tree and use the results of the four children squares to compute the visits of the corresponding parent square. Let us have a look at one valid visit of a parent square, specified by the usage of the portals, which is represented by the green arrows in the given example.



Such a valid visit corresponds to one entry in the Dynamic Programming table. Our goal is the computation of the optimal length. Internally, there are less than $4m + 1$ portals. This leads again to $3^{4m+1} = n^{O(1/\varepsilon)}$ possibilities. Furthermore, we consider all valid pairings that are consistent with the external portal usage. Again, using Catalan numbers, the number of these valid pairings is bounded by $n^{O(1/\varepsilon)}$. Thus, we have to regard a total of $n^{O(1/\varepsilon)} \cdot n^{O(1/\varepsilon)} = n^{O(1/\varepsilon)}$ possibilities. Each possibility is composed by the appropriate valid visits of the four children squares.

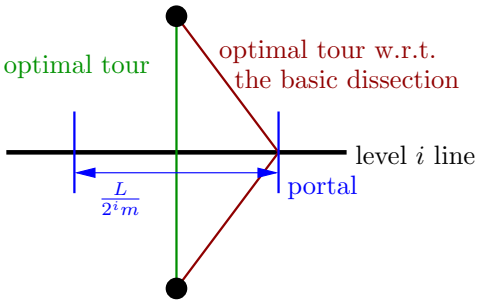


Since we have already computed the children squares, we can just sum up the optimal length of the appropriate visits of the children squares. We compute the optimal length for each of the $n^{O(1/\varepsilon)}$ possibilities and determine the minimum length of all possibilities.

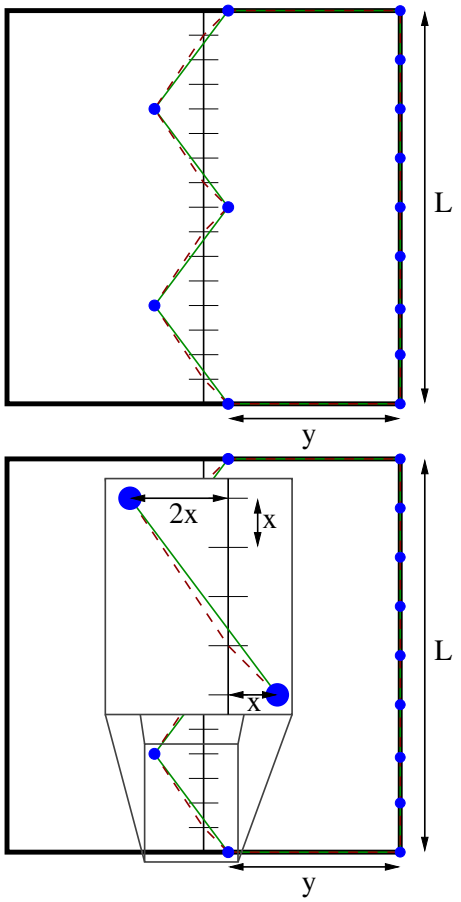
The total expense of this algorithm can be bounded by "table size $\times$ expense per entry" $= n^{O(1/\varepsilon)} \cdot n^{O(1/\varepsilon)} = n^{O(1/\varepsilon)}$.

4 Losses

The arbitrary crossing of lines is forbidden. This makes the Dynamic Programming approach possible, but it leads to an increase of the tour length.



The indirection is bounded by $L/(2^i m)$. We can notice that the indirection depends on the level of the line, i.e., the lower the level the greater the indirection. Due to this fact, we can construct an example where the indirection of an optimal tour that is well behaved w.r.t. the basic dissection exceeds the given error bound, namely an example where a level 1 line is crossed multiple times.

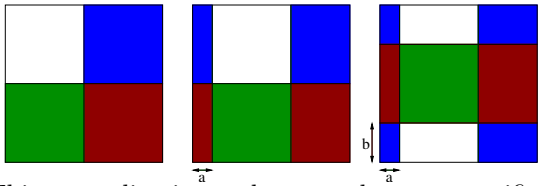


In this example the length of the optimal tour (green line) is $OPT = \frac{\sqrt{3^2+4^2}}{4}L + 2y + L$ and the length of the optimal tour w.r.t. the basic dissection (red line) is $A = \frac{\sqrt{2}+\sqrt{2^2+3^2}}{4}L+2y+L$. Since $y \leq \frac{L}{2}$, we obtain

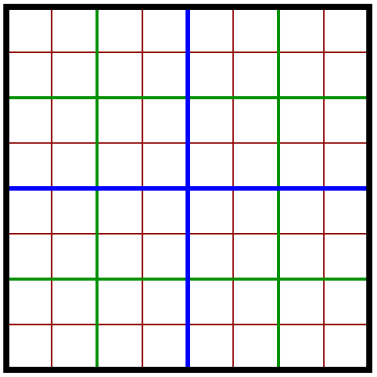
$$\frac{A}{OPT} \geq \frac{1.2549L + 2\frac{L}{2} + L}{1.25L + 2\frac{L}{2} + L} = 1.0015$$

Hence, the selection of an error parameter $\varepsilon < 0.0015$ fails.

In order to bypass this problem, we randomize the algorithm and generalize the concept of the dissection: we select integers a, b, $0 \leq a, b < L$ at random, move each vertical line from x to $(x + a)$ mod L and each horizontal line from y to $(y + b)$ mod L. We obtain the (a, b)-*shifted dissection*.



This generalization makes sure that any specific line has a random level. For example, the level of the middle line that we used for our counter-example for the basic dissection is not necessarily 1, but depends on the randomly chosen parameter a. As the indirection depends on the level of the line, we are interested in the probability that a randomly chosen line has a certain level in order to compute the expected value of the indirection. Therefore, let us have a look on the distribution of the levels of the lines.



There are 2^1 lines on level 1 (blue), 2^2 lines on level 2 (green), 2^3 lines on level 3 (red), and so on. Finally, there are 2^k lines on level k. Altogether, there are $2^{k+1} - 2$ lines. Therefore, the probability that a randomly chosen line has

level i is

$$p(i) = \frac{2^i}{2^{k+1} - 2} = \frac{2^i}{2L - 2} \leq \frac{2^i}{L}$$

As we already know, the maximum indirection when a level i line is crossed is

$$x(i) = \frac{L}{2^i m}$$

The expected value of the indirection X when one randomly chosen line is crossed is

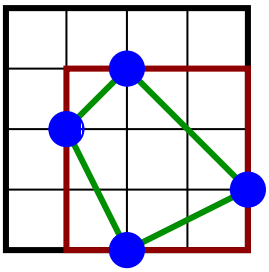
$$E(X) = \sum_{i=1}^{k} \left(\frac{L}{2^i m} \cdot \frac{2^i}{L} \right) = \frac{k}{m} \overset{(*)}{\leq} \varepsilon$$

$\left[(*) \text{ because of } m \geq \frac{k}{\varepsilon}\right]$

In order to be able to compute the expected value of the total indirection considering all line crossings, we have to bound the number of crossings. Let π be an optimal tour, and $N(\pi)$ be the total number of times π crosses horizontal and vertical grid lines. Then we have

$$N(\pi) \leq \sqrt{2} \cdot OPT$$

Proof:



$$N(\pi) = \|\pi\|_1 \leq \sqrt{2} \cdot \|\pi\|_2 = \sqrt{2} \cdot OPT$$

$\|\pi\|_1$ stands for the ℓ_1 norm (red line), $\|\pi\|_2$ for the Euclidean norm (green line). Obviously, the number of crossings of horizontal and vertical grid lines is equal to the ℓ_1 norm of the tour because we use a unit grid. The ℓ_1 norm and the Euclidean norm differ by a factor of at most $\sqrt{2}$. The length of the optimal tour OPT is measured in the Euclidean norm. □

Hence, the expected value of the total indirection Y is

$$E(Y) = N(\pi) \cdot E(X) \leq N(\pi) \cdot \varepsilon \leq \sqrt{2}\varepsilon \cdot OPT$$

Now we can use Markov's Inequality, which is in general

$$P(Y \geq a) \leq \frac{E(Y)}{a}$$

Here we select $a := 2\sqrt{2}\varepsilon \cdot OPT$ and obtain

$$P(Y \geq 2\sqrt{2}\varepsilon \cdot OPT) \leq \frac{\sqrt{2}\varepsilon \cdot OPT}{2\sqrt{2}\varepsilon \cdot OPT} = \frac{1}{2}$$

That means that the probability that the error bound $2\sqrt{2}\varepsilon$ is exceeded is less than or equal to $1/2$. Actually, we want an error parameter ε and not $2\sqrt{2}\varepsilon$, but this is no problem as we can choose a parameter $\varepsilon' > 0$ such that $2\sqrt{2}\varepsilon' = \varepsilon$. Of course, we can reduce the probability that the indirection is too long to $(1/2)^c$ by repeating the procedure for c different (a, b)-shifted dissections and choosing the shortest tour. Furthermore, we can derandomize the algorithm: we just try all $L^2 = O(n^4)$ dissections and, again, choose the shortest tour.

References

[ALM+92] S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy. Proof verification and hardness of approximation problems. In *IEEE Symposium on Foundations of Computer Science*, pages 14–23, 1992.

[Aro97] S. Arora. Nearly linear time approximation scheme for euclidean tsp and other geometric problems. In *38th IEEE Annual Symposium on Foundations of Computer Science*, pages 554–563, 1997.

[Chr76] N. Christofides. Worst-case analysis of a new heuristic for the traveling salesman problem. Technical report, Graduate School of Industrial Administration, Carnegie-Mellon University, Pittsburgh PA, 1976.

[Ski98] Steven S. Skiena. *The Algorithm Design Manual*. Springer, 1998.

[Vaz01] V. V. Vazirani. *Approximation Algorithms*. Springer, 2001.