Assignment 3

Simulated Annealing - Travelling Salesman Problem

Stochastic Simulation

University of Amsterdam

https://github.com/vdesgrange/Stochastic_simulation_project

James Kingsbury (13437542), Viviane Desgrange (13337688)

18th December 2020

Abstract

The travelling salesman problem is one of the most intensely studied problems in optimisation research. Given its ubiquity, it serves as a de-facto benchmark for a range of discrete optimisation methods. When devising new optimisation methods, researchers often look to natural phenomenon for inspiration. (Lihoreau, Chittka and Raine, 2010) analyse the complex navigational strategies of honey bees and suggest that the flower visitation sequences of the bees can be used to approximate a solution to the travelling salesman problem. In the 90s, a researcher at VU Amsterdam used the mechanics of an ant colony to attempt to approximate a solution (Dorigo and Gambardella, 1997). Real ants use pheromones to find the shortest distance from a food source to their nest; Dorigo and Gambardella use a distributed algorithm modelled upon a set of co-operating agents called ANTs to find good solution to TSPs. Similarly, the algorithm we implement is inspired by techniques from metallurgy, in which heating and controlled cooling of a solid is used to increase the size of its crystals and reduce their defects. With this algorithm we achieve good local minimums for each of our three problem sets. We achieved minimums of 431, 2792 and 55416 for the eil51, a280 and pcb442 problems respectively. We also explored various parameter settings for the algorithm, focusing on the cooling mechanism and markov chain length.

1 Theory

1.1 Travelling Salesman Problem

The travelling salesman problem (TSP) was formulated in the 1800's largely by the Irish mathematician and statistician Sir William Rowan Hamilton. Hamilton defined the notion of a Hamiltonian path, a path upon a complex network which visits each node of the graph exactly once with no repeat edges. A Hamiltonian cycle is simply a Hamiltonian path which begins and ends on the same vertex. Hamilton dreamt up numerous puzzles concerning Hamiltonian paths, the most enduring of which is the travelling salesman problem, (Graves, 1882). The travelling salesman problem can be formulated as follows: "Given a list of cities and the distance between each pair of cities, what is the shortest existing route so that a traveler can visit each city exactly once and return to the origin city?". A variant is the "decision version" where given a length, we must find if there is a solution path of shorter length. The most basic version of this problem, which we will detailed as "symmetric" travelling salesman, can be formally represented as a mathematical structure by a complete undirected graph where cities are represented by the graph vertices, the paths between each pair of cities are represented by edges and the distance between each pair of cities are associated to the edge's weight. Moreover this graph is complete, as in the definition of the problem the salesman can travel between every pair of cities. The solution to the original problem can be described as a Hamiltonian cycle in which the edge weights are minimised.
Let's remember a few definitions of the existing complexity class: P is the set of problems that can be resolved in a polynomial time, NP is the set of problems whose solutions can be verified in a polynomial time. NP-hard is the set of problems which are at least as hard as the hardest problem in NP and can be reduce in a polynomial time to an NP-complete problem. A problem is said to be NP-complete if it is both in NP and NP-hard. While the symmetric travelling salesman problem is of simple definition, it appears that his resolution isn't: for a set of n cities, the number of existing Hamiltonian path is $n!$; we remove the first city which does not impact on the solution and given that it is an undirected graph we can divide the number of paths: there are $\frac{1}{2}(n-1)!$ paths to verify. For instance, for 51 cities, it represents $\approx 1.52 \times 10^{64}$ paths.
Therefore, it is not possible on a deterministic Turing machine to resolve this problem nor to verify a given solution in a polynomial time. It appears that the travelling salesman problem belongs to the class of NP-hard problems. The decision based variation of this problem, however, is NP-complete given that its solution can be verified in a polynomial time.
In this paper we will only focus on the original symmetric travelling salesman problem. As we just saw that it is not possible to obtain the exact solution of this problem, the main topic of this paper will focus on existing stochastic methods to obtain an approximation of the solution. If we come back to the mathematical definition of our problem, this means we will investigate a method, the Simulated annealing, to minimise the sum of the weights in the Hamiltonian cycle, in order to get close to the global minimum.

1.2 Simulated annealing

Simulated annealing is a meta-algorithm for global optimisation and more broadly a generalisation of the Hastings-Metropolis method, first described in (Metropolis, 1953). The function of Hastings Metropolis is to create a Markov Chain with limiting probabilities such that we can simulate some probability distribution which is otherwise very tricky to sample from. The simulated annealing method was introduced independently by (Kirkpatrick, Gelatt & Vecchi, 1983) and in (Černý, 1985). Both algorithms are stochastic, generating new points to move to at random. Both algorithms move to a new random point with a certain probability, which is based on the difference (or the ratio) of the current and new proposed point in the search space. Where they differ is in their acceptance/rejection criterion. HM moves to a new point based on the ratio of the current point and a new proposed random point $min(\frac{new}{old}, 1)$. If this ratio is bigger than 1 (the new point has higher likelihood than the current point) then it will move to this new point immediately. Otherwise, if the new point has lower likelihood, the algorithm will move to this point with some probability - based on the ratio. Simulated annealing has an additional parameter (temperature) which scales the difference by a certain amount $exp(\frac{new-old}{t})$. When the temperature is very high the difference won't have any meaningful impact on the decision, so the algorithm will always accept any new point, which means it will move in a random fashion. When the temperature is very low the criterion will evaluate to 0 for worse points, so the algorithm will move deterministically, only accepting solutions with a lower value or a lower path distance in our case.

The general algorithm can be described as such: One starts by constructing a permutation of our cities which forms our initial guess at the minimised path length. At each step of the inner loop, we permute the order of our path in some way. The length of the modified path is then calculated and compared to the path before modification, producing the cost difference. If the cost difference is negative, the change is accepted; if the tour length increases, the change is rejected with some probability, in which case one simply keeps the old tour at that step. Such a stochastic construction of a sequence of states is called a Markov chain. The stochastic accept/reject part is supposed to simulate a random fluctuation due to temperature effects, and the temperature is a parameter which measures the bias towards short tours. If one wants to get to the globally optimal tour, one has to move the temperature down towards zero, corresponding to a strong bias in favor of short tours, this reduction of temperature occurs on each iteration of our outer loop. Thus, one makes the temperature vary with time, and the way this is done is called the annealing schedule, and the result is simulated annealing.

Including a temperature scheme for the Metropolis algorithm is equivalent to the process of annealing in thermodynamics, hence simulated annealing. The probability of finding the system in a particular energy state is given by the Botzmann-Gibbs distribution.

$$p_i \approx e^{-\frac{E_i}{kT}} \approx e^{-\frac{h(X_i)}{T}} \tag{1}$$

The function above is written such that we minimise E and the temperature T is used as a control variable. The function $h(X_i)$ in the function above will be the distance difference between our proposed path and the current path. The simulated annealing algorithm consists of two nested loops: the outer loop and the inner loop. Our outer loop iteratively decreases the temperature which influences the acceptance probability of new states which have a longer path than the current best path. Our inner loop runs Metropolis Monte Carlo at a particular state. Pseudo-code for the algorithm is shown below.

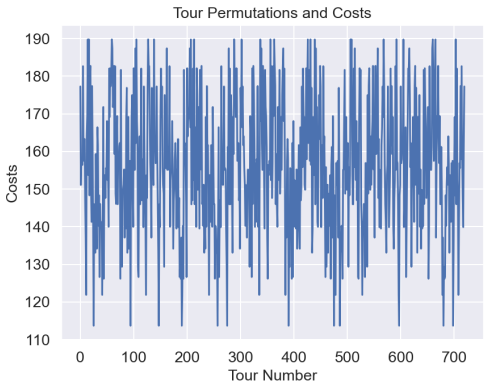
Above, we mention that the proposed new path on each iteration is created by permutation of the path from the previous state of the chain. We do not use the standard 2-opt, but opt for a slightly more elaborate scheme. First of all, a segment of the path is selected, with its start and end cities chosen at random from our current path. Then, we decide with identical probability to either: reverse or transport our path each with a probability of 0.5. If reverse is selected, an alternative path is generated in which the cities in the chosen segment are reversed in order of visit. If transport is selected, the segment is clipped out of its current position in the path and spliced in at a randomly chosen point in the remainder of the path.

In Figure 1b we have generated the costs of all possible tours of a TSP problem with 6 cities, with total possible permutations of $6! = 720$. The tours with steep downward spikes correspond to good local minimum or potentially global minimum. There is a great deal of variation in the cost function, even for a small problem. Thus, it is easy for our algorithm to become trapped in a local minimum.

One problem with simulated annealing is that it can be hard to determine the initial temperature in an optimum way. If the initial temperature is set too high, then we waste computation time searching solutions which are definitely not optimum, if we start the temperature too low, then we will definitely reduce the quality of the search. One other issue with tuning the algorithm is which 'cooling schedule' to use, again this causes a trade-off between computation time and search quality. The question, put simply is: what amount of iterations

```
create initial solution s
set initial temperature t
while s < max_steps
    while c < chain_length
        s' = perturb s
        ΔE = E(s') -E(s)
        if(ΔE <= 0) then
            s = s'
        elseif(rnd(0,1) < exp(-ΔE/t)) then
            s = s'
        end if
        c = c + 1
     end do
     decrease t
     s = s + 1
end do
```

(a) Simulated Annealing Pseudo-code



(b) Permutations and Costs

Figure 1

are sufficient at each temperature? Adjusting our cooling schedule and analysing the results is one way to try to answer this question.

1.3 Cooling schedule

Cooling schedule can be defined has the set composed of starting temperature T_0, the method for lowering temperature and the stop condition for annealing process. The cooling schedule being problem dependant, each components will influence the progress to an optimal solution and so must be be individually investigated.

1.3.1 Initial temperature T_0

The initial temperature T_0 must be carefully set has it will influence acceptance rate of bad configurations of the Metropolis algorithm during creation of the Markov chain (Ben-Ameur, 2003). A general assumption is that in first steps of the algorithm, the acceptance rate of a bad configuration should be close to the rejection rate. To determine this initial temperature, we can logically suppose that the lowering method nor the Markov chain length will have little influence on the acceptance rates of the first few steps. Indeed, given that the acceptance condition of a bad configuration (See Equation 1) depends on the cost difference and the current temperature, in theory, if we want an acceptance rate of bad configuration of at least 50%, we should choose an initial temperature such that:

$$T_0 \approx \frac{E(\Delta c)}{ln(X)} \; with \; X = 0.5 \tag{2}$$

Where Δc is the positive cost difference between the latest and newest configuration (the work from (Ben-Ameur, 2003) and (Johnson et al, 1989) gave us confirmation of this equation). While we expect the cost difference to increase with the number of cities, a simple methodology we can establish to get an approximation of T_0 will be to generate random permutation of the path configuration for each problem studied, store each cost differences and study the different confidence intervals (See Section 2.2.1).

1.3.2 Logarithmic cooling schedule

The following and main component is the function evaluated to cool the temperature in simulated annealing. We focus our attention on the scientific literature (Geman & Geman, 1984), without considering too many details about the demonstration behind it, which suggests that if the temperature T at step n respect the bound:

$$T(n) \geq \frac{C}{log(1+n)} \tag{3}$$

With a constant C independent from n but dependant from the problem studied, then when $n \to \infty$ the simulated annealing algorithm will converge to the global minimum.
An issue behind this formula is the presence of logarithmic function: as we work with a finite number of steps

N, the temperature T will reach a plateau (See Section 2.2.2) and it won't be possible to converge to a global minimum in an acceptable number of steps. For instance, to reach a temperature of 0.1 with $C = 1$, the number of steps n required will be: $T(n) = \frac{1}{log(1+n)} = 0.1 \Rightarrow n = 10^{10} - 1$.

A way to by-pass this limitation, is to perform a preliminary investigation on the early steps of the simulated annealing algorithm in order to determined an initial temperature T_0 (See previous Section 1.3.1) adequate for the studied problem and small enough to allow convergence to the global minimum despite presence of the logarithmic function. We should use T_0 as a value for the constant C (from the Equation 3). We will see later (See Section 2.2.2) that it remains possible to obtain interesting results.

1.3.3 Geometric cooling schedule

Given that we work on a finite time, we must consider other cooling schedule function to come to a fair optimum. The geometric cooling schedule can be considered as a simple and efficient algorithm in practice, as we will be able to observe in the following experiments (See Section 2.1 and 2.2):

$$T(n) = \alpha^n \times T_0, \quad 0 \leq \alpha \leq 1 \tag{4}$$

In comparison to the logarithmic schedule, with $T_0 = 1$, for $T(n) = 0.9^n \times T_0 = 0.1 \Rightarrow n \approx 22$.

While we have resolve the finite time problem, the next interrogation we have is to know if it exists a lower bound to the constant α beyond which our algorithm would get trapped in a local optima; we will try to determine if such a lower bound exists during our experiments.

2 Results

2.1 Local optima

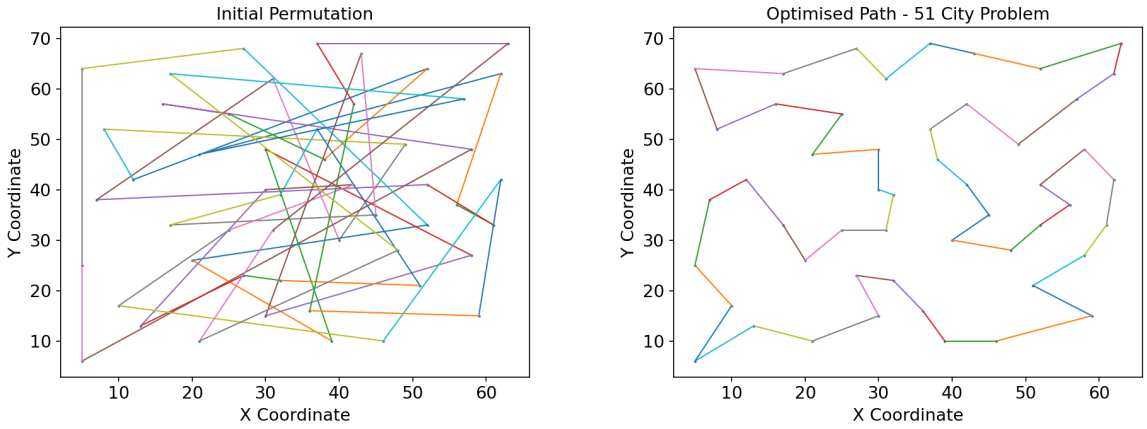
Our first experiment involves attempting to find a global minimum or at least, a very good local minimum for each of our three problem sets. While our method guarantees a convergence upon running sufficiently large (infinite) number of iterations, we of course cannot run infinite solutions. In any case, TSP is a notoriously difficult problem and the solution search space even for a 51 city problem is 51 factorial, so finding a good local minimum would be satisfactory for the scope of this assignment. Note that the a280 data-set contains a duplicated city, city 171 and city 172 have identical co-ordinates. We dealt with this by deleting one of the duplicate cities.

Since our method is inherently stochastic, we need to perform multiple repetitions and analyse the summary statistics to gain a good understanding of the performance. For each of our problems sets, we run 30 simulations and in each simulation record the minimum path length we obtained. We created a sample of our path length and computer summary statistics over 30 simulations. In each simulation, we perform 500 iterations of the outer loop and at each temperature setting we construct a markov chain of 500 states. We start at a temperature of 50 and in each step we decrease the temperature by 10% according to a geometric schedule. The methods to obtain the sample variances and associated confidence intervals were explained in our first assignment, (Kingsbury & Desgrange, 2020).

Problem	Best Minimum	Mean Minimum Path	Sample Variance	95pc Conf Interval
eil51	431.1	439.2	19.4	[437.1, 441.3]
a280	2792.01	3097.2	5092.7	[3063.8, 3130.6]
pcb442	55416.73	57231.5	1262038.1	[55836.6, 58626.4]

Table 1: Simulation Results - 30 Simulations

Table 1 shows the mean minimum path lengths and associated statistics achieved for each of our three problems. Since in all three of our problems the cities are distributed widely over the cartesian coordinate space, it is intuitive that the problems with a greater number of cities have a greater mean path length. We also see increasing sample variance for problems with a larger number of cities. Since, the search space is the factorial of the number of cities, it follows that there are far more local minimums to be caught in with larger problems, leading a greater variance in the obtained path lengths. Since the number of simulations stayed constant for each problem set but the variance increased, the increase in the range of our confidence intervals also makes sense.



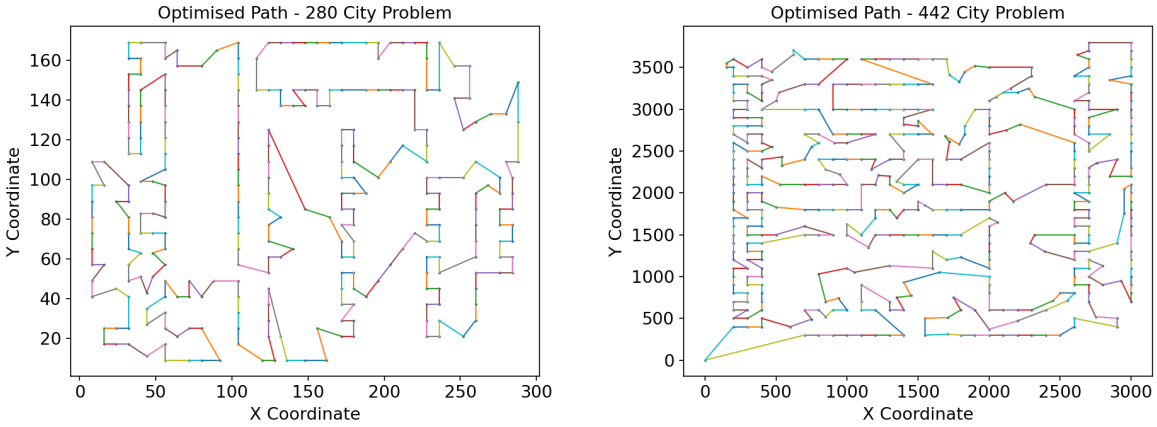
(a) Initial Random Permutation 51 City Problem

(b) Minimised Path 51 City Problem

Figure 2

Figure 2a shows the initial permutation of our 51 city problem, this is the first random tour we generate when starting the algorithm. This order has a total path length of 1636.9. Our best minimised path length for

the 51 city problem is show in Figure 2b, this solution has a path length of 431.1, to achieve this path length we required markov chains 1000 states long and 1000 total steps of our outer loop.



(a) 280 City Problem

(b) 442 City Problem

Figure 3

Figure 3a and Figure 3b show the city permutations associated with our minimum solutions for the 280 city problem and 442 city problem respectively. With large numbers of cities, the search space becomes incredibly large in this problem, meaning that we ran into computational constraints. Adding more states to our markov chains became prohibitively slow after a length of around 1000, in particular when running enough simulations/repetitions to obtain a meaningful sample size. The values for our local optima are shown in Table 1. All of these solutions were obtained by running the algorithm for 1000 iterations, chain length of 1000 and geometric cooling schedule. While all of our solutions are good local optima, it appears that in terms of path length, the difference between our achieved minimum and the global optimum increases as we increase the number of cities.

2.2 Cooling schedule

In the theoretical part (See Section 1.3), we have performed the analytical analysis of the equations used by the simulated annealing algorithm. We highlighted several properties such as an analytical approximation of the optimum initial temperature T_0 and the expected behavior of different cooling schedule functions, but raised as well some interrogations such as the lower bound to the constants used by geometric cooling schedule beyond which our algorithm would be trapped in a local optima. The next steps would be to verify or answer them.

2.2.1 Initial temperature T_0

Before experimenting with various cooling schedule methods, we will verify the initial temperature T_0 interval determined previously, such that the acceptance/rejection rate of a bad configuration would be around 50%. In the early stage of our experiments we got confirmation after running several simulations that the lowering method nor the Markov chain length have a small influence on the acceptance rates of the first few steps. Therefore, to get an estimation of the initial temperature T_0 (See Equation 2), we generated a large set of random permutations (reverse or transport operations used in the simulated annealing graph) and computed the sample mean, variance and confidence interval of the cost difference Λc of each problems (See Table 2 and Figure 4).

In order to verify the results obtained in the Table 2, we then observed the early steps of the simulated annealing algorithm with a set of initial temperature T_0 close to these theoretical values. We selected the geometric cooling schedule of the form $T(n) = .9^n T_0$ previously used in Section 2.1, we set the Markov chain length to a large constant (See Section 2.3), and focused our attention on the first steps of the simulated annealing algorithm.

While the mean cost difference and initial temperature appears to be intimately linked (See Figure 5a); in the Figure 5b (See Appendix A.1 for the Problems eil51 and pcb442), it appears that our approximations of the optimal T_0 are close to the optimal values and we obtained the required acceptance rates. A few adjustments

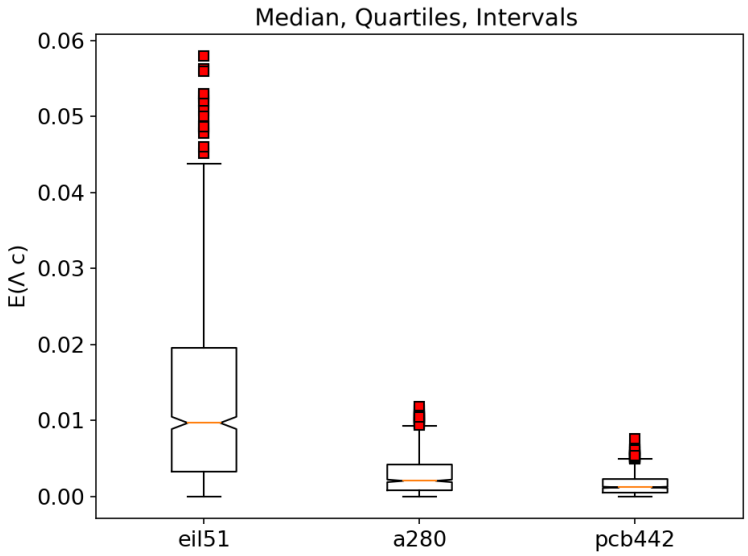


Figure 4: Notched box plot of cost difference for eil51, a280 and pcb442 problems.

Problem	E(Λc)	Var(Λc)	95% Conf Interval	$\hat{T_0}$
eil51	≈ 0.0128	$\approx 1.3 \times 10^{-4}$	$[1.21 \times 10^{-2}, 1.35 \times 10^{-2}]$	$[1.74 \times 10^{-2}, 1.95 \times 10^{-2}]$
a280	≈ 0.0028	$\approx 6.1 \times 10^{-6}$	$[2.6 \times 10^{-3}, 2.9 \times 10^{-3}]$	$[3.75 \times 10^{-3}, 4.18 \times 10^{-3}]$
pcb442	≈ 0.0016	$\approx 1.9 \times 10^{-6}$	$[1.5 \times 10^{-3}, 1.7 \times 10^{-3}]$	$[2.16 \times 10^{-3}, 2.45 \times 10^{-3}]$

Table 2: Estimation of optimal initial temperature $T_0 = -\frac{E(\Lambda c)}{ln(0.5)}$, based on estimated cost difference Λc, for problems eil51, a280 and pcb442.

were required to get an ideal acceptance rate, however it could be due to the selection of new configurations, which was not used when doing random permutation.

2.2.2 Logarithmic cooling schedule

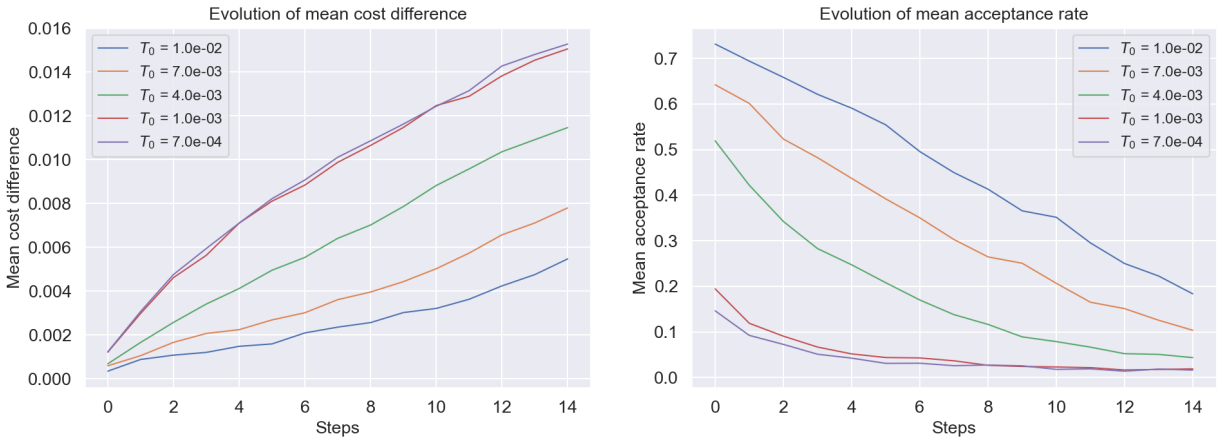
Now that we understand how to select the initial temperature T_0 to start the simulated annealing algorithm, we can investigate the main component, the cooling schedule.
As our method is stochastic, we have to run multiples simulations and analyse the statistics from the results to get an accurate comprehension of the influence of lowering method on convergence. Our experiments were perform on the a280 problem, with 30 simulations and a fixed Markov chain of length equal to 250, a value large enough to consider accurate results (See Section 2.3).

The Figures 6a and 6b introduce the effect of the logarithmic and geometric cooling schedule on the convergence to the global minimum.

We will investigate the logarithmic cooling schedule (See Section 1.3):

$$T(n) = \frac{C}{log(1+n)} \quad with \quad C = T_0 \tag{5}$$

As we know, a major issue with the logarithmic cooling schedule is that the number of steps required to reach the global minimum will follow an exponential function (See Section 1.3).
In our first experiment, we fixed a non-adequate initial temperature $T_0 = 8 \times 10^{-3}$ and a maximal number of steps $N = 400$. We are able to observe in the Figure 6b that the minimal path distance reached a plateau, the algorithm is not able to converge to a global minimal optima. Logically, $T(400) = \frac{8 \times 10^{-3}}{log(401)} \approx 3.0 \times 10^{-3}$, a temperature with an acceptance rate of 40% (See Section 2.2.1) which explains the algorithm is not able to



(a) Evolution of the cost difference for different initial temperatures T_0 in 280 cities problem.

(b) Evolution of the acceptance rate of bad configuration for different initial temperatures T_0 in 280 cities problem.

Figure 5: While Λc and T_0 are intimately linked variables, it appears that the consideration of the magnitude of Λc to set T_0 helps to get an adequate acceptance rate.

converge to the global minimum (See Table 3).

To add additional steps will not resolve the issue, but it is possible to by-pass it by performing a preliminary investigation on the early steps of the algorithm, and determined a T_0 small enough (See Section 2.2.1). In the case where we fixed $T_0 = 4 \times 10^{-3}$, we obtain $T(400) = \frac{4\times10^{-3}}{log(401)} \approx 1.5 \times 10^{-3}$ a temperature with an acceptance rate of 20% which is small enough to get more accurate results. (See Table 3 and Figure 6a).

Initial temperature T_0	Mean Minimum Path	Sample Variance	95pc Confidence Interval
4×10^{-3}	3900.68	12117.7	[3859.57, 3941.78]
8×10^{-3}	20843.7	1.05×10^6	[20461.02, 21226.46]

Table 3: Convergence to a local optimum in problem a280 using a logarithmic cooling schedule.

2.2.3 Geometric cooling schedule

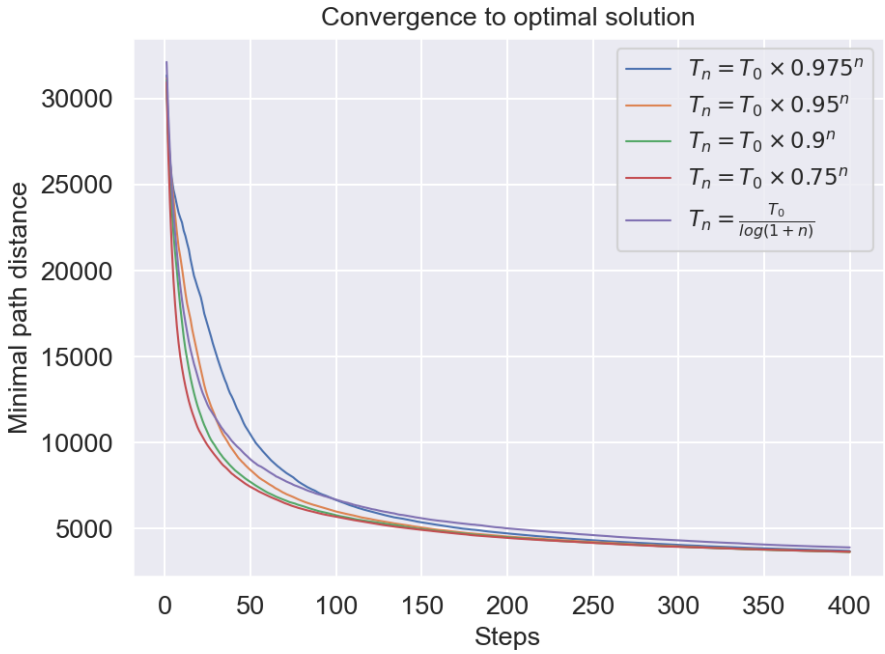
As we work on finite time, we must consider other types of cooling schedule to come to a fair optimum. The geometric cooling schedule can be considered as a simple and efficient algorithm in practice (See Section 1.3), as we observed during our first experiment to retrieve a local optimum (See Section 2.1):

$$T(n) = \alpha^n T_0 \tag{6}$$

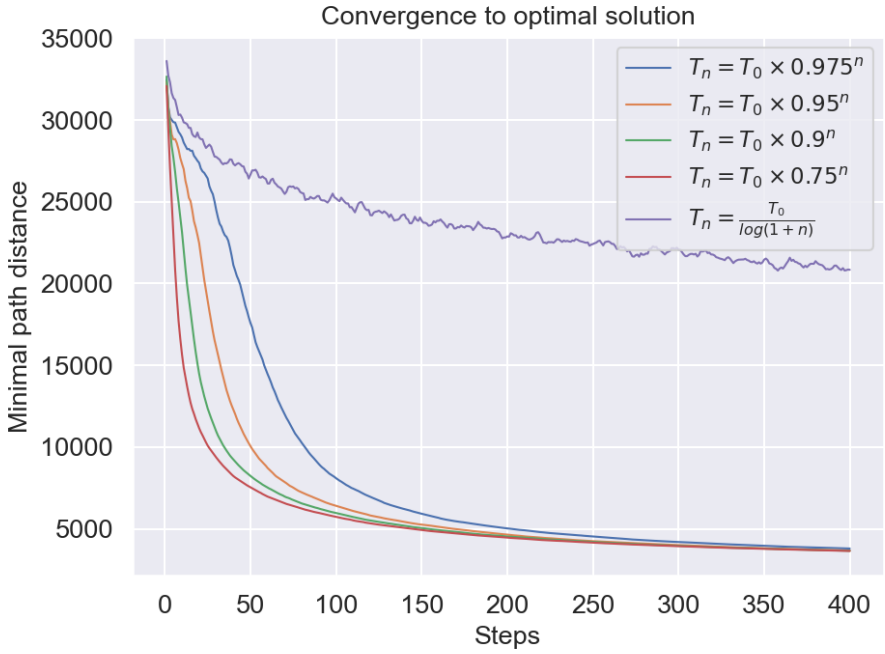
The Figures 6a and 6b, show the evolution of acceptance rates according to the initial temperatures $T_0 = 8 \times 10^{-3}$ and $T_0 = 4 \times 10^{-3}$. While the logarithmic cooling schedule reach a plateau in one of the case, the geometric cooling schedule is able in every case to converge to a global minimum.

We can see that a lower constant α will help converge faster to a global minimum. Given this information, we wonder if there is any reason to not directly provide a small value of α and use from the beginning a geometric cooling schedule instead of a logarithmic cooling schedule ? An assumption we have would that we might get stuck in a local optima. Therefore, we decided to set up an experiment with low values of *alpha* to see the behavior of the simulated annealing algorithm.

We observe in the Table 4 that low values of α, which force a fast drop of the temperature T, give worse results than bigger values of α (See Figure 6a): the algorithm get stuck in a local optimum. It confirms the idea that we can't simply give a low value of initial temperature T_0 nor use a cooling schedule too fast.



(a) Evolution of convergence to an optimal solution for geometric and logarithmic cooling schedule with an initial temperature $T_0 = 4 \times 10^{-3}$. With a preliminary investigation of the data generated from simulated annealing on early steps of the TSP problem to obtain an accurate initial temperature, it is possible to converge to an optimal solution with every cooling schedule.

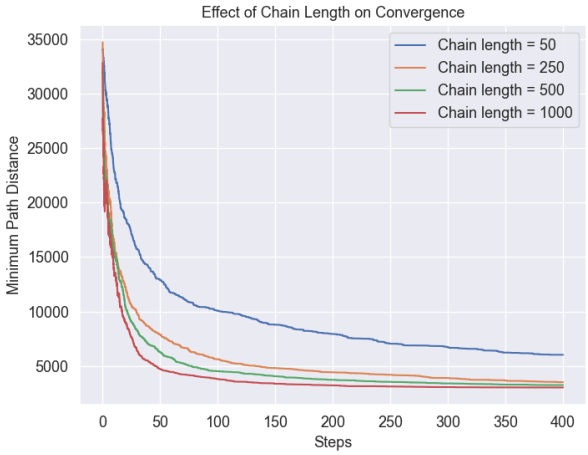


(b) Evolution of convergence to an optimal solution for geometric and logarithmic cooling schedule with an initial temperature set at $T_0 = 8 \times 10^{-3}$. Depending of the class of cooling schedule selected, an invalid initial temperature $T0$ will exacerbate a cooling schedule function whose reflect badly on the convergence to an optimal solution.

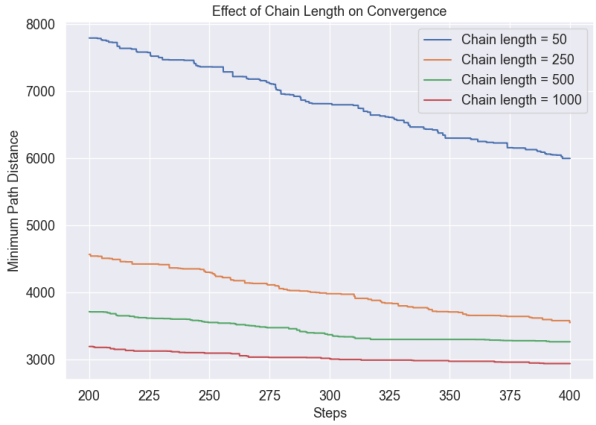
Figure 6: Evolution of convergence to an optimal solution for geometric and logarithmic cooling schedule.

α	Mean Minimum Path	Sample Variance	95pc Confidence Interval
0.975	3699.46	6942.92	[3668.34, 3730.57]
0.95	3636.05	6064.35	[3606.97, 3665.12]
0.9	3640.08	7704.65	[3607.30, 3672.85]
0.75	3649.46	6450.11	[3619.46, 3679.44]
0.4	4188.57	15330.2	[4142.34, 4234.80]
0.3	4188.94	10572.3	[4150.54, 4227.33]
0.2	4217.7	9819.29	[4180.69, 4254.69]
0.1	4203.46	9259.43	[4167.53, 4239.39]
0.05	4227.91	6981.77	[4196.71, 4259.11]

Table 4: Convergence to a local optimum in problem a280, using geometric cooling schedule. A simulated annealing algorithm using a geometric cooling schedule reducing the temperature too quickly will get trap in a local optimum.



(a) Effect of Chain Length - Steps 0 to 400



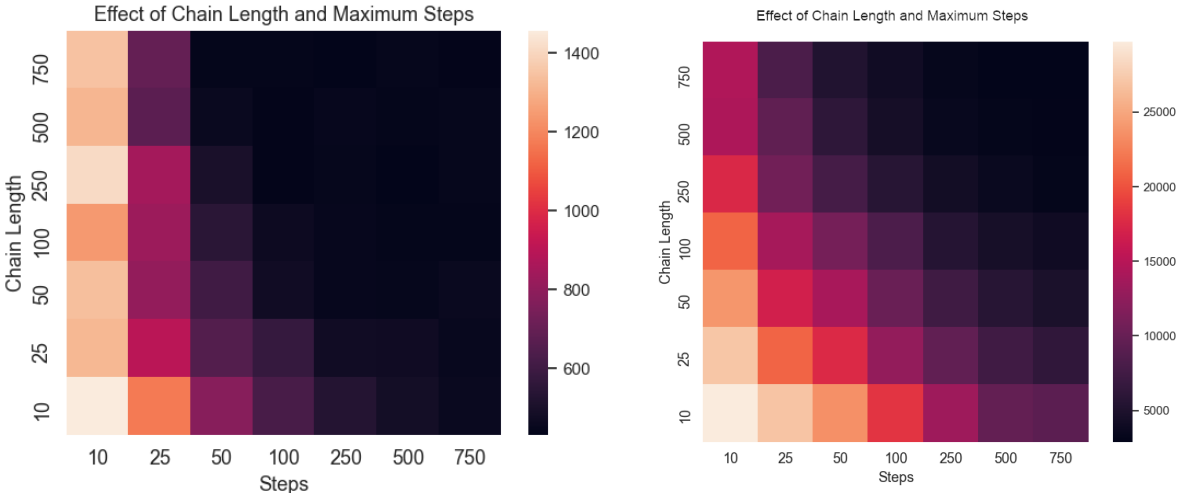
(b) Effect of Chain Length - Steps 200 to 400

Figure 7

2.3 Effect of Chain Length

We will now investigate how varying the length of our markov chains impacts the convergence rate of our annealing process. In principle, to find our asymptotic distribution we require a markov chain of infinite length. Then, as $T \to 0$, the annealing algorithm has to create infinitely many of these infinite chains to guarantee that the global minimum path length is reached. In practice, none of this is practical and we must set a fixed length of the chain to be constructed within each step. We must repeat the annealing algorithm with different initial route permutations (and different random numbers) on each run. We can then say with some level of certainty that we have found the global minimum, or at least a good local minimum. Sometimes, authors try to use specific schemes to dynamically determine the optimum length of a chain for a given temperature. In (Javidrad, Nazari & Hamidreza, 2018), the authors propose a method where they tune the chain length per iteration according to variance in the objective function value. Meaning that once the solution settles into an optimum for some temperature level and the variance of the objective function is below some margin, they decide the chain length is enough. However, in most practical settings the chain length is predetermined experimentally and considered constant throughout the simulation. That is the approach we will take here. The dynamics of chain length are also entangled with the temperature schedule. If at a particular step, our probability distribution approaches that of the stationary asymptotic distribution, then a large decrease in our temperature will move us far away from this stationary distribution and we will require a very long markov chain to bring us back. However, if the temperature change is only small, then we are still reasonably close to the stationary distribution and chain of small length will be sufficient to get us back to the asymtotic position. Thus, the magnitude of the changes in our temperature at each step may influence the markov chain length required for a reasonably good path length solution.

Our experiment is conducted over 30 simulations, with chain lengths of 50, 250, 500 and 1000. For each chain length we perform 400 steps of the outer loop with varying temperature levels. We concentrate our analysis on the a280 problem here. Figure 7a shows the convergence of various annealing algorithms with varying chain lengths. In general, longer chain lengths lead to better estimates for our minimum path, especially at lower steps and thus higher temperature values. Longer chains lead to quicker convergence to a lower minimum (as measured in steps). This is intuitive: a longer chain length gives our algorithm the opportunity to explore a greater variety of candidate paths for a given temperature setting. However, it does appear there is decreasing returns to chain length. Moving from a chain length of 500 to a length of 1000 improves convergence but not as much as increasing the length from 250 to 500. Adding one state to a chain which is already 1000 links long will have smaller effect on convergence than adding a state to a chain with 50 links. Figure 7b shows the same data, but with a zoomed view to highlight the part of the convergence graph at higher values of steps and thus lower temperature. We want to highlight that, as we increase steps, it appears that the distance between the various solutions for the different chain lengths decreases. So, the difference in minimum solution length is less pronounced for higher step values. To some extent, we might be able to offset a decrease in chain length with more iterations of our outer loop.



(a) Heat Map of Minimum Distance, eil51

(b) Heat Map of Minimum Distance, a280

Figure 8

To investigate the relative gains in solution accuracy we can achieve by increasing outer loop iterations versus the accuracy gains achieved by increasing chain length, we run another experiment on the a280 problem. This time, we run 30 simulations with different combination of step counts and chain lengths. The temperature is decreased by 10% after each step. The results of our simulations are shown in the form of a heat map in Figure 8a and Figure 8b. Our scale colouring shows the average minimum path distance achieved with each combination of parameter settings, darker shades indicate a lower minimum path length. The best solutions for our problem are found in segments with both high steps and a high chain length and the worst solutions are found in the segments with the lowest steps and the lowest chain lengths. Interestingly, we seem to obtain better estimates for very low chain lengths and high steps than we do for very low steps and high chain length. When comparing the two heatmaps, we see that for the larger problem, it tends to require a larger number of steps and larger chain length to achieve relatively good solutions.

To say with some degree of certainty if longer markov chain lengths increase our mean minimum path length(at higher temperatures), we will perform a hypothesis test on the different chain lengths. We can use the two-sample parametric Welch test to determine whether there is a significant difference in the convergence to the minimum between the two chain lengths. Welch's test is used to determine if two population means are equal. Welch's test is an adaption of the student's t-test, except that it has a high level of reliability when two samples have different levels of variance. Our null hypothesis is that the minimum path length of a 250 state chain is not statistically different from the minimum path length of a 1000 state chain.

Chain Length	Mean Minimum Path	Sample Variance
250	3611	8681
500	3175	4260
1000	2968	2472

Table 5

$$t = \frac{\overline{X}_1 - \overline{X}_2}{\sqrt{\frac{s_1^2}{N_1} + \frac{s_2^2}{N_2}}} = \frac{3175 - 2968}{\sqrt{\frac{4260}{30} + \frac{2472}{30}}} = 13.8 \tag{7}$$

Where $\overline{X}_1$ is the mean minimum path distance from our algorithm with a chain length of 500, and $\overline{X}_2$ is the equivalent with a chain length of 1000. s_1 and s_2 denote the sample standard deviation for our sample minimums for each chain length. N_1 and N_2 denotes the number of data points, which in our case is the same for each of our two samples. We computed our t statistic with the Scipy.stats.ttest_ind() function. The critical value to which we compare our test statistic is $T_f^{-1}(\frac{p+1}{2})$ where f is the degrees of freedom as described by the Welch–Satterthwaite equation. The absolute value of our test statistic is clearly larger than the critical value, so we reject the null hypothesis that the chain length increase has no effect on our mean minimum path length. We can say that with statistical significance, increasing the number of chains from 500 to 1000 does lead to better path solutions.

Conclusion

Throughout this paper we have explored the use of the simulated annealing methods to optimise solutions to a discrete minimisation problem in combinatorics. Given the challenging nature of optimising a function within such a large search space, our method has allowed us to obtain very good results, even with relatively low computing resources. In part 1 of the paper, we found great local optima for all 3 of our problems. We also obtained summary statistics for our estimates, suggesting that while our method does find good minimums the majority of the time, it also has the potential to become stuck in sub-optimal local minima as suggested by large sample variance in the best minimums over a series of repetitions.

In the second part of the paper, we confirmed the expected behavior we got from the analytical analysis of the equations used in simulated annealing, that is to say the influence of initial temperature on the acceptance criteria as well as the behavior of geometrical and logarithmic cooling schedule. We saw that logarithmic cooling schedule was not in practice a suitable solution, while the geometric cooling schedule was a simple and efficient method. We also confirmed the importance of the initial temperature.

In the final part of the paper, we are able to confirm that increasing the length of the markov chains in our simulation does indeed improve the quality of our optimised paths. We also see that adding additional length to our chains has diminishing returns and that increased chain length has causes greater marginal performance increase at higher temperatures than at lower temperatures.

Given more time to investigate the problem, we would be particularly interested in exploring the use of the ant colony method described in (Dorigo and Gambardella, 1997). The authors claim that this algorithm outperforms both simulated annealing and other nature-inspired discrete optimisation routines.

References

1. V. Černý. A thermodynamical approach to the travelling salesman problem: an efficient simulation algorithm. Journal of Optimization Theory and Applications, 45:41-51, 1985

2. Dorigo, M., and L. M. Gambardella. "Ant Colony System: A Cooperative Learning Approach to the travelling Salesman Problem." IEEE Transactions on Evolutionary Computation, vol. 1, no. 1, 1997, pp. 53–66.

3. Graves (RP), 1882. Life of Sir William Rowan Hamilton. Hodges.

4. Javidrad, Farhad & Nazari, M. & Javidrad, Hamidreza. (2018). A variable Markov chain length strategy for improving simulated annealing convergence behavior: An experimental verification.

5. Lihoreau, M., Chittka, L. and Raine, N.E., 2010. Travel optimization by foraging bumblebees through readjustments of traplines after discovery of new feeding locations. The American Naturalist, 176(6), pp.744-757.

6. S. Kirkpatrick, C. D. Gelatt and M. P. Vecchi. Optimization by Simulated Annealing, Science, 220(4598): 671-680, 1983

7. Metropolis, N., Rosenbluth, A.W., Rosenbluth, M.N., Teller, A.H. and Teller, E., 1953. Equation of state calculations by fast computing machines. The journal of chemical physics, 21(6), pp.1087-1092.

8. S. P. Brooks, N. Friel and R. King. Classical Model Selection via Simulated Annealing. Journal of the Royal Statistical Society. Series B (Statistical Methodology), 2003, Vol. 65, No. 2 (2003), pp. 503-520

9. S. Geman, D. Geman. Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images. IEEE Transation on pattern analysis and machine intelligence, vol.PAMI-6, No.6, November 1984.

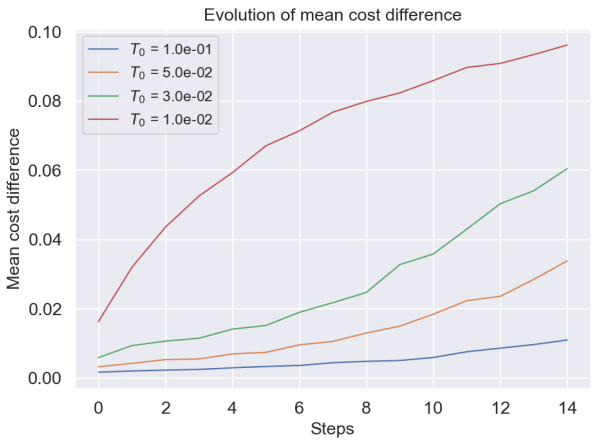
10. W. Ben-Ameur. Computing the initial temperature of simulated annealing. Computational Optimization and Applications 29, no. 3 (2004): 369-385

11. D.S. Johnson, C.R. Aragon, L.A. McGeoch, and C. Schevon, Optimization by Simulated Annealing: An Experimental Evaluation. Part I, Graph Partitioning. Operations Research, vol. 37, pp, 865–892, 1989.

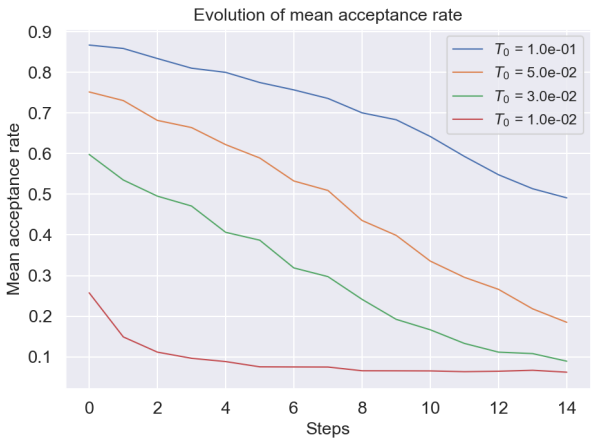
Appendices

A Experiments - Cooling schedule

A.1 Initial temperature T_0

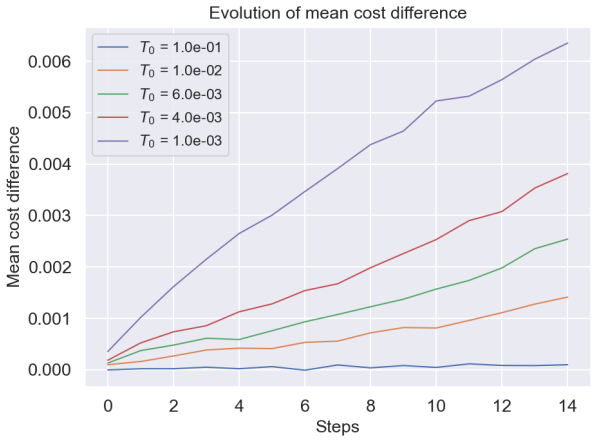


(a) Evolution of the cost difference for different initial temperatures T_0 in 51 cities problem.

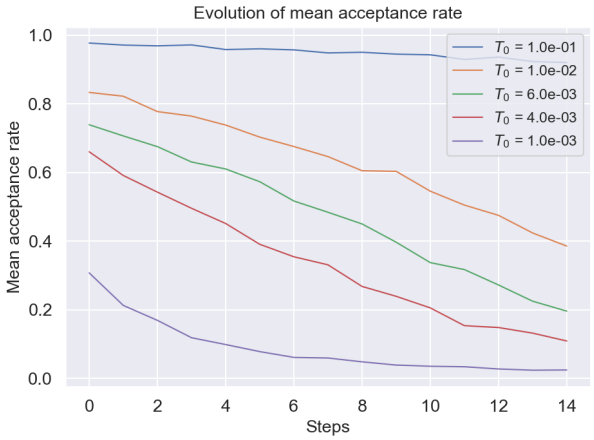


(b) Evolution of the acceptance rate of bad configuration for different initial temperatures T_0 in 51 cities problem.

Figure 9: Problem eil51



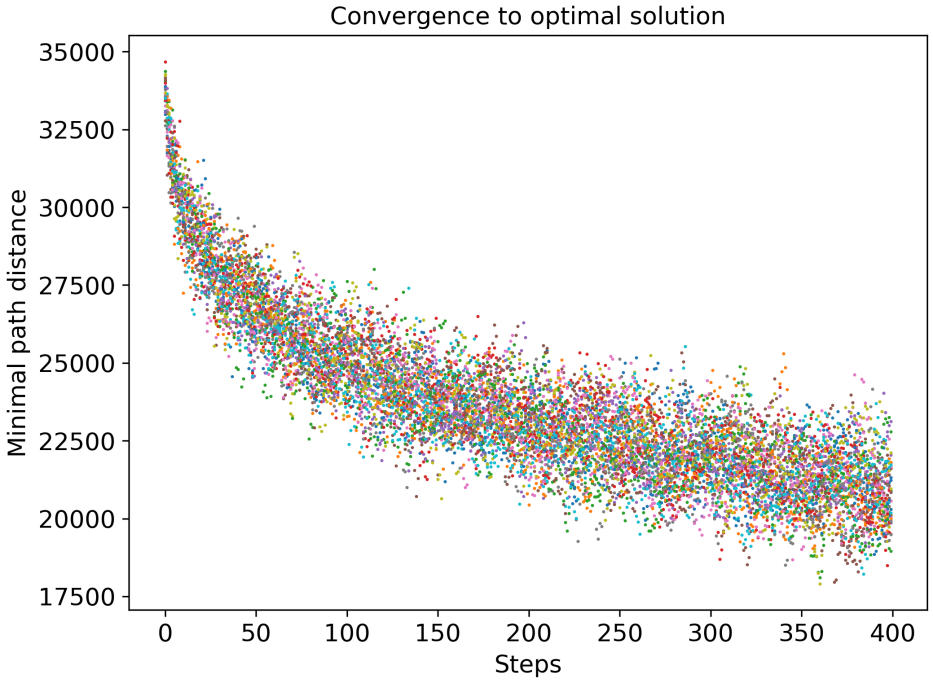
(a) Evolution of the cost difference for different initial temperatures T_0 in 442 cities problem.



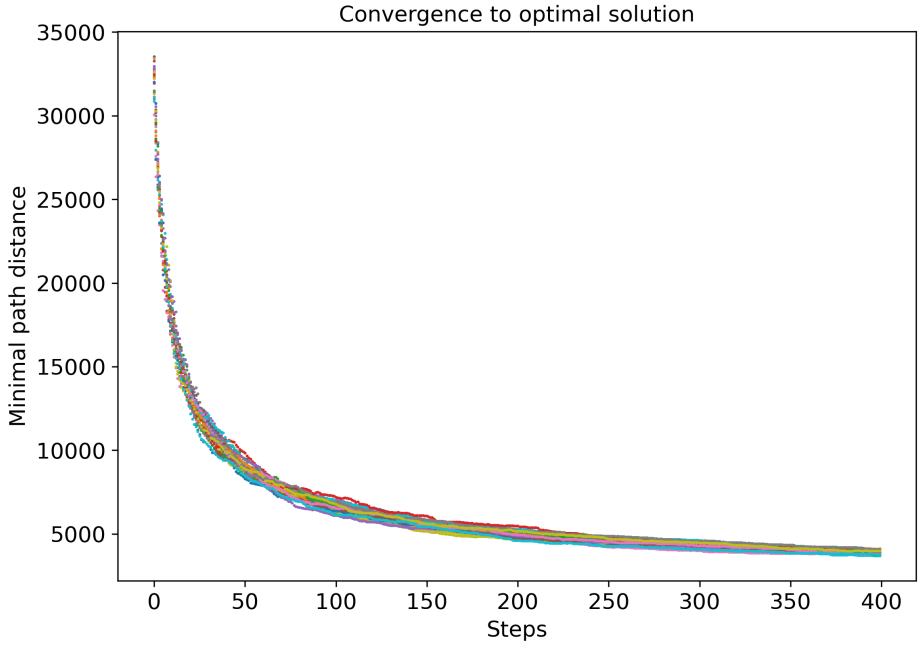
(b) Evolution of the acceptance rate of bad configuration for different initial temperatures T_0 in 442 cities problem.

Figure 10: Problem pcb442

A.2 Logarithmic cooling schedule



(a) Evolution of the path distance with 30 simulations of simulated annealing algorithm using logarithmic cooling schedule with $T_0 = 8 \times 10^{-3}$ to resolve problem 280



(b) Evolution of the path distance with 30 simulations of simulated annealing algorithm using logarithmic cooling schedule with $T_0 = 4 \times 10^{-3}$ to resolve problem 280

Figure 11: Evolution of the path distance with 30 simulations of simulated annealing algorithm using logarithmic cooling schedule