Master Thesis – Applied Mathematics

Average-Case Analysis of the 2-opt Heuristic for the TSP

Jaap J. A. Slootbeek

Supervisor:

Dr. B. Manthey

University of Twente
Faculty of Electrical Engineering, Mathematics and Computer Science
Chair of Discrete Mathematics and Mathematical Programming

Abstract

The traveling salesman problem is a well-known $\mathcal{NP}$-hard optimization problem. Approximation algorithms help us to solve larger instances. One such approximation algorithm is the 2-opt heuristic, a local search algorithm. We prove upper bounds on the expected approximation ratio for the 2-opt heuristic for the traveling salesman problem for three cases. When the distances are randomly and independently drawn from the uniform distribution on $[0, 1]$ we show that the expected approximation ratio is $O(\sqrt{n \log(n)})$. For instances where the distances are 1 with probability p and 2 otherwise, we prove a bound for *fixed* p strictly between 0 and 1 that gives an upper bound on the number of heavy edges in the worst local optimum solution with respect to the 2-opt heuristic of $O(\log(n))$. For instances where the edges distances are randomly and independently drawn from the exponential distribution, we can upper bound the expected approximation ratio by $O\left(\sqrt{n \log^3(n)}\right)$.

Table of Contents

1 Introduction **4**

1.1 TSP and the 2-opt heuristic 4

1.2 Outline of this thesis 6

2 Related work **6**

3 Generic on $[0, 1]$ **8**

4 Discrete distribution **13**

5 Exponential distribution **16**

6 Simulations **22**

6.1 Introduction 22

6.2 TSP Model 22

6.3 WLO Model 24

6.4 Uniform on $[0, 1]$ 26

6.5 Discrete distribution 28

6.6 Exponential distribution 29

7 Discussion **32**

8 Conclusions **33**

Bibliography **35**

List of Theorems **36**

A Graphical results for the discrete case **37**

B Simulation data **42**

1 Introduction

In this section we discuss what the Traveling Salesman Problem is, how it can be solved and what approach we working on in this thesis. We also discuss the structure of this thesis

1.1 TSP and the 2-opt heuristic

Imagine you would want to take a tour to all capitals of mainland Europe. Our tour should both start and end in Amsterdam but apart from that the order is free. You find the cities in figure 1 As you are probably already thinking of some order to visit the cities in, you might be implicitly trying to minimize the distance traveled in total. After all, going from Amsterdam to Athens to Brussels to Budapest does not seem to make much sense as part of our route. This idea of minimizing the total distance covered (or for that matter, total cost) leads us to the Traveling Salesman Problem (TSP). The problem is defined as follows:

Definition 1.1 (Traveling Salesman Problem (TSP)) *An instance of TSP is a complete graph with distances for all edges. A solution is a Hamilton cycle. That is, a solution is a cycle visiting all nodes exactly once. The goal is to minimize the sum of the edge distances on such a cycle.*

In the remainder of this thesis we denote the number of nodes in an instance by n. From this we get some requirements. Firstly, the route has to be circular, that is, it has to end where it started. Secondly, it has to visit each city exactly once. Most importantly however, there has to be no other route that meets these requirements but that is shorter.

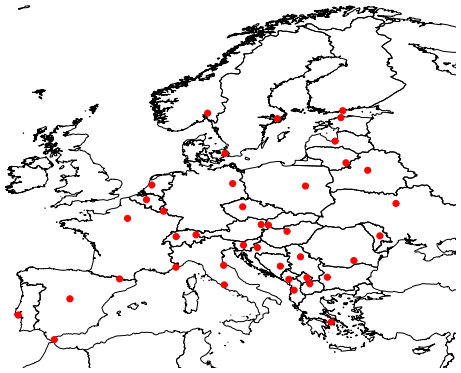

Figure 1: European capitals that should be included in the tour

A naive method for finding a solution is merely trying every possible route. If we want to visit n cities and we know where we want to start there are about $(n-1)!/2$ different routes possible. While this may work for small instances, for larger instances this leads to issues such as extremely long processing times. For instance, for the 40 cities in figure 1 there are over 10^{46} possible routes, all starting and ending in Amsterdam. We would like to have a method that is a bit smarter than just trying everything and which hopefully is also faster.

To look at the complexity of the problem, we must consider the decision version of the TSP. Instead of asking for the cheapest route we ask to find a route that is cheaper than a value k if it exists. For completeness we define the problem as follows.

Definition 1.2 (Traveling Salesman Problem decision version (TSP-d)) *An instance of TSP-d is a complete graph with distances for all edges. Also given is a value K. A solution is a Hamilton cycle. The goal is whether to decide if there exists a solution where the sum of the edge distances on such a cycle is at most K.*

In 1972 Karp showed that the TSP-d is $\mathcal{NP}$-complete and TSP $\mathcal{NP}$-hard. This means that, assuming $\mathcal{P} \neq \mathcal{NP}$ we see that TSP cannot be solved in polynomial time. It appears that we need to limit our expectations for the algorithms we can find. Nevertheless, the question whether $\mathcal{P} = \mathcal{NP}$ or not is still an open question.

Over time several methods that solve the TSP have been developed. The brute force method we saw before runs has a running time in $O(n!)$, since the work on a given route is done in polynomial time. Apart from the brute force method one of the options is dynamic programming. An algorithm by Held and Karp (1962) uses dynamic programming to solve the TSP. It also uses the property that every subpath of a path of minimum distance is itself also of minimum distance. The running time for this algorithm is in $O(n^2 2^n))$. Other options include the successful branch-and-cut approach, as used in Concorde (Applegate et al., 2011). Using the branch-and-cut approach with multiple separation routines and column generation Concorde is able to solve large instances of the TSP. Concorde was able to solve an instance with 85900 nodes to optimality (Applegate et al., 2009).

The previous algorithms all find the optimal solution. In some cases we accept a near-optimal solution instead of a fully optimal solution if the near-optimal solution is found in less time. In this case we use heuristics. These aim to find good (enough) solutions within reasonable time. We categorize these heuristics into two main categories, constructive heuristics and iterative improvement heuristics. Constructive heuristics aim to build a tour from scratch. An example of such a heuristic is the nearest neighbour heuristic (Kizilateş and Nuriyeva, 2013). At every point the salesman selects the closest other node that has not yet been visited until a full tour is found. The algorithm we analyze in this paper falls into the iterative improvement category.

One of the more common ways to achieve iterative improvement by using local search (Aarts and Lenstra, 2003). In local search, each solution has a neighbourhood and in each iteration we select the best solution in the neighbourhood of the current solution. More formally, each optimization problem instance is a pair (S, f) where S contains all solutions and $f : S \to \mathbb{R}$ gives each solution a value. If we consider a minimization problem, like the TSP, we want to find the solution $s^\star \in S$ with the lowest $f(s^\star)$. Let $N(s)$ be the neighbourhood of s. In each step of the local search we look for the best solution in the neighbourhood of the current solution. More formally, if s_c is our current solution we want to find $\tilde{s}$ such that for all s' in $N(s_c)$ we have $f(\tilde{s}) \leq f(s')$. If the best solution we find is the current solution then we have found a local optimum. This locally optimal solution does *not* have to be the global optimal solution, that is, the best solution overall. Local search is a broad technique that has been applied to several fields. We tailor it to our specific problem by defining the neighbourhoods. If needed we can also define how the initial solution is created.

The 2-opt heuristic is a local search algorithm for solving the traveling salesman problem. We see the main idea of the 2-opt heuristic algorithm in figure 2. The idea is that where a route crosses over itself we reorder the nodes so the crossing is removed. In the local search algorithm we try for all pairs of edges if the crossed over version is shorter than the original. That is, the

neighbourhood of a solution s is all solutions where two edges have been removed and replaced by two other edges to create a feasible Hamilton cycle. This idea was introduced by Croes (1958). The algorithm is shown as Algorithm 1.1. Note that this algorithm only finds local optima under the 2-opt neighbourhood. This means that we cannot perform another 2-opt step on the route to find a better route. However, this does not guarantee an optimal solution for the TSP.

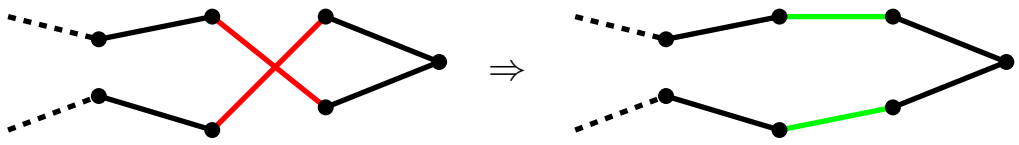

Figure 2: Main idea of 2-opt

Algorithm 1.1 2-OPT operation

Input: A complete graph with distances defined on the edges and a route and its distance
Output: A 2-opt optimal route

repeat
  for $i \in$ Nodes eligible to be swapped **do**
    for $j \in$ Nodes eligible to be swapped such that $j > i$ **do**
      Apply 2-opt swap to i and j: create the new route as follows:
      Take route upto i and add in order
      Take route from j to i (both including) and add in reverse order
      Take the route after k and add in order
      Calculate new distance
      if new distance $<$ distance **then**
        Update route to include new ordering
      end if
    end for
  end for
until no improvement is made

1.2 Outline of this thesis

The remainder of this thesis is in broad terms divided into two parts. In the first part (Sections 3 upto 5) we present the theoretical results. We analyse three different classes of instances, firstly, instances where the edge distances are drawn from a slightly generalized version of the uniform distribution on $[0, 1]$ in Section 3. Secondly, we consider in Section 4 instances where the edges have a distance of either 1 or 2, this means we consider a discrete distribution. As third case we consider instances where the edge distances are drawn from the exponential distribution in Section 5. The second part contains the simulation results. In Section 6 we compare the theoretical results to experimental observations. After that we will discuss our results and findings in Section 7 and present our conclusions in Section 8.

2 Related work

Before starting to describe our own work in the field of the 2-opt approximation ratio we would like to first see what others have done before us. In this section we present results from literature. From the knowledge we gain from this literature research we can better place our results in the

field. We discuss several results related to the average-case approximation results we present as well as to other approximation results.

This thesis expands on a paper by Engels and Manthey (2009). They proved that for instances with n nodes and edge weights drawn uniformly from $[0, 1]$ and independently the expected approximation ratio is $O(\sqrt{n} \cdot (\log(n)^{3/2})$. For this case we prove a better upper bound in the next section.

Chandra et al. (1999) have proved that the worst-case performance ratio for TSP instances where the triangle inequality holds is at most $4\sqrt{n}$ for all n and at least $\frac{1}{4}\sqrt{n}$ for infinitely many n. On the other hand Grover (1992) proved that for *any* symmetric TSP instance any 2-optimal route has a length that is at most as high as the average length of all tours.

We can prove results in the forms of worst-case approximation ratios, which can be too pessimistic due to the very very specific features in the instances and the average-case approximation ratios, which can be dominated by completely random instances that do not represent the real-life instances well. Smoothed analysis Spielman and Teng (2004) forms a hybrid of both these cases. We let an adversary specify an instance. Instead of using that instance directly (which is the case in the worst-case analysis) we apply a slight (random) perturbation to it. If we take the expected value of this random perturbation we get from the smoothed performance the expected performance. The idea behind this approach is that in practice instances are usually subject to a small amount of noise. When we look at the title of the paper by Spielman and Teng (2004), "Smoothed Analysis of Algorithms: Why the Simplex Algorithm Usually Takes Polynomial Time" we get one of the results in this paper. It can be very confusing to observe a good performance but having a quite bad theoretical bound. As the theoretical bound is often due to some unlikely or unrealistic instances, using this methods should allow for more realistic conclusions. This method has also been employed for the 2-opt heuristic.

An important measure when using the 2-opt heuristic is how many steps are needed to reach a 2-opt optimal solution. Englert et al. (2014) showed that expected length of any 2-opt improvement path is $\tilde{O}(n^{4+1/3} \cdot \phi^{8/3})$. For intuition you can think of ϕ to be proportional to $1/\sigma^d$, being a perturbation parameter. In other cases, such as when the initial tour is constructed by an insertion heuristic, this upper bound can be improved further. Manthey and Veenstra (2013) look at the Euclidean TSP which means that cities are placed on $[0, 1]^d$. The distances then depend on the locations of the points: we can for example use the Euclidean or the Manhattan norms to determine the distances. The instances are perturbed by independent Gaussian distributions with mean 0 and standard deviation σ. For d-dimensional instances of n points the bound on the number of steps needed is $O(\sqrt{d}n^4 D_{\max}^4 \sigma^{-4})$ for the Euclidean norm and $O(d^2 n^4 D_{\max} \sigma^{-1})$ for the Manhattan norm. In this $D_{\max}$ such that $x \in [-D_{\max}; D_{\max}]^d$ with a probability of at least $1 - 1/n!$ for all points x.

Englert et al. (2014) also showed an upper bound on the expected approximation ratio of the Euclidean TSP with respect to all L_p metrics of $O(\sqrt[d]{\phi})$. Künnemann and Manthey (2015) also worked on the approximation ratio. They were able to prove that for instances of n points in $[0, 1]^d$ perturbed by Gaussians of standard deviation $\sigma \leq 1$ the approximation ratio is in $O(\log(1/\sigma))$.

Next to starting from a random initial tour for the 2-opt heuristic we can also start with a construction heuristic such as the spanning tree heuristic. That guarantees an approximation ratio of 2 even before we start using 2-opt. Lastly we want to highlight the real life performance of the 2-opt heuristic. Despite it's relative simplicity it will, for Euclidean instances, usually get within 5% of the optimal value, even with a large number of nodes (Johnson and McGeoch, 1997). The results for random distance matrices are not as good as for Euclidean instances it is still superior to Christofides' algorithm, the best of the tested construction heuristics by Johnson and McGeoch (1997).

3 Generic on $[0, 1]$

The goal of this section is to find a result for instances where the distances are drawn from a standard uniform distribution. However, as the extension of this proof to a more general case was found to give the same result, we present this generalisation. We find a result gives a bound on the growth of $\mathbb{E}(\text{WLO}_n/\text{OPT}_n)$ where WLO_n is the worst local optimum that can be attained by the 2-opt operation and OPT_n is the optimal TSP solution

Assume we have a generic distribution G which can take only values on $[0, 1]$ with a probability density function $f(x)$ and a cumulative distribution function $F(x)$ defined. We have the following properties:

- $F(0) = 0$, by definition
- $F(1) = 1$, by definition
- $F(x)$ is non-decreasing, by definition
- $F(x) \geq x$
- $f(x)$ is non-increasing

Examples of this include $F(x) = x$ (uniform on $[0, 1]$) and $F(x) = \sqrt{x}$. Before we can start looking at the specifics for this set of instances we first need to look at how we are going to count edges.

Lemma 3.1 *There exist at least $\frac{mn}{64}$ pairs of edges where at least one edge is* heavy *and where the edges that are inserted by performing a 2-opt operation on these edges are disjoint.*

Proof. We call an edge *heavy* if it has a weight greater than η. We want to find a lower bound on the amount of pairs of edges where at least one edge is heavy and where the edges that will be inserted by performing the 2-opt operation (on those edges) are disjoint. If a combination of two edges have the latter property we call that combination *independent*. We want to colour the edges such that any pair of edges with colour *blue* forms to an independent combination. We colour the edges *red* and *blue* such that there are no two adjacent blue edges. For an odd number of edges this leads to two adjacent red edges. This colouring has the property that when we use the blue edges the edges that will be inserted by performing the 2-opt operation (on those edges) are disjoint. Note that this colouring can be rotated over the edges in the cycle without losing its properties. However we choose the colouring that has the most heavy blue edges.

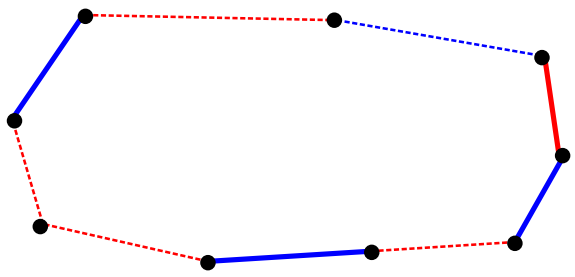

Figure 3: A possible colouring, continuous lines are heavy, dashed lines are not heavy.

For ever heavy edge look at a colouring in which that heavy edge is coloured blue. Each heavy blue edge can be paired with one of the $0.5 \cdot (n-3)$ other blue edges. In total we will have at least

$0.5 \cdot (m-1)$ heavy edges that are blue. We use the rough bound $mn/64$ for the total amount of possible independent combinations of a heavy edge and any other edge, taking into account that there some combinations could be counted double.

Consider a combination with blue edges e and e' with e heavy which is possible without loss of generality. When performing the 2-opt operation these edges are replaced by the edges f and f' as determined by the 2-opt operation. We know that if $w(e)+w(e') > w(f)+w(f')$ that H is not a locally optimal tour. Since the weights are non-negative we know that $w(e)+w(e') \geq \eta$. So, if $w(f)+w(f') < \eta$ we know that H is not optimal. We use this frequently in the remainder of this thesis. ■

We now have a lower bound on how many pairs we can find with properties we find desirable. Using this we are able to bound the probability that a cycle with some heavy edges is locally optimal under the 2-opt operation.

Lemma 3.2 *Let H be any fixed Hamiltonian cycle and let $\eta \in (0,1]$. Assume H contains at least $m \geq 4$ edges of weight at least η. The weights of the edges are random variables taken from G. Let the weights of the edges on the cycle be known. Then*

$$\mathbb{P}(H \text{ is locally optimal}) \leq e^{-\frac{F^2(\eta)mn}{64}}$$

Proof. Consider a pair of edges from lemma 3.1. This is a independent pair of edges e and e' with e heavy which is possible without loss of generality. When performing the 2-opt operation these edges are replaced by the edges f and f' as determined by the 2-opt operation. We know that if $w(e)+w(e') > w(f)+w(f')$ that H is not a locally optimal tour. Since the weights are non-negative we know that $w(e)+w(e') \geq \eta$. So, if $w(f)+w(f') < \eta$ we know that H is not optimal.

Now we determine $\mathbb{P}(w(f)+w(f') < \eta)$ for $\eta \in [0,1]$:

$$\begin{aligned}
\mathbb{P}(w(f)+w(f') < \eta) &= \int_0^\eta \mathbb{P}(w(f)+w(f') < \eta \mid w(f) = x)\, dx \\
&= \int_0^\eta \mathbb{P}(w(f') < \eta - x | w(f) = x) dx \\
&= \int_0^\eta F(\eta - x) f(x) dx \\
&\leq \int_0^\eta F(\eta) f(x) dx \\
&= F(\eta) \cdot \int_0^\eta f(x) dx \\
&= F(\eta) \cdot (F(\eta) - F(0)) \\
&= F^2(\eta)
\end{aligned}$$

where the inequality follows from the cumulative distribution function being non-decreasing.

So for a combination an improving 2-opt operations is possible with probability $F^2(\eta)$. In order for a tour H to be optimal, it cannot have any improving 2-opt operations, in particular not for the $mn/64$ independent combinations we found. That means that

$$\mathbb{P}(H \text{ is locally optimal}) \leq \left(1 - F^2(\eta)\right)^{\frac{mn}{64}} \leq e^{-\frac{F^2(\eta)mn}{64}} \tag{1}$$

using the inequality $1 - x \leq e^{-x}$ for $x = F^2(\eta)$. ■

Now we know how the probability of a tour being optimal can be estimated. Now we can look into the probability of the worst solution generated by 2-opt being over a certain weight. For this we use the following lemma.

Lemma 3.3 *For any $c > 8$ and a distribution G as above, we have*

$$\mathbb{P}\left(WLO_n \geq 6c\sqrt{n\log(n)}\right) \leq \exp\left(n\log(n)\left(1 - \frac{1}{64}c^2\right)\right)$$

Proof. Define $m_i = 3^{-i}n,\ \eta_i = 2^i\eta$ and $\eta = c \cdot \sqrt{\log(n)n^{-1}}$. We are going to look at a tour H which contains at most m_i edges of weight at least η_i. First, if $i \geq \log(n)$ we have $m_i < 4$ and $\eta_i > 1$. Because the weights are at most 1 it is sufficient to consider $i \in [0, \ldots, \log(n) - 1]$. If a tour H contains at most m_i edges of weight at least η_i then

$$w(H) \leq \sum_{i=0}^{\log(n)-1} m_i\eta_{i+1}.$$

We can see this as follows: for each i we count the number of edges with weight more that η_i, we count these with weight η_{i+1}. For some edges this may be too low but these will be counted again for a higher i. Hence we achieve an upper bound on the weight of the tour.

We have

$$m_i\eta_{i+1} = 2\left(\frac{2}{3}\right)^i \eta n = 2c\left(\frac{2}{3}\right)^i \sqrt{n\log(n)}.$$

Using this, we have

$$w(H) \leq \sum_{i=0}^{\log(n)-1} m_i\eta_{i+1} = \sum_{i=0}^{\log(n)-1} 2c\left(\frac{2}{3}\right)^i \sqrt{n\log(n)} = 2c\sqrt{n\log(n)} \sum_{i=0}^{\log(n)-1} \left(\frac{2}{3}\right)^i \leq 6c\sqrt{n\log(n)}$$

where the last inequality uses the geometric series

We now want to estimate the probability that the tour H which contains at most m_i edges of weight at least η_i. For the probability of this to happen, we refer back to Lemma 3.2 and first look at the case for a fixed i. Fix any tour H. The probability that H is locally optimal provided it contains at least m_i edges of weight at least η_i (call these conditions $\star_i$) is at most $\exp(-\frac{F^2(\eta_i)m_in}{64})$. Thus,

$$\begin{aligned}\mathbb{P}(H \text{ is optimal under } \star_i) &\leq \exp\left(-\frac{F^2(\eta_i)m_in}{64}\right) \\ &= \exp\left(-\frac{-F^2(2^i\eta)3^{-i}n^2}{64}\right)\end{aligned}$$

We use Boole's inequality to bound the probability that H is locally optimal from above, provided there exists an $i \in [0, \ldots, \log(n) - 1]$ for which H contains at least m_i edges of weight at least η_i. Again using Boole's inequality, we determine an upper bound to the probability that one of the $n!$ possible tours is locally optimal, provided that it contains at least m_i edges of weight η_i for some i. This probability is at most

$$n! \cdot \log(n) \cdot \exp\left(-\frac{-F^2(2^{\log(n)}\eta)3^{-\log(n)}n^2}{64}\right).$$

We work on this expression to find

$$\begin{aligned}\mathbb{P}\left(\mathrm{WLO}_n \geq 6c\sqrt{n\log(n)}\right) &\leq n! \cdot \log(n) \cdot \exp\left(-\frac{F^2(2^{\log(n)}\eta)3^{-\log(n)}n^2}{64}\right) \\ &\leq n^n \cdot \exp\left(-\frac{F^2(2^{\log(n)}\eta)n^2}{64 \cdot 3^{\log(n)}}\right) \\ &= \exp\left(n\log(n) - \frac{F^2(2^{\log(n)}\eta)n^2}{64 \cdot 3^{\log(n)}}\right) \\ &= \exp\left(n\left(\log(n) - \frac{F^2(2^{\log(n)}\eta)n}{64 \cdot 3^{\log(n)}}\right)\right)\end{aligned}$$

For this probability to go to 0 for large n, we need $64 \cdot 3^{\log(n)} \cdot \log(n) \leq F^2(2^{\log(n)}\eta)n$. This is true if $64 \cdot 3^{\log(n)} \cdot \frac{\log(n)}{n} \leq F^2(2^{\log(n)}\eta)$ or

$$8 \cdot \left(\sqrt{3}\right)^{\log(n)} \sqrt{\frac{\log(n)}{n}} \leq F(2^{\log(n)}\eta) = F\left(c \cdot 2^{\log(n)} \cdot \sqrt{\frac{\log(n)}{n}}\right)$$

We can see this is at least true if $F(x) \geq x$ for $x \in (0, 1]$ and $c > 8$. In this case, we can use the following bound:

$$\frac{F^2(2^{\log(n)}\eta)n}{64 \cdot 3^{\log(n)}} \geq \frac{4^{\log(n)} \cdot \eta^2 n}{64 \cdot 3^{\log(n)}} = \frac{1}{64}\left(\frac{4}{3}\right)^{\log(n)} c^2 \frac{\log(n)}{n} n \geq \frac{1}{64}c^2\log(n)$$

The result follows: For any $c > 2$ and a distribution G with $F(x) > x$ for $x \in (0, 1]$ we have

$$\mathbb{P}\left(\mathrm{WLO}_n \geq 6c\sqrt{n\log(n)}\right) \leq \exp\left(n\log(n)\left(1 - \frac{1}{64}c^2\right)\right). \tag{2}$$

We also note here that for $c > 8$ this probability is strictly decreasing in n and approaches zero at least exponentially fast as n increases.

■

We remark that, with our constraint on $F(x)$ and c, the probability that the worst local optimum worse than $6c\sqrt{n\log(n)}$ goes to zero quickly for high n. We also present the following lemma to bound the optimal solution of an instance.

Lemma 3.4 *For any $n \geq 2$ and $c \in [0, 1]$, we have $\mathbb{P}(OPT_n \leq c) \leq nF(c)c^{n-1}f(0)^{n-1}$*

Proof. First we look at $\mathbb{P}(w(H) \leq c)$. We claim that for a instance with n nodes we have

$$\mathbb{P}\left(w(H)_n \leq c\right) = \frac{F(c)c^{n-1}f(0)^{n-1}}{(n-1)!} \tag{3}$$

We prove this using induction. For $n = 1$ we get

$$\mathbb{P}(w(H)_1 \leq c) = F(c)$$

Now for the induction step. Assume (3) is true for $n \leq k-1$. Now we work on $\mathbb{P}(w(H)_k \leq c)$.

$$\begin{aligned}
\mathbb{P}(w(H)_k \leq c) &= \int_{x=0}^{c} \mathbb{P}(w(H)_{k-1} \leq c-x) f(c) dx \\
&\leq \int_{x=0}^{c} \mathbb{P}(w(H)_{k-1} \leq c-x) f(0) dx \\
&= f(0) \int_{x=0}^{c} \mathbb{P}(w(H)_{k-1} \leq c-x) dx \\
&\overset{\text{IH}}{=} f(0) \int_{x=0}^{c} \frac{F(c-x)(c-x)^{k-2} f(0)^{k-2}}{(k-2)!} dx \\
&\leq f(0)^{k-1} \int_{x=0}^{c} \frac{F(x)(x)^{k-2}}{(k-2)!} dx \\
&\leq f(0)^{k-1} F(c) \int_{x=0}^{c} \frac{(x)^{k-2}}{(k-2)!} dx \\
&= \frac{f(0)^{k-1} F(c) c^{k-1}}{(k-1)!}
\end{aligned}$$

Now we use Boole's inequality to bound the probability that there exists a tour with $w(H) \leq c$ to find the result. ■

Theorem 3.5 (Result for general distribution) *Fix a distribution G for the weights of the edges with $F_G(x) \geq x$ for $0 < x \leq 1$. We have*

$$\mathbb{E}\left[\frac{WLO_n}{OPT_n}\right] \in O\left(\sqrt{n \log(n)}\right)$$

Proof. Assume $\text{WLO}_n/\text{OPT}_n > 6c^2\sqrt{n\log(n)}$ for $c > 8$. Then $\text{WLO}_n \geq 6c\sqrt{n\log(n)}$ or $\text{OPT}_n \leq \frac{1}{c}$. The probability that $\text{WLO}_n \geq 6c\sqrt{n\log(n)}$ is given by Lemma 3.3. The probability that $\text{OPT}_n < \frac{1}{c}$ is less than $F(1/c)f(0)^{n-1}(1/c)^{1-n}n$. The probability of either of these events happening is at most $F(1/c)f(0)^{n-1}\left(\frac{1}{c}\right)^{n-1} n + \exp\left(n\log(n)\left(1-\frac{1}{64}c^2\right)\right)$ for $c > 8$. Then we know for all ξ such that $\xi > f(0)^4$ and $\xi > 64^2$ that

$$\frac{\text{WLO}_n}{\text{OPT}_n} \leq 6\sqrt{n\log(n)} \cdot \int_{\xi}^{\infty} cP(c)dc^2 + O(\sqrt{n\log(n)}) \tag{4}$$

We perform the substitution $x = c^2$ and find

$$\frac{\text{WLO}_n}{\text{OPT}_n} \leq 3\sqrt{n\log(n)} \cdot \int_{\sqrt{\xi}}^{\infty} P(\sqrt{x})dx + O(\sqrt{n\log(n)}) \tag{5}$$

We calculate $\int_{x=\sqrt{\xi}}^{\infty} P(\sqrt{x})dx$ by splitting it into the two separate parts

$$\int_{x=\sqrt{\xi}}^{\infty} P(\sqrt{x})dx = \int_{x=\sqrt{\xi}}^{\infty} F(1/\sqrt{x})f(0)^{n-1}\left(\frac{1}{\sqrt{x}}\right)^{n-1} n dx + \int_{x=\sqrt{\xi}}^{\infty} \exp\left(n\log(n)\left(1-\frac{1}{64}x\right)\right) dx.$$

We work on these independently to find

$$\int_{x=\sqrt{\xi}}^{\infty} F(1/\sqrt{x}) f(0)^{n-1} \left(\frac{1}{\sqrt{x}}\right)^{n-1} n dx = \frac{2nF\left(\frac{1}{\sqrt{x}}\right)\xi^{\frac{3}{4}-\frac{n}{4}} f(0)^{n-1}}{n-3} \qquad \text{and}$$

$$\int_{x=\sqrt{\xi}}^{\infty} \exp\left(n\log(n)\left(1-\frac{1}{64}x\right)\right) dx = \frac{64n^{-\frac{n\sqrt{\xi}}{64}+n-1}}{\log(n)}$$

Combining these we find

$$\begin{aligned}\int_{x=\sqrt{\xi}}^{\infty} P(\sqrt{x}) dx &= \frac{2nF\left(\frac{1}{\sqrt{x}}\right)\xi^{\frac{3}{4}-\frac{n}{4}} f(0)^{n-1}}{n-3} + \frac{64n^{-\frac{n\sqrt{\xi}}{64}+n-1}}{\log(n)} \\ &= \frac{F(1/\sqrt{\xi})\xi^{\frac{3}{4}}}{f(0)} \cdot \left(\frac{f(0)}{\xi^{\frac{1}{4}}}\right)^{n} \cdot \frac{2n}{n-3} + \frac{64n^{-\frac{n\sqrt{\xi}}{64}+n-1}}{\log(n)}.\end{aligned}$$

This function goes to zero exponentially fast as n increases provided we indeed have that $\xi > 64^2$ and $\xi > f(0)^4$. We can say that $\int_{c=\xi}^{\infty} cP(c)\ dc^2 \in O(1)$. This combined with (4) leads to the following result

$$\mathbb{E}\left[\frac{\mathrm{WLO}_n}{\mathrm{OPT}_n}\right] \leq O\left(\sqrt{n\log(n)}\right) \cdot O(1) + O(\sqrt{n\log(n)}) \in O\left(\sqrt{n\log(n)}\right) \tag{6}$$

■

Corollary 3.6 (Uniform distributions) *Let the weight of the edges be drawn from the any uniform distribution on* $[0, \chi]$ *with* $0 < \chi \leq 1$*. We have*

$$\mathbb{E}\left[\frac{WLO_n}{OPT_n}\right] \in O\left(\sqrt{n\log(n)}\right)$$

Corollary 3.7 (Standard uniform distribution) *Let the weight of the edges be drawn from the* ***standard uniform*** *distribution, on* $[0, 1]$*. We have*

$$\mathbb{E}\left[\frac{WLO_n}{OPT_n}\right] \in O\left(\sqrt{n\log(n)}\right)$$

4 Discrete distribution

In this section we will consider graphs where the distances between points are one of two values. This is a discrete distribution. We find a result that bounds to growth of the number of heavy edges used in the Worst Local Optimum solution under the 2-opt operation. As there are only two choices for the heights, a heavy edge is an edge whose weight is the higher of the two. In this section we use the weights 1 and 2. Edges with weight 2 are therefor called *heavy*.

Consider a complete graph with n vertices where the weights of the edges are independently determined as follows:

$$w(e) = \begin{cases} 1 \text{ w.p. } p \\ 2 \text{ w.p. } 1-p \end{cases}$$

Lemma 4.1 *Let H be any fixed Hamiltonian cycle. Assume H contains at least m edges of weight 2. Let the weight for the edges on H be known. The weights for the other edges are iid with $w(e) = 1$ w.p. p and $w(e) = 2$ w.p. $1 - p$. Then*

$$\mathbb{P}(H \text{ is locally optimal}) \leq (1 - \xi(p))^{\frac{nm}{64}} \leq e^{-\frac{\xi(p)nm}{64}}$$

with $\xi(p) = 2p - 3p^2 + 2p^3$

Proof. By Lemma 3.1 there are at least $mn/64$ independent pairs with at least one heavy edge. Look at such a combination (e, e') with $w(e) = 2$. 2-opt will be able to improve the tour to include the edges (f, f') instead of (e, e') if $w(e) + w(e') > w(f) + w(f')$. Looking closer at this we see:

$$w(e) + w(e') > w(f) + w(f') \Rightarrow 2 + w(e') > w(f) + w(f') \Rightarrow w(f) + w(f') - w(e') < 2$$

Since we know the probabilities for the weights of the edge we can fill out the following table:

$w(f)$	$w(f')$	$w(e)$	$w(f) + w(f') - w(e)$	$w(f) + w(f') - w(e') < 2$	Probability
1	1	1	1	✓	p^3
1	1	2	0	✓	$p^2(1-p)$
1	2	1	2	×	
1	2	2	1	✓	$p(1-p)^2$
2	1	1	2	×	
2	1	2	1	✓	$p(1-p)^2$
2	2	1	3	×	
2	2	2	2	×	
				Total:	$p^3 + p^2(1-p) + 2p(1-p)^2$

$$p^3 + p^2(1-p) + 2p(1-p)^2 = p\left(2p^2 - 3p + 2\right)$$

This means that the probability that a tour H with m edges of weight two is is locally optimal is bounded from above by $\left(1 - 2p + 3p^2 - 2p^3\right)^{\frac{nm}{64}}$. Denote $2p - 3p^2 + 2p^3$ (see figure 4) as $\xi(p)$ to simplify notation. We then get

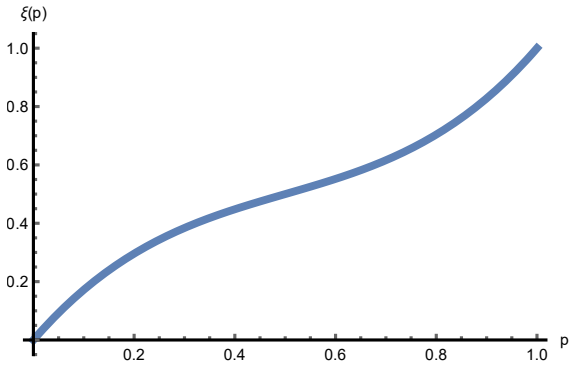

Figure 4: A plot of $\xi(p)$ for $0 \leq p \leq 1$

$$\mathbb{P}(H \text{ is locally optimal}) \leq (1 - \xi(p))^{\frac{nm}{64}} \leq e^{-\frac{\xi(p)nm}{64}} \tag{7}$$

using the inequality $1 - x \leq e^{-x}$ for $x = \xi(p)$. ■

Lemma 4.2 *For any m we have*

$$\mathbb{P}(WLO_n \geq n+m) \leq \exp\left(n\left(\log(n) - \frac{1}{4}m\xi(p)\right)\right)$$

Proof. We are going to look at a tour H which contains at most m edges of weight 2. We can bound the weight of such a tour from above as

$$w(H) \leq \sum_{e\in E} w(e) = n+m.$$

We refer back to Lemma 4.1 to find the probability of a tour of this weight being locally optimal. The probability that H is locally optimal, given that it contains at least m edges of weight 2 (denote this condition by $\Diamond$) is at most $\exp(-\frac{1}{64}\xi(p)nm)$.

We use Boole's inequality to bound the probability that one of the $n!$ possible tours is locally optimal from above, provided that it contains at least m edges of weight 2. We get

$$\mathbb{P}\left(H \text{ is locally optimal under } \Diamond\right) \leq n! \cdot \exp\left(-\frac{1}{64}\xi(p)nm\right).$$

By bounding $n!$ from above by $n^n = \exp(n\log(n))$ we find the result

$$\mathbb{P}(WLO_n \geq n+m) \leq \exp\left(n\left(\log(n) - \frac{1}{64}m\xi(p)\right)\right)$$

We note that this probability goes to 0 for large n if $\log(n) \leq \frac{1}{64}m\xi(p)$, from which we gather that that is the case if the number of heavy edges is larger than $\log(n)$.

■

Theorem 4.3 (Result for the discrete case) *Denote by $\#\mathsf{Heavy}$ the number of edges of weight 2 in the Worst Local optimum. For a constant p we have*

$$\mathbb{E}(\#\mathsf{Heavy}) \in O(\log(n)).$$

Proof. Assume we have more than $c\cdot\log(n)$ heavy edges with $c \geq 5$ and $c \geq \frac{5}{\xi(p)}$. Since $c\log(n) > \log(n)$ we can use Lemma 4.2. Thus we find that the probability P_c of this happening is at most

$$\exp\left(n\log(n)\left(1 - \frac{c}{4}\xi(p)\right)\right).$$

In the case we have more than $c\log(n)$ heavy edges, we will still no more than n heavy edges. It follows that

$$\mathbb{E}\left(\#\mathsf{Heavy}\right) \leq n\cdot P_c + O(\log(n)).$$

We know that $P_c \geq \exp(n\log(n)(1-5/4) = \exp(-0.25n\log(n)) = n^{-n/4}$ because of the constraints on c. We then also know for large n that $n\cdot P_c$ is in $O(\log(n))$. The result follows.

■

5 Exponential distribution

After looking at the uniform distribution and discrete cases we also look at instances where the distances are drawn from the exponential distribution. In this section we will first see how we are going around the issue that the distances do not have an upper bound. After that we look for a order bound on $\mathbb{E}(\mathrm{WLO}_n/\mathrm{OPT}_n)$.

Consider a complete graph with n vertices where the weights of the edges are independently drawn from the exponential distribution with parameter $\lambda = 1$. We also assume we have at least 4 edges in the Hamiltonian cycles, so $n \geq 4$.

Remember that random variables with an exponential distribution can reach arbitrarily large values. We want to have a bound for this such that the probability of an edge having a weight over the bound is very small. First we find a bound on the probability that the maximum edge weight is more than q.

Lemma 5.1

$$\mathbb{P}\left(\max_{e\in E} w(e) > q\right) \leq 2ne^{-q} \text{ for } q > 0.25$$

Proof. We use that the weights of the edges are independent.

$$\begin{aligned}
\mathbb{P}(\max_{e\in E} w(e) > q) &= 1 - \mathbb{P}(\max_{e\in E} w(e) \leq q) \\
&= 1 - \mathbb{P}\left(w(e_1) < q, w(e_2) < q, \ldots, w(e_n) < q\right) \\
&= 1 - \mathbb{P}\left(w(e_1) < q\right)\mathbb{P}\left(w(e_2) < q\right)\ldots\mathbb{P}\left(w(e_n) < q\right) \\
&= 1 - (1 - e^{-q})^n \\
&= 1 - (1 + (-e^{-q}))^n \\
&\leq 1 - \left(e^{-2e^{-q}}\right)^n \quad \text{by using } 1 + 0.5x \geq e^x \text{ for } -1.5 \leq x \leq 0 \\
&= 1 - \left(e^{-2ne^{-q}}\right) \\
&\leq 1 - \left(1 - 2ne^{-q}\right) \text{ by using } e^x \geq 1 + x \text{ for all } x \\
&= 2ne^{-q}
\end{aligned}$$

■

We bound the weight at $\kappa = 100\log(n)$ and find $\mathbb{P}(\max_{e\in E} w(e) > \kappa) \leq 2n^{-99}$ using Lemma 5.1. We refer back to this bound on the weight of edges as the *κ-bound*.

In this case we have to work with a slightly different cumulative density function for the weights of the edges. The mass that in the exponential distribution on values over κ is now equally divided over the remainder. This means that the cumulative density function is a scaling of the original one. See Figure 5 for a graphical representation. We have $\mathbb{P}(w(e) = q)_L = c \cdot \mathbb{P}(w(e) = q)$ and $\mathbb{P}(w(e) \leq q)_L = c \cdot \mathbb{P}(w(e) \leq q)$ where the subscript L indicates the limited version and c is very close to 1. However, to make notation cleaner we use $c = 2$.

We now look into the probability that a cycle H is locally optimal and find an upper bound for that.

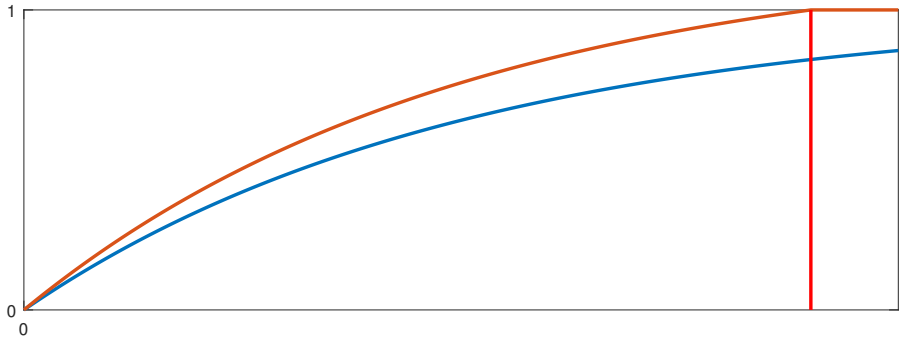

Figure 5: We choose the red vertical line as our κ-bound. The cumulative density function is scaled with a constant factor being $\frac{1}{F(\kappa)}$ so it reaches one exactly at κ.

Lemma 5.2 *Let $\eta > 0$. Let H be any fixed Hamiltonian cycle. Assume H contains at least m edges of weight at least η. The weights for the other edges are iid $Exp(\lambda)$.*

$$\mathbb{P}(H \text{ is locally optimal}) \leq 4(\eta+1)e^{-\eta\frac{mn}{4}}$$

Proof.

We call an edge *heavy* if it has a weight greater than η. By Lemma 3.1 there are at least $mn/64$ independent pairs of edges where at least one edge is heavy. Consider such a pair e and e' with e heavy which is possible without loss of generality. When performing the 2-opt operation these edges are replaced by the edges f and f' as determined by the 2-opt operation. We know that if $w(e)+w(e') > w(f)+w(f')$ that H is not a locally optimal tour. Since the weights are non-negative we know that $w(e) + w(e') \geq \eta$. So, if $w(f) + w(f') < \eta$ we know that H is not optimal.

Now we determine $\mathbb{P}(w(f)+w(f') < \eta \mid \kappa\text{-bound})$, the probability that the edges that are inserted by a 2-opt operation have a low enough weight.

$$\begin{aligned}
\mathbb{P}(w(f) + w(f') \leq \eta \mid \kappa\text{-bound}) &\leq \int_0^\eta 2 \cdot F(x) \cdot 2 \cdot f(\eta - x)dx \\
&= \int_0^\eta 4\left(1 - e^{-x}\right) e^{-(\eta-x)} \\
&= 4\left(\int_0^\eta e^{-\eta+x}dx - \int_0^\eta e^{-\eta}dx\right) \\
&= 4\left(\int_0^\eta e^{-\eta+x}dx - \eta e^{-\eta}\right) \\
&= 4\left(\int_0^\eta e^{-x}dx - \eta e^{-\eta}\right) \\
&= 4\left(\left[-e^{-x}\right]_0^\eta - \eta e^{-\eta}\right) \\
&= 4 - 4(\eta+1)e^{-\eta}.
\end{aligned}$$

For an independent combination of edges where one is heavy an improving 2-opt operations is possible with probability at most $1 - (\eta+1)e^{-\eta}$. In order for a tour H to be optimal, it cannot have any improving 2-opt operations, in particular not for the $mn/64$ independent combinations

we found. This means that

$$\mathbb{P}(H \text{ is locally optimal}) \leq 4(\eta+1)e^{-\eta\frac{mn}{64}} - 3 \leq 4(\eta+1)e^{-\eta\frac{mn}{64}} \tag{8}$$

■

Lemma 5.3 *For any $c \geq 50$ we have*

$$\mathbb{P}\left(WLO_n \geq 20c\sqrt{n\log(n)}\cdot\log(n)\right) \leq cn^{10}\exp\left(n\log(n) - c\frac{\sqrt{\frac{\log(n)}{n}}n^2}{64}\right)$$

Proof. We find the probability that a cycle H is over a certain weight. For that we need to use a way to calculate the weight of a constructed cycle for which we use Lemma 5.2. Let the classes i run from 0 upwards. We are going to look at a tour H which contains at most m_i edges of weight at least η_i. For each i we count the number of edges with weight more that η_i. We count these with weight η_{i+1}. For some edges this may be too low but these are counted again for a higher i. This gives the following expression for the weight

$$w(H) \leq \sum_{i=0}^{\infty} m_i\eta_{i+1}.$$

Define $m_i = 2^{-i}n$, $\eta_i = 2^i\eta$ and $\eta = c\cdot\sqrt{\log(n)n^{-1}}$. Because the weights are at most $100\log(n)$ under the κ-bound it is sufficient to consider $i \in [0,\ldots,10\log(n)-1]$, as when $i \geq 10\log(n)$ we have $n_i > \kappa$. If a tour H contains at most m_i edges of weight at least η_i then

$$w(H) \leq \sum_{i=0}^{10\log(n)-1} m_i\eta_{i+1}.$$

We have

$$m_i\eta_{i+1} = 2\eta n = 2c\sqrt{n\log(n)}.$$

As we use the κ-bound we find

$$w(H) \leq \sum_{i=0}^{10\log(n)-1} m_i\eta_{i+1} = \sum_{i=0}^{10\log(n)-1} 2c\sqrt{n\log(n)} = c\sqrt{n\log(n)}\sum_{i=0}^{10\log(n)-1} 2 \leq 20c\sqrt{n\log(n)}\cdot\log(n).$$

We now want to estimate the probability that the tour H which contains at most m_i edges of weight at least η_i is optimal. For the probability of this to happen, we refer back to Lemma 5.2. The bound we get is $4(\eta+1)e^{-\eta\frac{mn}{10}}$.

We first look at the case for a fixed i. Fix any tour H. The probability that H is locally optimal provided it contains at least m_i edges of weight at least η_i (call these conditions $\star_i$) is at most $4\cdot(\eta_i+1)\exp\left(-\eta_i\frac{m_in}{10}\right)$. Thus,

$$\begin{aligned}\mathbb{P}(H \text{ is optimal under } \star_i) &\leq 4\cdot(\eta_i+1)\exp\left(-\eta_i\frac{m_in}{64}\right)\\ &= 4\cdot(2^i\eta+1)\exp\left(-\frac{\eta m^2}{64}\cdot\right)\end{aligned}$$

We use Boole's inequality to bound the probability that H is locally optimal from above, provided there exists an $i \in [0, \ldots, 10\log(n) - 1]$ for which H contains at least m_i edges of weight at least η_i. Again using Boole's inequality, we determine an upper bound to the probability that one of the $n!$ possible tours is locally optimal, provided that it contains at least m_i edges of weight η_i for some i. This probability is at most

$$4 \cdot n! \cdot 10\log(n) \cdot (2^{10\log(n)}\eta + 1) \exp\left(-\frac{\eta n^2}{64}\cdot\right).$$

We work on this expression to find

$$\begin{aligned}
\mathbb{P}\left(\mathrm{WLO}_n \geq 20c\sqrt{n\log(n)} \cdot \log(n) \mid \kappa\text{-bound}\right) &\leq 4 \cdot n! \cdot 10\log(n) \cdot (2^{10\log(n)}\eta + 1) \exp\left(-\frac{\eta n^2}{64}\right) \\
&\leq n^n \cdot 10\log(n) \cdot (2^{10\log(n)}\eta + 1) \exp\left(-\frac{\eta n^2}{64}\right) \\
&= (2^{10\log(n)}\eta + 1) \exp\left(n\log(n) - \frac{\eta n^2}{64}\right) \\
&= \left(2^{10\log(n)} c\sqrt{\frac{\log(n)}{n}} + 1\right) \exp\left(n\log(n) - c\frac{\sqrt{\frac{\log(n)}{n}}n^2}{64}\right)
\end{aligned}$$

We note that for $n \to \infty$ $\mathbb{P}\left(\mathrm{WLO}_n \geq 20c\sqrt{n\log(n)} \mid \kappa\text{-bound}\right)$ goes to zero. When $c \geq 50$ the maximum is before $n = 5$.

We include the properties of the κ-bound to find

$$\begin{aligned}
&\mathbb{P}\left(\mathrm{WLO}_n \geq 20c\sqrt{n\log^3(n)}\right) \\
&= \mathbb{P}\left(\mathrm{WLO}_n \geq 20c\sqrt{n\log^3(n)} \mid \kappa\text{-bound}\right) + \mathbb{P}\left(\mathrm{WLO}_n \geq 20c\sqrt{n\log^3(n)} \mid \text{no } \kappa\text{-bound}\right) \\
&\leq \left(2^{10\log(n)} c\sqrt{\frac{\log(n)}{n}} + 1\right) \exp\left(n\log(n) - c\frac{\sqrt{\frac{\log(n)}{n}}n^2}{64}\right) \cdot \left(1 - \left(\frac{1}{2n^{99}}\right)\right) + 1 \cdot \left(\frac{1}{2n^{99}}\right)
\end{aligned}$$

where the last 1 is an upper bound for any probability.

When n grows larger the left side of the above equation dominates. From this we conclude

$$\begin{aligned}
\mathbb{P}\left(\mathrm{WLO}_n \geq 20c\sqrt{n\log(n)} \cdot \log(n)\right) &\leq \left(2^{10\log(n)} c\sqrt{\frac{\log(n)}{n}} + 1\right) \exp\left(n\log(n) - c\frac{\sqrt{\frac{\log(n)}{n}}n^2}{64}\right) \\
&\leq \left(e^{10\log(n)} c\right) \exp\left(n\log(n) - c\frac{\sqrt{\frac{\log(n)}{n}}n^2}{64}\right) \\
&\leq cn^{10} \exp\left(n\log(n) - c\frac{\sqrt{\frac{\log(n)}{n}}n^2}{64}\right)
\end{aligned}$$

■

As we have a result on WLO_n and we want to work on $\mathbb{E}(\mathrm{WLO}_n/\mathrm{OPT}_n)$ we also need to find a result for OPT_n.

Lemma 5.4

$$\mathbb{P}(OPT_n \leq c) \leq c^n$$

Proof. We are going to determine the probability that a cycle is lighter than c. With that, we can determine the probability that all of the $n!$ cycles are lighter than c. The result we find also holds for the shortest cycle.

We have for a cycle C

$$\mathbb{P}\left(w(e_1) + w(e_2) + \ldots + w(e_n) \leq c\right) \leq \frac{c^n}{n!}$$

We prove this by induction. For the base case we have

$$\begin{aligned}
\mathbb{P}(w(e_1) \leq c) &= \int_0^c e^{-x} dx \\
&\leq \int_0^c 1 dx \\
&= \frac{c^1}{1!} = c
\end{aligned}$$

Now we assume that $\mathbb{P}(w(e_1) + w(e_2) + \ldots + w(e_{n-1}) \leq c) \leq c^{n-1}/(n-1)!$. If this is the case we have

$$\begin{aligned}
\mathbb{P}\left(w(e_1) + w(e_2) + \ldots + w(e_n) \leq c\right) &= \int_0^c e^{-x} \mathbb{P}\left(w(e_1) + w(e_2) + \ldots + w(e_{n-1}) \leq c - x\right)\ dx \\
&\leq \int_0^c e^{-x} \frac{(c-x)^{n-1}}{(n-1)!}\ dx \\
&\leq \int_0^c \frac{(c-x)^{n-1}}{(n-1)!}\ dx \\
&= \int_0^c \frac{x^{n-1}}{(n-1)!}\ dx \\
&= \left[\frac{x^n}{n!}\right]_0^c \\
&= \frac{c^n}{n!}
\end{aligned}$$

We use the union bound to find

$$\mathbb{P}(\text{weight of some cycle} \leq c) \leq n! \cdot \frac{c^n}{n!} = c^n$$

■

Now we prove the final result of this part.

Theorem 5.5 (Result for the exponential case) *We have*

$$\mathbb{E}\left[\frac{WLO_n}{OPT_n}\right] \in O\left(\sqrt{n \log^3(n)}\right)$$

Proof. Assume $\mathrm{WLO}_n/\mathrm{OPT}_n > 20c^2\sqrt{n\log^3(n)}$ for $c > 200$. Then $\mathrm{WLO}_n \geq 20c\sqrt{n\log^3(n)}$ or $\mathrm{OPT}_n \leq \frac{1}{c}$. The probability that $\mathrm{WLO}_n \geq 20c\sqrt{n\log^3(n)}$ is given by Lemma 5.3. The probability that $\mathrm{OPT}_n < \frac{1}{c}$ is at most c^{-n} by Lemma 5.4. The probability of either of these events happening is at most

$$P(c) = cn^{10}\exp\left(n\log(n) - c\frac{\sqrt{\frac{\log(n)}{n}}n^2}{64}\right) + c^{-n}$$

We then have two cases, $\mathrm{WLO}_n/\mathrm{OPT}_n > 20c^2\sqrt{n\log^3(n)}$ and $\mathrm{WLO}_n/\mathrm{OPT}_n \leq 20c^2\sqrt{n\log^3(n)}$. We have a probability limiting the first case and for the second case we have a bound on the value of the ratio. We will see that the probability for this first case goes to zero very fast.

This leads to

$$\frac{\mathrm{WLO}_n}{\mathrm{OPT}_n} \leq 20\sqrt{n\log^3(n)} \cdot \int_{4000}^{\infty} cP(c)dc^2 + O\left(\sqrt{n\log^3(n)}\right) \tag{9}$$

and after substituting $x = c^2$ we find

$$\frac{\mathrm{WLO}_n}{\mathrm{OPT}_n} \leq 10\sqrt{n\log^3(n)} \cdot \int_{4000}^{\infty} P(\sqrt{x})dx + O\left(\sqrt{n\log^3(n)}\right). \tag{10}$$

We calculate $\int_{x=4000}^{\infty} P(\sqrt{x})dx$ by splitting it into the two separate parts

$$\int_{x=4000}^{\infty} P(\sqrt{x})dx = \int_{x=4000}^{\infty} \sqrt{x}^{-n}dx + \int_{x=4000}^{\infty} \sqrt{x}n^{10}\exp\left(n\log(n) - \sqrt{x}\frac{\sqrt{\frac{\log(n)}{n}}n^2}{64}\right)dx.$$

We work on these independently to find

$$\int_{x=4000}^{\infty} \sqrt{x}^{-n}dx = \frac{2^{6-\frac{5n}{2}}5^{3-\frac{3n}{2}}}{n-2} \qquad \text{and}$$

$$\int_{x=4000}^{\infty} \sqrt{x}n^{10}\exp\left(n\log(n) - \sqrt{x}\frac{\sqrt{\frac{\log(n)}{n}}n^2}{64}\right)dx$$
$$= \frac{4096n^{n+4}e^{\frac{1}{8}(-5)\sqrt{\frac{5}{2}}n^2\sqrt{\frac{\log(n)}{n}}}\left(125n^3\log(n) + 80\sqrt{10}n^2\sqrt{\frac{\log(n)}{n}} + 256\right)}{\left(\frac{\log(n)}{n}\right)^{3/2}}$$

Combining these we find

$$\int_{x=4000}^{\infty} P(\sqrt{x})dx = \frac{2^{6-\frac{5n}{2}}5^{3-\frac{3n}{2}}}{n-2} + \frac{4096n^{n+4}e^{\frac{1}{8}(-5)\sqrt{\frac{5}{2}}n^2\sqrt{\frac{\log(n)}{n}}}\left(125n^3\log(n) + 80\sqrt{10}n^2\sqrt{\frac{\log(n)}{n}} + 256\right)}{\left(\frac{\log(n)}{n}\right)^{3/2}}.$$

The dominating term is $\exp\left(\frac{1}{8}(-5)\sqrt{\frac{5}{2}}n^2\sqrt{\frac{\log(n)}{n}}\right)$ which for $n > 4$ is always smaller than $\exp(-n)$. Thus this function goes to zero at least exponentially fast as n increases for $n > 4$. We can say that $\int_{c=200}^{\infty} P(c)\, dc \in O(1)$. This combined with (10) leads to the following result

$$\mathbb{E}\left[\frac{\mathrm{WLO}_n}{\mathrm{OPT}_n}\right] \leq O\left(\sqrt{n\log^3(n)}\right)\cdot O(1) + O(\sqrt{n\log^3(n)}) \in O\left(\sqrt{n\log^3(n)}\right) \tag{11}$$

■

6 Simulations

6.1 Introduction

After finding theoretical results we also want to verify our results experimentally. For this we ran some simulations. In these simulations we use integer linear programs to solve different instances both to optimality as well as to the worst local optimum with respect to the 2-opt neighbourhood. First we describe the models and procedures. The project files are based on the material by Bodo Manthey for the course "Optimization Modeling". We use the software AIMMS to model the problems and CPLEX 12.6.2 to solve them. The experiments were run on a Windows 10 desktop machine with an Intel core i5 3.2 GHz processor and 16 GB of RAM. The time results may not be entirely consistent because of the usage of a desktop on which other applications are running (e.g. windows update). While the goal of the experiment is not to compare times, it may have affected the size of the instances that can be solved. However, as we see later the problems itself are very large and we expect this effect to be negligible.

We recall that solving the TSP is already an NP-hard problem. When solving for the worst local optimum we need another problem. We want to find a solution to the TSP (that is, a tour that visits each node exactly once). However, this solution also needs to be 2-optimal. That means there cannot be an improving 2-opt operation. We formalize this notion as follows:

$$\text{If } (u,v) \text{ and } (x,y) \text{ are containted in the tour, then } d_{u,v} + d_{x,y} \leq d_{u,x} + d_{y,v}. \tag{12}$$

where u, v, x, y are distinct nodes, $\mathsf{Tour}(u,v)$ is true if and only if the edge (u,v) is used and $\mathsf{Distance}(u,v)$ gives the distance from u to v. We clearly see the 2-opt idea in this condition. To solve for the worst local optimum (WLO) we have instances like in the TSP. Solutions are tours that visit all nodes exactly once and meet condition (12). The goal is then to *maximize* the total distance.

As we see solving the WLO offers additional challenges and constraints. In the section we first describe the model and procedures used. After that we present the results for the three different classes of instances covered in this thesis. In the next section we discuss these results in relation with the theoretical results found before.

6.2 TSP Model

We present the model in a systematic way by listing the sets used, the parameters that are assumed to be known beforehand, the variables, the constraints and the goal.

Sets	**Description**
u, v, x, y	Set of all nodes
α	Set of sub tour elimination constraints

Parameters	**Description**
$d_{u,v}$	Length of the edge from u to v

Variables	**Description**
L	Length of the tour
$e_{u,v}$	Indicator variable, 1 if edge (u, v) is used in the tour, 0 otherwise
i_u	Indegree of u, the amount of edges in the tour that go into u
o_u	Outdegree of u, the amount of edges in the tour that go out of u

Constraints		**Description**
$L = \sum_{u,v} e_{u,v} \cdot d_{u,v}$		Definition for total length
$i_u = \sum_{v\|v\neq u} e_{u,v}$	$\forall\, u$	Definition for indegree
$o_u = \sum_{v\|v\neq u} e_{v,u}$	$\forall\, u$	Definition for outdegree
$i_u = 1$	$\forall\, u$	Each node has one edge entering
$o_u = 1$	$\forall\, u$	Each node has one edge leaving
$e_{u,v} + e_{v,u} \leq 1$	$\forall\, (u, v) \| u \neq v$	Prevent cycles of length 2 from occuring.
(13)	$\forall\, \alpha$	Subtour elimination
(14)	$\forall\, \alpha$	Subtour elimination

Goal	**Description**
$\min L$	Minimize the total length of the tour

In itself this model can give non-feasible solutions. A solution can consist of multiple cycles such that each node is on exactly one cycle. This is of course not a feasible solution, we are looking for a *single* tour that visits all nodes. To combat this we add *sub tour elimination constraints*. We could add constraints that rule out all cycles of length less than n. However, this would result in very many constraints in our problem. To get some of the benefits we do add constraints that prevent cycles of length 2, that is, the tour just going between two nodes. However, this does not prevent all subtours. Instead we look at the solution we have. If we find multiple subtours we add a constraint to the LP that prevents that subtour from occuring. Then we solve the LP again and repeat this procedure until we find a single tour through all points.

We formalize the described procedure by describing it in pseudocode in Algorithm 6.1. Given a solution that could consist of multiple subtours we first want to find the subtours. For this we take an as-of-yet unlabeled node, give it a new label and give all nodes connected to it the same label. We keep taking new unlabeled points as starting point until we have covered all nodes. After that we to ensure that in the next iteration these subtours cannot exist anymore. We do this by demanding that there is at least one edge going into the subtour and one edge going out.

This gives the following constraints

$$\sum_{v \in \mathcal{C}} i_v \geq 1 + \sum_{u,v \in \mathcal{C}} e_{u,v} \qquad \forall \text{ subtours } \mathcal{C} \qquad \text{(indegree)} \tag{13}$$

$$\sum_{v \in \mathcal{C}} o_v \geq 1 + \sum_{u,v \in \mathcal{C}} e_{v,u} \qquad \forall \text{ subtours } \mathcal{C} \qquad \text{(outdegree).} \tag{14}$$

By iteratively running Algorithm 6.1, adding the returned constraints to α and solving the LP we find the optimal solution for a TSP instance. We terminate when we have indeed found a single tour that visits all nodes.

Algorithm 6.1 Subtour elimination

Input: Solution possibly containing subtours
Output: Either subtour elimination constraints or `OK` if none are needed

```
1:  subtour_u ← 0 ∀u
2:  #_subtour ← 0
3:  #_reached ← 0
4:  while not all nodes are labeled do
5:    #_subtour ← #_subtour + 1
6:    for all nodes u do {find the new starting node for a subtour}
7:      if subtour_u = 0 then
8:        subtour_u ← #_subtour
9:        break
10:     end if
11:   end for
12:   while subtour is updated do {label all nodes in this subtour}
13:     for all nodes u, v do
14:       if e_{u,v} + e_{v,u} > 0 and subtour_u = #_subtour and subtour_v ≠ #_subtour then
15:         subtour_v ← #_subtour
16:       end if
17:     end for
18:   end while
19: end while
20: if #_subtour = 1 then
21:   return OK
22: end if
23: for all subtour C do
24:   return constraints (13) and (14) for subtour C.
25: end for
```

6.3 WLO Model

When we want to find the worst local optimum we want to find the tour of maximum length that we cannot improve further. This means that we change our goal to maximization.

After that we need to add the constraints that make sure the solution is indeed a local optimum, that is, we can find no improving 2-opt operation. For that we need to assure that Condition (12) holds for all edges in the tour. We however use a different formulation for the constraint:

$$e_{u,v} + e_{x,y} \leq 1 \qquad \text{for all distinct } u, v, x, y \text{ with } d_{u,v} + d_{x,y} > d_{u,x} + d_{y,v} \tag{15}$$

If we would add all of these $O(n^4)$ constraints the problem would get too big even for a large number of nodes. So instead we add these 2-opt optimality constraints if we find they are needed.

That is, after each iteration we check for which u, v, x and y Constraint 15 is violated and add the respective constraints to our problem. This will eventually add all constraints that are needed for solving the WLO problem. During trial experiments we noted that added slightly more constraints that would strictly be required helps the solving. That is, instead of just adding Constraint 15 for the needed u, v, x and y we add some other related constraints as well. Once we have distinct u, v, x and y we add the following constraints: If $d_{u,v} + d_{x,y} \geq d_{u,x} + d_{y,v}$ we add Constraint 15 for u, v, x, y if at least one of the edges (u, v) or (x, y) is used. If $d_{u,v} + d_{x,y} < d_{u,x} + d_{y,v}$ we add Constraint 15 for u, x, y, v if we use both edges (u, v) and (x, y).

The solving of this problem is then done according to Algorithm 6.2. In line 3 we solve the problem described in this section, again using Algorithm 6.1 for the subtour elimination. We keep iteratively applying the same procedure but in every iteration we add one or more constraints to ensure the solutions are 2-opt optimal. If we can no longer find constraints to add we must have found a 2-opt optimal solution. Since we are maximizing the tour distance, it will also be the worst local optimum.

The entire model is as follows:

Sets	**Description**
u, v, x, y	Set of all nodes
α	Set of sub tour elimination constraints
β	Set of 2-opt optimality constraints

Parameters	**Description**
$d_{u,v}$	Length of the edge from u to v

Variables	**Description**
L	Length of the tour
$e_{u,v}$	Indicator variable, one if edge (u, v) is used in the tour, zero otherwise
i_u	Indegree of u, the amount of edges in the tour that go into u
o_u	Outdegree of u, the amount of edges in the tour that go out of u

Constraints		**Description**
$L = \sum_{u,v} e_{u,v} \cdot d_{u,v}$		Definition for total length
$i_u = \sum_{v\|v\neq u} e_{u,v}$	$\forall\, u$	Definition for indegree
$o_u = \sum_{v\|v\neq u} e_{v,u}$	$\forall\, u$	Definition for outdegree
$i_u = 1$	$\forall\, u$	Each node has one edge entering
$o_u = 1$	$\forall\, u$	Each node has one edge leaving
$e_{u,v} + e_{v,u} \leq 1$	$\forall\, (u, v) \| u \neq v$	Prevent cycles of length 2 from occuring.
(13)	$\forall\, \alpha$	Subtour elimination
(14)	$\forall\, \alpha$	Subtour elimination
(15)	$\forall\, \beta$	2-opt optimality

Goal	**Description**
$\max L$	Maximize the total length of the tour

Algorithm 6.2 Worst local optimum constrain generation

Input: Solution possibly containing subtours
Output: Either subtour elimination constraints or `OK` if none are needed

```
1: while 1=1 do
2:   found← 0
3:   Solve TSP(max) including subtour elimination
4:   for all nodes u, v, x, y distinct do
5:     if e_{u,v} + e_{x,y} ≥ 1 then
6:       if d_{u,v} + d_{x,y} ≥ d_{u,x} + d_{y,v} then
7:         Add constraint (15) for (u, v), (x, y) to β
8:         if e_{u,v} + e_{x,y} = 2 then
9:           found← 1
10:        end if
11:      end if
12:      if e_{u,v} + e_{x,y} = 2 then
13:        if d_{u,v} + d_{x,y} < d_{u,x} + d_{y,v} then
14:          Add constraint (15) for (u, x), (y, v) to β
15:        end if
16:      end if
17:    end if
18:  end for
19:  if found= 0 then
20:    break
21:  end if
22: end while
```

6.4 Uniform on [0, 1]

As we have now determine how we are going to find the solution for a given instance, we consider other related matters. First of all, for the uniform case we calculate both the optimal solution OPT and the worst local optimum solution WLO. For every instance we solve we then calculate the fraction WLO/OPT. If the worst local optimum is equal to the optimal solution we get a fraction of 1. Otherwise we get a fraction larger than 1.

Now we know what to do with each individual instance we need to generate instances. Our version of the TSP is symmetric but does not satisfy the triangle inequality. Symmetric means that the distance from a to b is equal to the distance from b to a. When an instance satisfies the triangle inequality we mean that for all nodes u, v, w the distance from u to v directly is not longer than the distance from u to w and from w to v added together. We read in the introduction that an instance is a complete graph on n nodes with distances defined for every edge. This means we can represent a TSP instance as a distance matrix, which makes the generation of instances easier. In the distance matrix we find both the number of edges and the distances for each edge. Thus to generate an instance, we should create a matrix of size $n \times n$ and fill it with distances such that the matrix is symmetric. We assure symmetry by only filling out the values above the diagonal and then mirroring them. The algorithm for this in a general case is shown as Algorithm 6.3. For the uniform case we choose for distribution G of course Uniform$[0, 1]$.

For this case simulations were done for 5 upto 27 nodes. First 25 iterations per choice for the number of nodes upto 24. After that 15 extra iterations for 13 and 14 nodes were performed as the standard deviation was considerably higher than for other nodes. For 25 nodes 19 simulations were performed, whereas for 26 nodes there are 15 iterations and for 27 nodes only 7. This low number of iteration is due to the extremely long execution times for solving these problems. For the

Algorithm 6.3 Generating instances

Input: A distribution for the edge distances G and a number of nodes n.
Output: A TSP instance represented by a distance matrix

1: Define nodes 1 to n
2: Create a $n \times n$ zero matrix d
3: **for all** nodes $u, v \mid v > u$ **do**
4: $\quad d_{u,v} \sim \mathsf{G}$
5: $\quad d_{v,u} = d_{u,v}$
6: **end for**

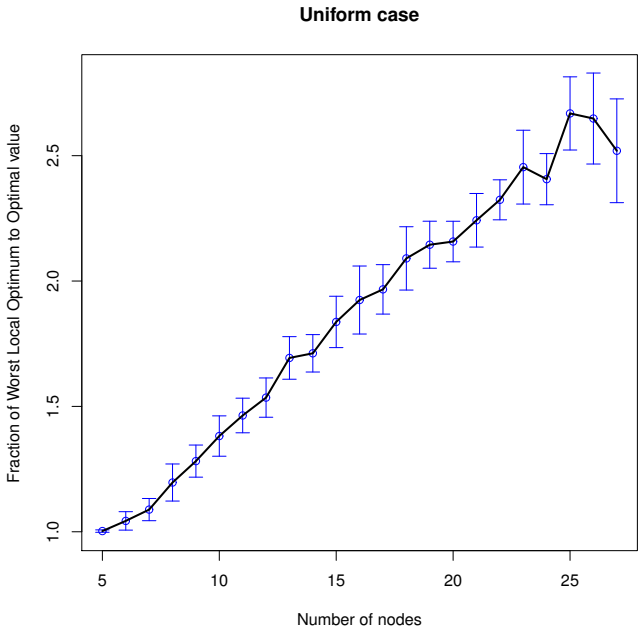

Figure 6: Simulation results for the uniform case with 95% confidence intervals

simulations with 27 nodes, on average 217483 2-opt constraints were generated with an estimated standard deviation of 8081. The time for one iteration with 27 nodes is about 12790 seconds, more than 3.5 hours.

The results for this case are shown in Figure 6. This figure shows the average fraction found by the simulations along with a 95% confidence interval. The matching data is shown in Table 11.

Another matter to consider is if the simulation results agree with the theoretical results. This is the experimental validation. For the uniform case we proved a bound of $O\left(\sqrt{n\log(n)}\right)$ in Section 3. We check this by looking at

$$\frac{\overline{\left(\frac{\widehat{\mathsf{WLO}}_n}{\widehat{\mathsf{OPT}}_n}\right)}}{\sqrt{n\log(n)}}. \tag{16}$$

Here $\widehat{\mathsf{WLO}}_n$ and $\widehat{\mathsf{OPT}}_n$ indicate the experimentally obtained values for WLO_n and OPT_n respectively. We then compute the average value of $\widehat{\mathsf{WLO}}_n/\widehat{\mathsf{OPT}}_n$ which is indicated by the bar over this fraction. This gives us the experimental estimate for $\mathbb{E}[\mathsf{WLO}_n/\mathsf{OPT}_n]$.

Number of nodes	Estimated average	Estimated standard deviation	Number of iterations
5	1.002762061	0.012200828	25
6	1.043333176	0.093769865	25
7	1.088438295	0.112921123	25
8	1.19643585	0.188674616	25
9	1.281828959	0.163432418	25
10	1.381807972	0.205388772	25
11	1.463787693	0.176103551	25
12	1.535149605	0.199928777	25
13	1.693302675	0.274069222	40
14	1.711762586	0.240691278	40
15	1.836924843	0.261228214	25
16	1.924290247	0.345988974	25
17	1.966761186	0.251515322	25
18	2.090280041	0.322149668	25
19	2.144849502	0.23920854	25
20	2.157567109	0.291760846	50
21	2.242299372	0.272612426	25
22	2.323941757	0.257405588	40
23	2.454255232	0.375520725	25
24	2.406278271	0.260228707	25
25	2.668498752	0.324578531	19
26	2.648143386	0.358483189	15
27	2.519533236	0.279337927	7

Table 11: Simulation results for the uniform case

If $\mathbb{E}[\mathsf{WLO}_n/\mathsf{OPT}_n]$ is in $O(\sqrt{n\log(n)})$ we know that there is a positive constant M such that for large enough n we have $|\mathbb{E}[\mathsf{WLO}_n/\mathsf{OPT}_n]| \leq M \cdot |\sqrt{n\log(n)}|$. In particular this means that for large enough n we have

$$\frac{\mathbb{E}\left[\mathsf{WLO}_n/\mathsf{OPT}_n\right]}{\sqrt{n\log(n)}} \leq M.$$

As such we expect the quantity in Equation (16) to eventually be a horizontal line. This would indicate that we have found a good bound. If the line would go down we divide by something that grows more quickly than the quantity $\mathsf{WLO}_n/\mathsf{OPT}_n$ and we expect there to exist a beter bound. The results for the experiments are shown in Figure 7.

In this figure we do not see the results that we would expect to see. The line does not seem to converge to a horizontal line based on the data we collected. We discuss the significance of the experimental results in the next chapter.

6.5 Discrete distribution

In the case of the discrete distribution we make some changes compared to the previous section. In the case of the discrete distribution we focused more on counting how many heavy edges are used in the worst local optimum solution. We denote by $\#\mathsf{Heavy}$ the number of heavy edges in the worst local optimum solution. Large parts of the set-up of the experiments remain the same. We consider 5, 10, 15, 20, 25 and 30 nodes. As the discrete distribution also needs to have a parameter p, which indicates the probability with which an edge is **not** heavy. To generate the instances we use Algorithm 6.3. The generation of the random variables is handled as follows: Get a random number σ from the uniform distribution on $[0, 1]$. If $\sigma \leq p$ then the edge has weight 1,

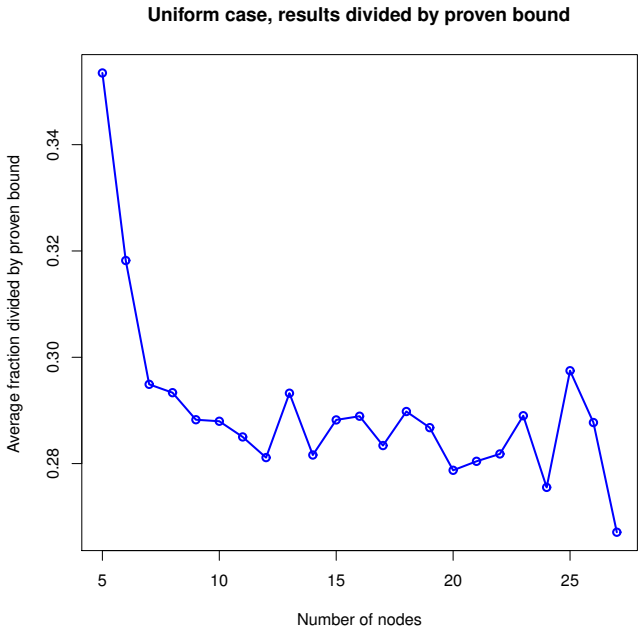

Figure 7: Experimental validation for the uniform case

otherwise weight 2. We tested the following fixed values for p: 0.05, 0.1, 0.2, 0.5, 0.8, 0.9, 0.95, 0.99. Next to that we also tested the following expressions for p: $1/n$, $1/\sqrt{n}$ and $1/\log(n)$. Note that the theoretical results do not necessarily hold for these expressions, as it is only proven for fixed values of p. The results of these simulations are presented in Figures 8 and 9 and Figures 13 to 21 in Appendix A. As in the uniform case we also want to see if we can validate the order bound proved theoretically. Our theoretical result showed that the number of heavy edges in the worst local optimum with respect to the 2-opt neighbourhood, $\mathbb{E}(\#\mathsf{Heavy}), \in O(\log(n))$ for a constant p. Hence we expect

$$\frac{\#\mathsf{Heavy}}{\log(n)}$$

to eventually be (close to) a horizontal line. The results of this is shown in Figure 10. As before we do not clearly see the horizontal line we hope for. We do however notice the symmetry in the results around $p = 0.5$. The complete tables with results are shown in the appendix.

6.6 Exponential distribution

Finally we also want to experimentally check this exponential case. We again consider the fraction $\mathsf{WLO}_n/\mathsf{OPT}_n$. For the generation of instances we also use Algorithm 6.3 using the Exponential distribution with scale 1 for G. AIMMS has a function that generates random values drawn from the exponential distribution with lower bound `l` and scale `s`: `Exponential(l,s)`. We use `Exponential(0,1)` to get a distribution with lower bound 0 and scale 1 and thus rate 1. We did 25 iterations for 5, 7, 9, 11, 13, 15, 17, 19, 21, 23 and 25 nodes. These results are shown in the appendix and visually in Figure 11.

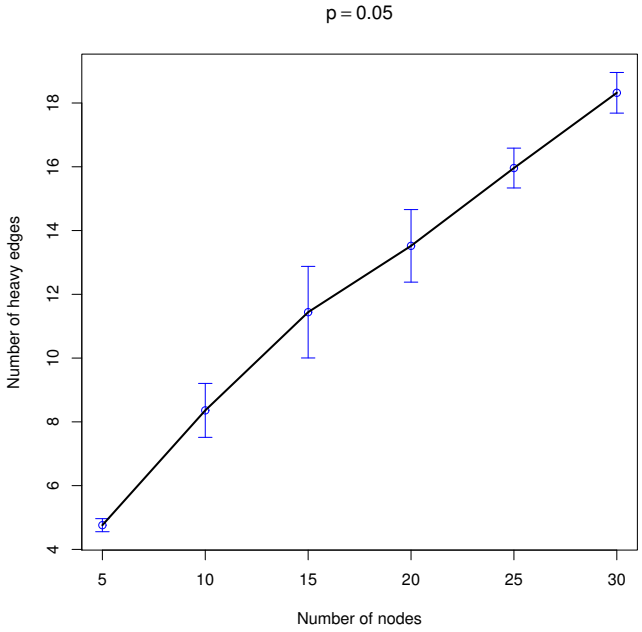

Figure 8: Simulation results for the discrete case ($p = 0.05$) with 95% confidence intervals

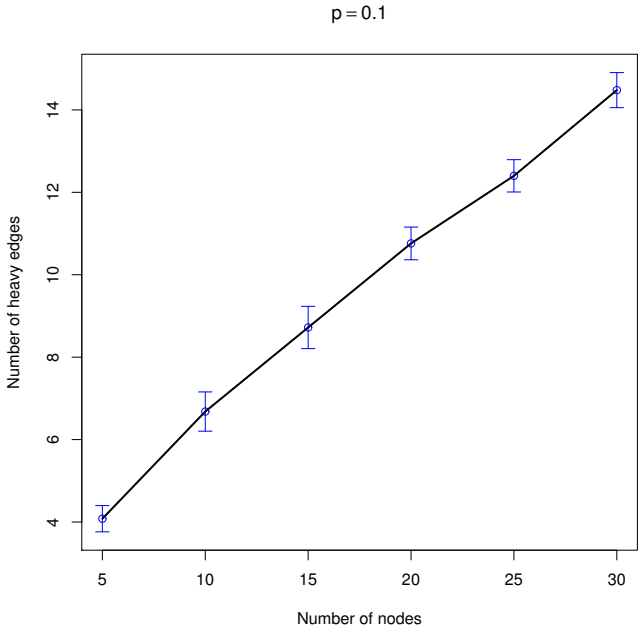

Figure 9: Simulation results for the discrete case ($p = 0.1$) with 95% confidence intervals

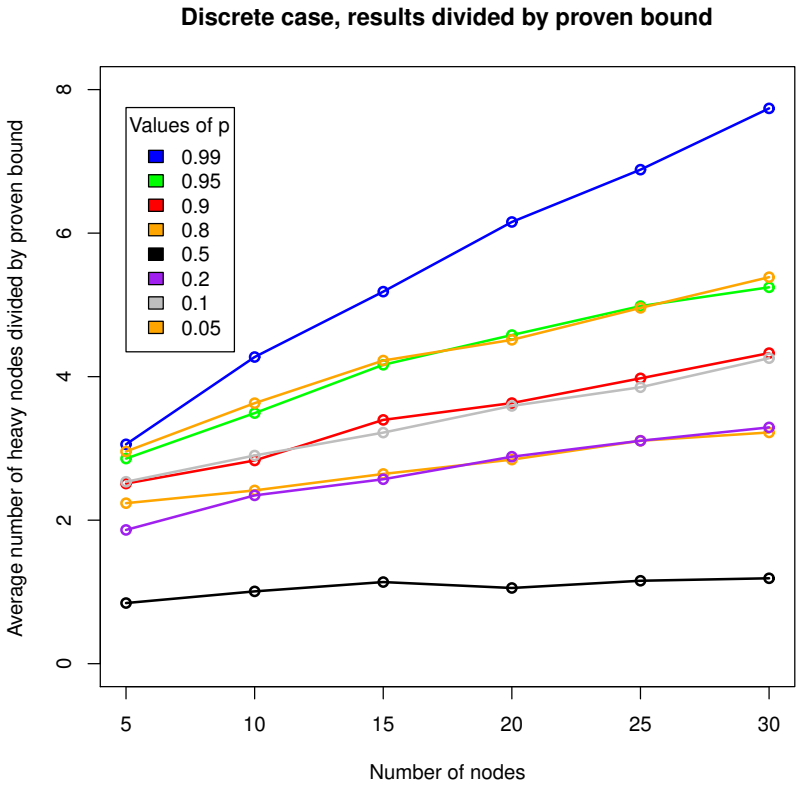

Figure 10: Experimental validation for the discrete case

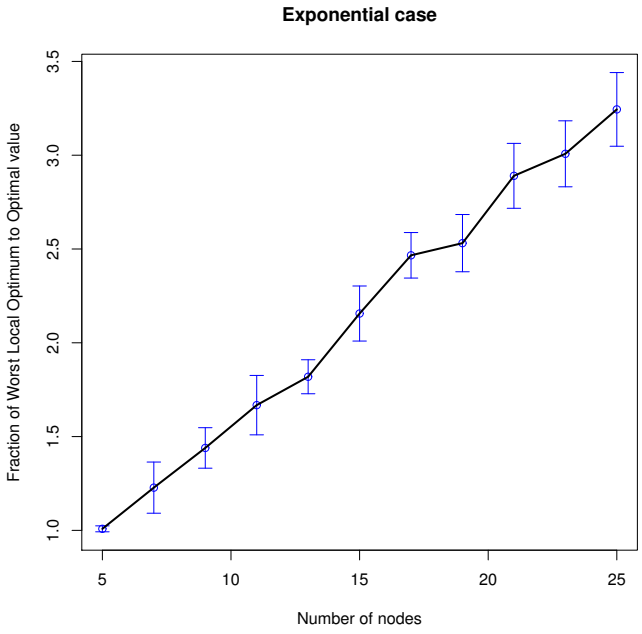

Figure 11: Simulation results for the exponential case with 95% confidence intervals

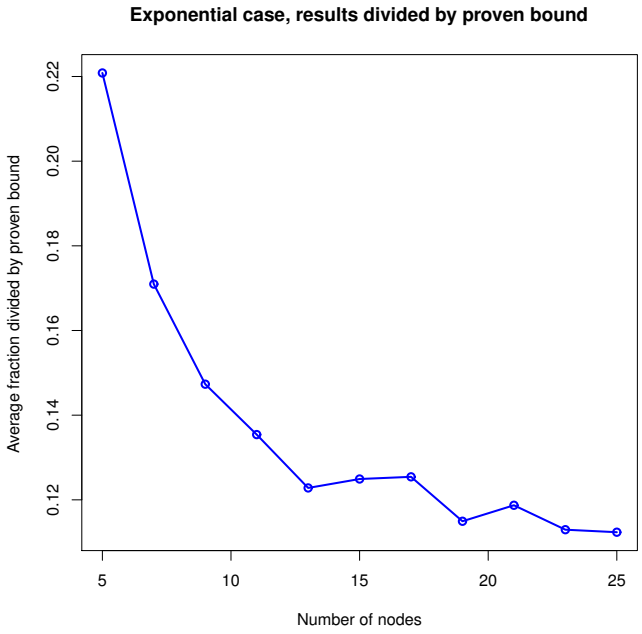

Figure 12: Experimental validation for the exponential case

The theoretical result we proved says that $\mathbb{E}(\mathsf{WLO}_n/\mathsf{OPT}_n) \in O\left(\sqrt{n\log^3(n)}\right)$. To validate we again plot

$$\frac{\frac{\mathsf{WLO}_n}{\mathsf{OPT}_n}}{\sqrt{n\log^3(n)}}$$

and expect it to eventually be a horizontal line. The results of this are shown in Figure 12.

7 Discussion

In this section we discuss our results. We go through the different variants one by one and check how they fit with known results and whether the simulations support our results.

We first start by discussing the class of instances that we called generic on $[0,1]$. This part is both an extension and an improvement of the work done by Engels and Manthey (2009). We improved the upper bound on the average approximation ratio with respect to the 2-opt neighbourhood from $O\left(\sqrt{n}\cdot(\log(n))^{3/2}\right)$ to $O\left(\sqrt{n\log(n)}\right)$. Furthermore, instead of just considering distances drawn at random from the uniform distribution on $[0,1]$ we consider a slightly more general case where also distributions with a cumulative density function such as $F(x)=\sqrt{x}$ are also possible. We are able to consider all distributions with support $[0,1]$ with the properties that (i) the cumulative density function F satisfies $F(x)\leq x$ and (ii) the probability density function f is monotonically decreasing on $[0,1]$. This also includes uniform distributions on $[0,x]$ with $0<x\leq 1$. We also ran simulations to see the average approximation ratio in randomly generated instances. However, because the simulations are very time-intensive we were unable to run any with more than 27 nodes. The results are not consistent enough to make any claims regarding the support of the results because of this constraint. Overall, for this class of instances we have improved and

expended a previous result.

We now consider the case of discrete instances. In this thesis we used instances where each edge has a distance of 1 with probability p and 2 otherwise. Note that these distances are drawn at random and independently. For the discrete case we proved a result on the number of heavy edges in the worst local optimal solution. We show that in this case the number of heavy edges in the worst local optimum solution is in $O(\log(n))$ if p is a constant strictly between 0 and 1. As far as we are aware, this is the first average-case result on the approximation performance of 2-opt for discrete probability distributions. We also ran simulation for this case, for eight different values of p and for up to 30 nodes. For the validation for the constant values of p, as shown in Figure 10 we see the data is quite consistent. We also recognize a symmetry around $p = 0.5$. Also, for $p = 0.5$ the line seems to be pretty much horizontal, indicating that we have found a good upper bound. Also the other values of p allow for this as this are not decreasing and seem to level off over so slightly. However, before we make bold claims we must recall that the number of nodes used for these experiments is really quite low. Our results show behaviour that only needs to hold from a certain number of nodes onward. Hence we use caution in our claims. We still do feel that the experimental results support our theoretical results but experiments for larger instances are needed to make stronger claims.

Lastly we considered the instances where distances are drawn from a standard exponential distribution. Because the exponential distribution does not have a bounded support, we had to use some more elaborate techniques. We assume we *can* bound the distances and we choose the bound such that the probability if it being too low is extremely small. We have shown that the expected approximation ratio for the case with distances from the exponential distribution is in $O\left(\sqrt{n\log^3(n)}\right)$. As with the discrete probability distributions, this is the first average-case analysis of 2-opt with unbounded probability distributions, as far as we are aware. We also ran simulation to test our theoretical results. In this case we used up to 25 nodes. The experiment validation, found in Figure 12 shows a decreasing line which seems to level out. However, as before, because of the very limited number of nodes we have to cautiously approach the experiments. The data looks fairly consistent and we have an upper bound, a result we proved before. We cannot make claims regarding the quality of this upper bound as we would need more simulations with a higher number of nodes to get more useful results.

8 Conclusions

In this thesis we looked at the 2-opt heuristic for the traveling salesman problem. We have three main theoretical results, one of which is an improvement on earlier work and the other two are new results. Next to this we used simulations to experimentally check our bounds.

For the case with distances drawn randomly from the uniform distribution on $[0, 1]$ (and slightly more cases) we were able to show an upper bound on the expected approximation ratio of $O(\sqrt{n\log(n)})$. When the distances are either 1 with a fixed probability and 2 otherwise we were able to bound the number of edges with weight 2 in the worst locally optimal solution by $O(\log(n))$. Lastly for distances drawn from the standard exponential distribution we were able to show $O\left(\sqrt{n\log^3(n)}\right)$ is an upper bound for the expected approximation ratio.

The simulations showed some hints that the bounds are of the right magnitude, in particular for the discrete and exponential cases. However, due to the complexity of the problem we have to take these claims with a grain of salt. All our theoretical results are statements about the asymptotical

behaviour, whereas - in particular because finding worst local optima seems to be computationally difficult - our experiments are only for a relatively small number of nodes.

The results we presented in this thesis slightly improve the known upper bound on the performance of the 2-opt heuristic for the uniform distribution case. For the discrete case and the exponential case we presented new results. However, we do not know the exact bounds yet so there is certainly more work possible in this field, either by further improving the upper bounds or by showing lower bounds on the performance. In particular proving lower bounds seems to be quite difficult. We are not aware of any concrete techniques to prove lower bounds for this problem and thus this was kept out of this master thesis. This thesis only covers a very specific part of the fascinating field of the TSP, a problem that can appear easy at first sight, but also shows the limits of what we can achieve.

Acknowledgments
I would like to thank my supervisor Bodo Manthey for introducing the research topic of this thesis, for our diverse discussions and his helpful feedback and for guiding me through the labyrinth of the final project.

Bibliography

E.H.L. Aarts and J.K. Lenstra. *Local Search in Combinatorial Optimization*. Princeton University Press, 2003. ISBN 9780691115221. URL `https://books.google.nl/books?id=NWghN9G7q9MC`.

David L. Applegate, Robert E. Bixby, Vašek Chvátal, William Cook, Daniel G. Espinoza, Marcos Goycoolea, and Keld Helsgaun. Certification of an optimal TSP tour through 85,900 cities. *Operations Research Letters*, 37(1):11 – 15, 2009. ISSN 0167-6377. doi: http://dx.doi.org/10.1016/j.orl.2008.09.006.

D.L. Applegate, R.E. Bixby, V. Chvátal, and W.J. Cook. *The Traveling Salesman Problem: A Computational Study*. Princeton Series in Applied Mathematics. Princeton University Press, 2011. ISBN 9781400841103. URL `https://books.google.nl/books?id=zfIm94nNqPoC`.

B. Chandra, H. Karloff, and C. Tovey. New results on the old k-opt algorithm for the traveling salesman problem. *SIAM Journal on Computing*, 28(6):1998–2029, 1999.

Georges A Croes. A method for solving traveling-salesman problems. *Operations research*, 6(6): 791–812, 1958.

C. Engels and B. Manthey. Average-case approximation ratio of the 2-opt algorithm for the TSP. *Operations Research Letters*, 37(2):83–84, 2009. doi: 10.1016/j.orl.2008.12.002.

M. Englert, H. Röglin, and B. Vöcking. Worst case and probabilistic analysis of the 2-opt algorithm for the TSP. *Algorithmica*, 68(1):190–264, 2014. doi: 10.1007/s00453-013-9801-4.

Lov K. Grover. Local search and the local structure of NP-complete problems. *Operations Research Letters*, 12(4):235 – 243, 1992. ISSN 0167-6377. doi: 10.1016/0167-6377(92)90049-9.

Michael Held and Richard M. Karp. A dynamic programming approach to sequencing problems. *Journal of the Society for Industrial and Applied Mathematics*, 10(1):196–210, 1962. ISSN 03684245. URL `http://www.jstor.org/stable/2098806`.

David S Johnson and Lyle A McGeoch. The traveling salesman problem: A case study in local optimization. *Local search in combinatorial optimization*, 1:215–310, 1997.

Richard M. Karp. Reducibility among combinatorial problems. In Raymond E. Miller and James W. Thatcher, editors, *Proc. of a Symp. on the Complexity of Computer Computations*, pages 85–103. Plenum Press, 1972.

Gözde Kizilateş and Fidan Nuriyeva. *On the Nearest Neighbor Algorithms for the Traveling Salesman Problem*, pages 111–118. Springer International Publishing, Heidelberg, 2013. ISBN 978-3-319-00951-3. doi: 10.1007/978-3-319-00951-3_11.

M. Künnemann and B. Manthey. Towards understanding the smoothed approximation ratio of the 2-opt heuristic. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 9134:859–871, 2015. doi: 10.1007/978-3-662-47672-7_70.

B. Manthey and R. Veenstra. Smoothed analysis of the 2-opt heuristic for the TSP: Polynomial bounds for gaussian noise. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 8283 LNCS:579–589, 2013. doi: 10.1007/978-3-642-45030-3_54.

D.A. Spielman and S.-H. Teng. Smoothed analysis of algorithms: Why the simplex algorithm usually takes polynomial time. *Journal of the ACM*, 51(3):385–463, 2004. doi: 10.1145/990308.990310.

List of Theorems

1.1 Definition (Traveling Salesman Problem (TSP)) 4

1.2 Definition (Traveling Salesman Problem decision version (TSP-d)) 5

3.5 Theorem (Result for general distribution) 12

3.6 Corollary (Uniform distributions) 13

3.7 Corollary (Standard uniform distribution) 13

4.3 Theorem (Result for the discrete case) 15

5.5 Theorem (Result for the exponential case) 20

A Graphical results for the discrete case

Most of the graphical results for variants of the discrete distribution have been moved this appendix because of the large amount of graphics. The descriptions for what is shown here can be found in Section 6.5.

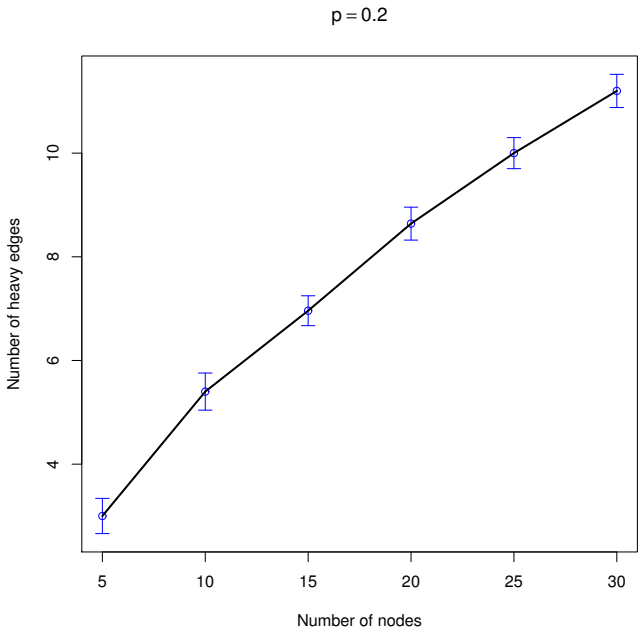

Figure 13: Simulation results for the discrete case ($p = 0.2$) with 95% confidence intervals

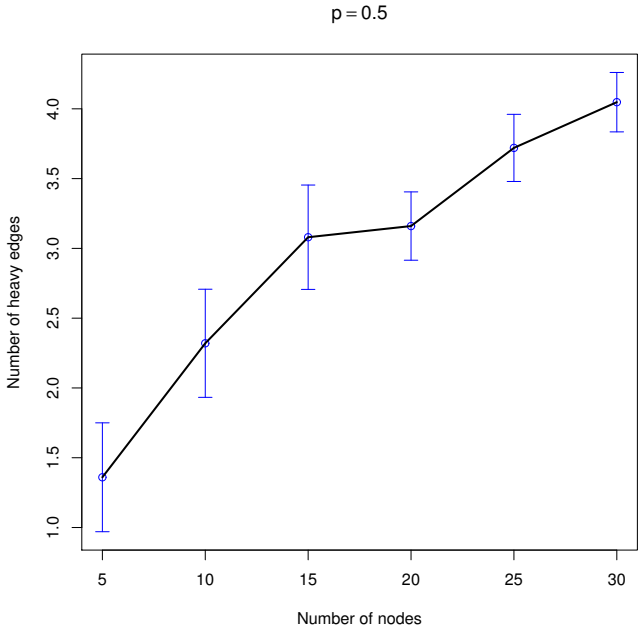

Figure 14: Simulation results for the discrete case ($p = 0.5$) with 95% confidence intervals

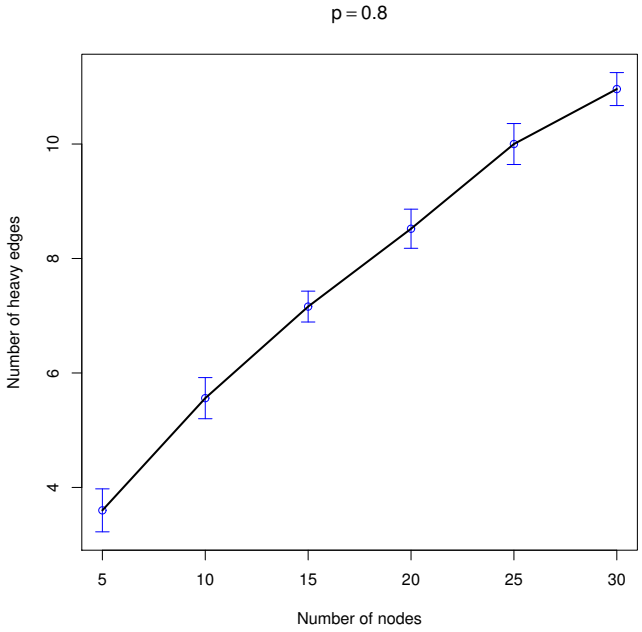

Figure 15: Simulation results for the discrete case ($p = 0.8$) with 95% confidence intervals

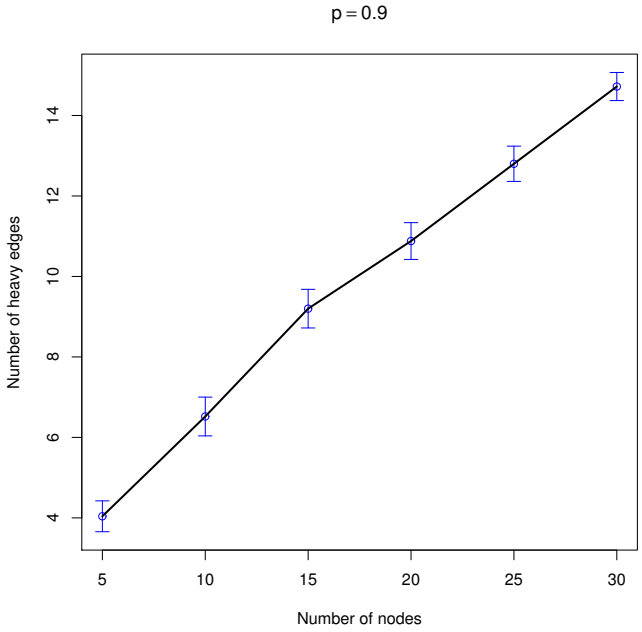

Figure 16: Simulation results for the discrete case ($p = 0.9$) with 95% confidence intervals

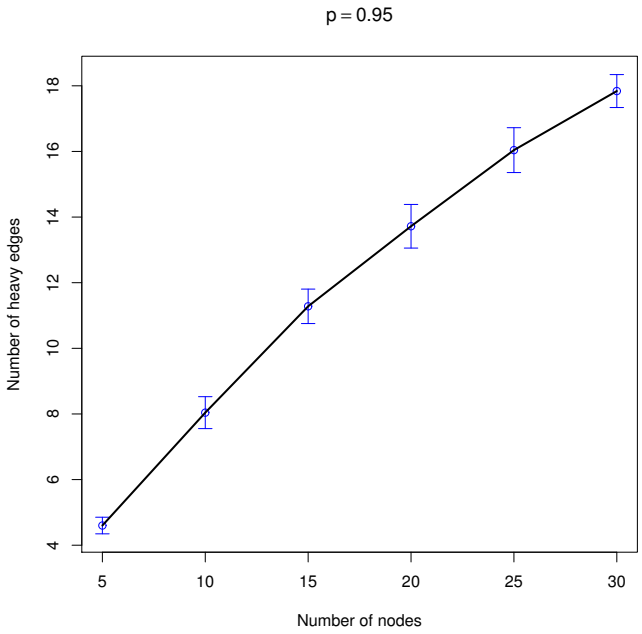

Figure 17: Simulation results for the discrete case ($p = 0.95$) with 95% confidence intervals

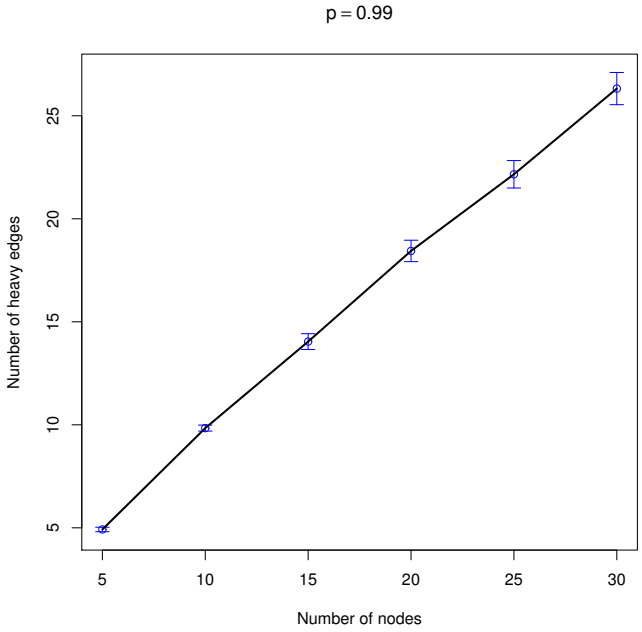

Figure 18: Simulation results for the discrete case ($p = 0.99$) with 95% confidence intervals

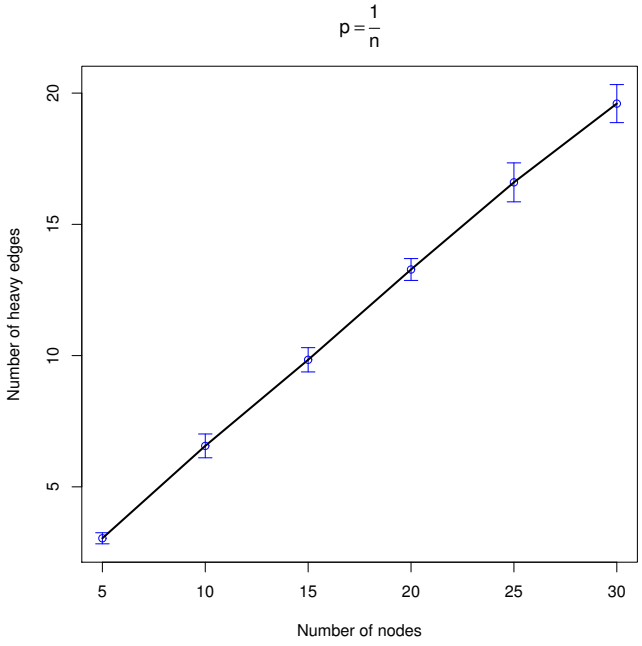

Figure 19: Simulation results for the discrete case ($p = 1/n$) with 95% confidence intervals

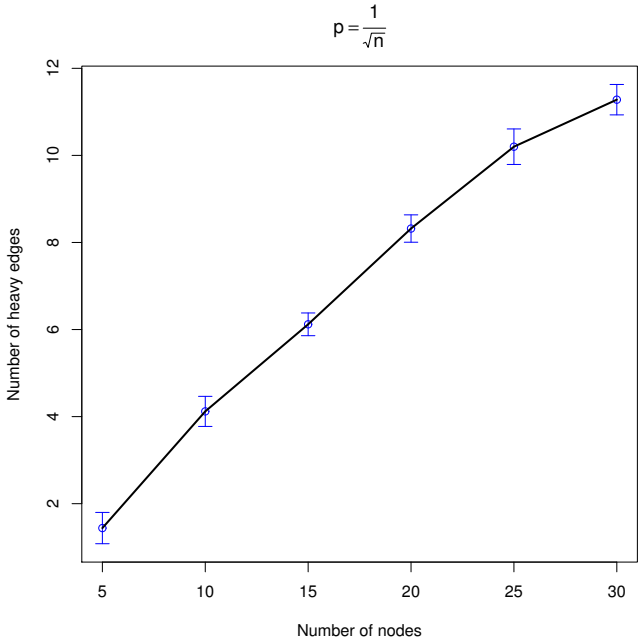

Figure 20: Simulation results for the discrete case ($p = 1/\sqrt{n}$) with 95% confidence intervals

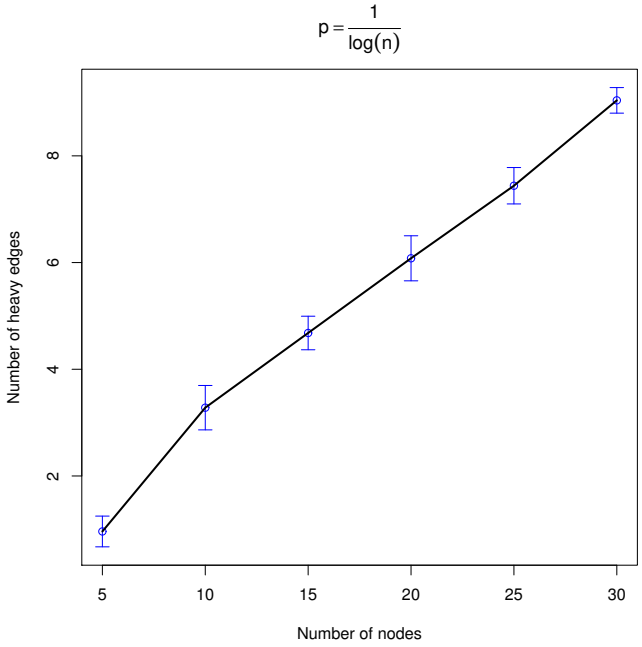

Figure 21: Simulation results for the discrete case ($p = 1/\log(n)$) with 95% confidence intervals

B Simulation data

In this section we present the simulation data from our experiments. For each of the cases and for each number of nodes we used in our experiments we will provided the summarized data. This data consists of the estimated average for the quantity that was measured, the estimated standard deviation and how many runs were performed for that number of nodes. For the uniformly drawn distances and for the exponentially drawn distances this quantity is the ratio of the worst locally optimal solution value to the optimal solution value. For the discrete instances this quantity is the number of edges of weight 2 in the worst locally optimal solution with respect to the 2-opt heuristic. We first show the results for the uniform distances, then the discrete distances and finally the exponential distances.

Uniform on [0,1]

Results for the uniform case

Number of nodes	Estimated average	Estimated standard deviation	Number of runs
5	1.002762061	0.012200828	25
6	1.043333176	0.093769865	25
7	1.088438295	0.112921123	25
8	1.19643585	0.188674616	25
9	1.281828959	0.163432418	25
10	1.381807972	0.205388772	25
11	1.463787693	0.176103551	25
12	1.535149605	0.199928777	25
13	1.693302675	0.274069222	40
14	1.711762586	0.240691278	40
15	1.836924843	0.261228214	25
16	1.924290247	0.345988974	25
17	1.966761186	0.251515322	25
18	2.090280041	0.322149668	25
19	2.144849502	0.23920854	25
20	2.157567109	0.291760846	50
21	2.242299372	0.272612426	25
22	2.323941757	0.257405588	40
23	2.454255232	0.375520725	25
24	2.406278271	0.260228707	25
25	2.668498752	0.324578531	19
26	2.648143386	0.358483189	15
27	2.519533236	0.279337927	7

Discrete distribution

Results for the discrete case with $p = 0.05$

Number of nodes	Estimated average	Estimated standard deviation	Number of runs
5	4.76	0.522812905	25
10	8.36	2.157931108	25
15	11.44	3.663786748	25
20	13.52	2.902872141	25
25	15.96	1.593737745	25
30	18.32	1.62583312	25

Results for the discrete case with $p = 0.1$

Number of nodes	Estimated average	Estimated standard deviation	Number of runs
5	4.08	0.81240384	25
10	6.68	1.215181742	25
15	8.72	1.307669683	25
20	10.76	1.011599394	25
25	12.4	1.0	25
30	14.48	1.084742673	25

Results for the discrete case with $p = 0.2$

Number of nodes	Estimated average	Estimated standard deviation	Number of runs
5	3	0.866025404	25
10	5.4	0.912870929	25
15	6.96	0.734846923	25
20	8.64	0.810349719	25
25	10	0.763762616	25
30	11.2	0.816496581	25

Results for the discrete case with $p = 0.5$

Number of nodes	Estimated average	Estimated standard deviation	Number of runs
5	1.36	0.994987437	25
10	2.32	0.988264472	25
15	3.08	0.953939201	25
20	3.16	0.6244998	25
25	3.72	0.613731755	25
30	4.05	0.497613352	21

Results for the discrete case with $p = 0.8$

Number of nodes	Estimated average	Estimated standard deviation	Number of runs
5	3.6	0.957427108	25
10	5.56	0.916515139	25
15	7.16	0.687992248	25
20	8.52	0.871779789	25
25	10	0.912870929	25
30	10.96	0.734846923	25

Results for the discrete case with $p = 0.9$

Number of nodes	Estimated average	Estimated standard deviation	Number of runs
5	4.04	0.978093383	25
10	6.52	1.228820573	25
15	9.2	1.224744871	25
20	10.88	1.166190379	25
25	12.8	1.118033989	25
30	14.72	0.890692614	25

Results for the discrete case with $p = 0.95$

Number of nodes	Estimated average	Estimated standard deviation	Number of runs
5	4.6	0.645497224	25
10	8.04	1.240967365	25
15	11.28	1.339153962	25
20	13.72	1.696073898	25
25	16.04	1.743559577	25
30	17.84	1.280624847	25

Results for the discrete case with $p = 0.99$

Number of nodes	Estimated average	Estimated standard deviation	Number of runs
5	4.92	0.276887462	25
10	9.84	0.374165739	25
15	14.04	0.978093383	25
20	18.44	1.325393023	25
25	22.16	1.7	25
30	26.32	1.994158135	25

Results for the discrete case with $p = 1/n$

Number of nodes	Estimated average	Estimated standard deviation	Number of runs
5	3.04	0.538516481	25
10	6.56	1.15758369	25
15	9.84	1.178982612	25
20	13.28	1.061445555	25
25	16.6	1.892969449	25
30	19.6	1.848422751	25

Results for the discrete case with $p = 1/\sqrt{n}$

Number of nodes	Estimated average	Estimated standard deviation	Number of runs
5	1.44	0.916515139	25
10	4.12	0.881286938	25
15	6.12	0.665832812	25
20	8.32	0.802080628	25
25	10.2	1.040833	25
30	11.28	0.890692614	25

Results for the discrete case with $p = 1/\log(n)$

Number of nodes	Estimated average	Estimated standard deviation	Number of runs
5	0.96	0.734846923	25
10	3.28	1061445555	25
15	4.68	0.802080628	25
20	6.08	1077032961	25
25	7.44	0.86986589	25
30	9.04	0.611010093	25

Exponential distribution

Results for the exponential case

Number of nodes	Estimated average	Estimated standard deviation	Number of runs
5	1.008228072	0.04046187	25
7	1.227827777	0.347940755	25
9	1.439429887	0.27544366	25
11	1.66761928	0.403903051	25
13	1.818967605	0.231253125	25
15	2.156022617	0.374459465	25
17	2.466399411	0.309579675	25
19	2.531272664	0.389247974	25
21	2.890053216	0.441448422	25
23	3.007487191	0.448693028	25
25	3.244194	0.50100888	25