Steiner tree problem

Antoine Huchet

February 12, 2018

Contents

1 Problem description

We are going to study the Steiner tree problem, which is defined as follows: Given a graph $G = (V, E)$, a weight function $w : e \mapsto \mathbb{R}$ assigning a weight w(e) for every $e \in V$ and a set of terminal nodes $T \subset V$. The goal is to find a subset of edges $M \subset E$ of minimal weight that connects all terminals in G.

This problem is NP-COMPLETE. Here we are going to be presenting and comparing different heuristic approaches to the problem.

2 Definition

2.1 Checking admissibility

Definition 2.1. Since a solution needs to connect all terminal nodes, I will define a solution to be admissible when all terminal nodes $t \in T$ are in the same connected component.

2.2 Gain function

Definition 2.2. My gain function will simply be the sum of all edges plus the diameter of the graph.

The diameter being the maximum eccentricity over all vertices. The eccentricity of a vertex being the distance (shortest path over the weighted edges) from that vertex to the furthest away vertex.

Remark 2.3. Note that the leaves of an optimal solution will necessarily be terminals.

3 Overview

The basic framework I will use is the one seen in class.

- First, I will initialise a solution from a known approximation algorithm.
- Then I will generate new solutions by considering variations of my current solutions.
- Finally I will select a few solutions from which I will iterate, optimizing the gain function.
- I will stop after a fixed amount of iterations.

- λ being the number of variations I am considering.
- μ being the number of solutions I am selecting.

Algorithm 1: GENERIC_FRAMEWORK (λ, μ, VARIATION, SELECTION, k, G, T)

Sample μ first feasible solutions $x_1, \ldots, x_\mu$ based on a known approximation algorithm.
while *number_iterations* $< k$ **do**
  $\{x_1, \ldots, x_\lambda\}$ = VARIATION(T, $\{x_1, \ldots, x_\mu\}$, λ)
  $\{x_1, \ldots, x_\mu\}$ = SELECTION($\{x_1, \ldots, x_{\lambda+\mu}\}$, μ)
return x that minimizes the gain function.

- VARIATION being a function used to consider variations of the current solutions.
- SELECTION being a function used to select μ solutions from the current solutions.
- k is a simple integer, G our graph and T our terminal nodes.

4 Initialisation

The problem will be initialised by the following 2-apprximation algorithm.

For all pair of terminals, compute the shortest path between all pair of terminal vertices. Then return the minimum spanning tree of this subgraph.

This algorithm is clearly correct as terminal nodes are never disconnected from the connected component.

Remark 4.1. I will admit that it is a 2-approximation as I don't want my report to be 20 pages and all that really matters is that it yields an admissible solution.

5 Variation

Here I will present the different ways of generating a new solution based on already computed solutions that I studied.

5.1 Mutation

I will refer as mutation all variations that modify a solution by only changing it a little bit.

5.1.1 Adding an edge

A straight-forward mutation that could be applied to an admissible solution would be adding an edge. The edge should have one end in the current solution as we have no interest in creating another connected component. The solution

will obviously stay admissible as no terminal nodes can be disconnected from that operation.

This operation sounds quite limited as it doesn't change the solution much. Therefore, we might want to modify the solution more to bypass those potential local optimums.

5.1.2 Adding a path

Another mutation that would be worth considering is adding a whole path to our current solution. In order to do so, we would randomly select two nodes from out curent solution then add a shortest path based on the graph G we started with.

This mutation is interesting as it changes the solution significantly. Solutions look better than the previous solution.

5.1.3 Clean the solution

After either of the previous two mutations, it is interesting to remove as many unnecessary edges as possible. An unnecessary edge is an edge that when removed doesn't make the solution unfeasible.

Unnecessary edges are removed in a random order.

This operation is more time consuming as it would entail checking if the solution is still feasible for each edge.

5.1.4 Multiple mutation

Definition 5.1 (Multiple mutation)**.** With a probability p add an edge, with probability $1 - p$ add a path then clean the solution.

5.2 Crossover

I will refer as crossover a variation that merges from two previous solutions.

The crossover mutation I will consider is merging two solutions $S_1 = (V_1, E_1)$ and $S_2 = (V_2, E_2)$ into $S = (V_1 \cup V_2, E_1 \cup E_2)$ then cleaning it as explained above.

5.3 Combining both

Another variation that I will consider is a mix of both a mutation and a crossover. Depending on how many offsprings we request, I will generate a multiple mutation then a crossover then a crossover on a multiple mutation.

This yields good variations of our current set of solutions.

6 Selection

In this section I will be presenting different ways of selecting solutions.

6.1 Elitist selection

The elitist selection will sample the best μ solutions out of the $\lambda + \mu$ solutions.

6.2 Elitist selection on offsprings only

The elitist selection on offsprings only will sample the μ best solutions out of the λ offpsrings.

It could be a problem when $\lambda < \mu$. If this happens, it will fall back to the previous elitist selection.

6.3 Fitness proportional selection

The fitness proportional selection samples μ solutions with probability proportional to their gain such that the solution that is the closest to the optimum has the highest probability of being chosen.

From my experience, this doesn't work very well as our solutions tend to have a relatively high gain (a few thousands) and aren't very far from each other (a few hundreds). Therefore the probability vector that I compute is close to uniform.

A workaround could be to compute the probability vector based only on the distance to the current best solution so that the probabilities aren't so close to uniform.

6.4 Boltzmann selection

The Boltzmann selection samples μ solutions out of the λ offsprings. It selects an offspring x if it is better than the best previous solutions y. If it is worse, it selects it with probability $e^{\frac{gain(x)-gain(y)}{T}}$. For some value of T.

6.5 Threshold selection

The Threshold selection samples μ solutions out of the λ offsprings. It selects an offspring x if it is better than the best previous solutions y. If it is worse, it selects it with probability $gain(x) > gain(y) + T$. For some value of T.

In my experiments, T is constant. It would be a good idea to have it follow a decreasing sequence of the form e^{-x}.

7 Comparison

Here I will be comparing the variation and selection strategies.

7.1 Variations

In figure 1, as predicted, the fitness proportional is too uniform therefore doesn't get neither better nor worse. It looks like the threshold variation isn't converging

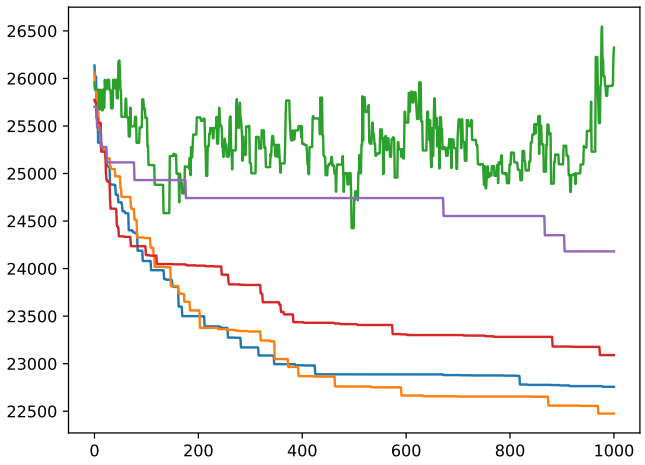

Figure 1: Comparaison for $\lambda = 5$, $\mu = 2$ and mutation variation of classic elitist selection (in blue), elitist selection on offsprings (in orange), fitness proportional (in green), Boltzmann with constant $T = 1000$ (in red) and Threshold selection with constant parameter $T = -150$ (in purple)

very quickly either. Maybe my T parameter was a little too big.
In figure 2, except for the fitness proportional selection, it looks like we are converging rather quickly to a not so great solution.
In figure 3, we observe that combining both variations yields interesting results.

7.2 Selections

In figure 4 and figure 5, we observe that the mutation variation and the mix of both variation and crossover yield interesting results. Crossover doesn't look as good.
In figure 6, we can see that as predicted, my fitness proportional selection stays quite uniform and non interesting.
In figure 7, once again the crossover isn't working so well. The two other variations are working well.
In figure 8, we can observe that the threshold selection performs big steps, meaning that my T variable could have been set a little too big.
In figure 9, I reduced the T variable. We can observe smaller steps, thus converging faster to a better solution.

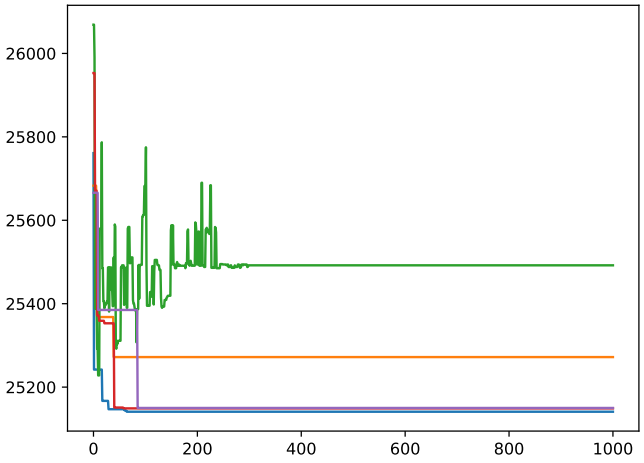

Figure 2: Comparaison for $\lambda = 5$, $\mu = 2$ and crossover variation of classic elitist selection (in blue), elitist selection on offsprings (in orange), fitness proportional (in green), Boltzmann with constant $T = 1000$ (in red) and Threshold selection with constant parameter $T = -150$ (in purple)

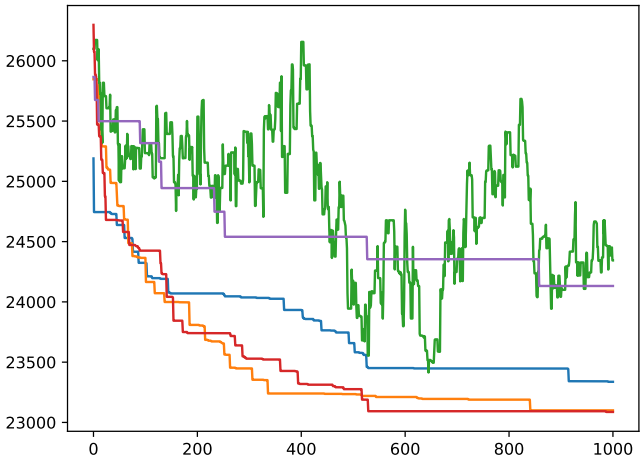

Figure 3: Comparaison for $\lambda = 5$, $\mu = 2$ and multiple variation of classic elitist selection (in blue), elitist selection on offsprings (in orange), fitness proportional (in green), Boltzmann with constant $T = 1000$ (in red) and Threshold selection with constant parameter $T = -150$ (in purple)

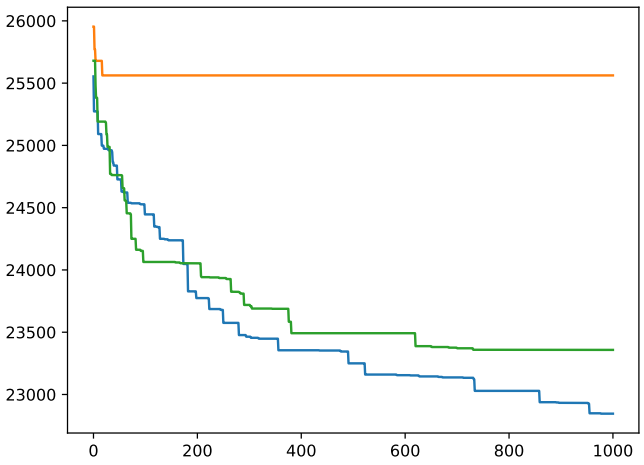

Figure 4: Comparaison for $\lambda = 5$, $\mu = 2$ and classic elitist selection of mutation variation (in blue), crossover variation (in orange) and another variation consisting of a mix of both (in green).

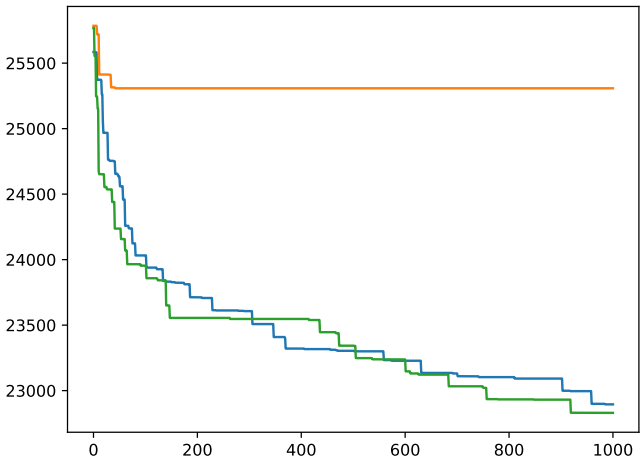

Figure 5: Comparaison for $\lambda = 5$, $\mu = 2$ and offsprings elitist selection of mutation variation (in blue), crossover variation (in orange) and another variation consisting of a mix of both (in green).

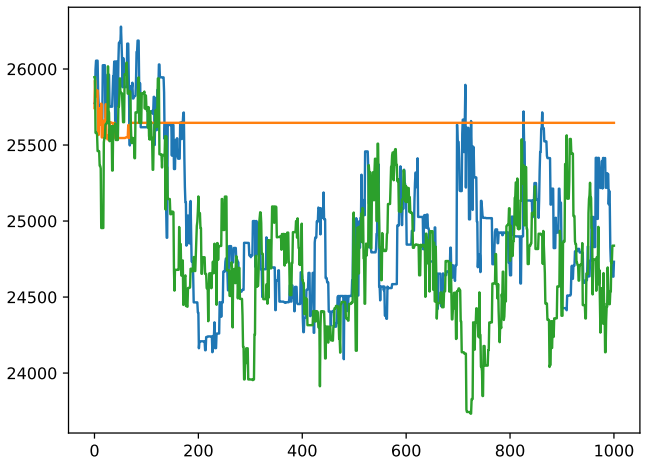

Figure 6: Comparaison for $\lambda = 5$, $\mu = 2$ and Fitness proportional selection of mutation variation (in blue), crossover variation (in orange) and another variation consisting of a mix of both (in green).

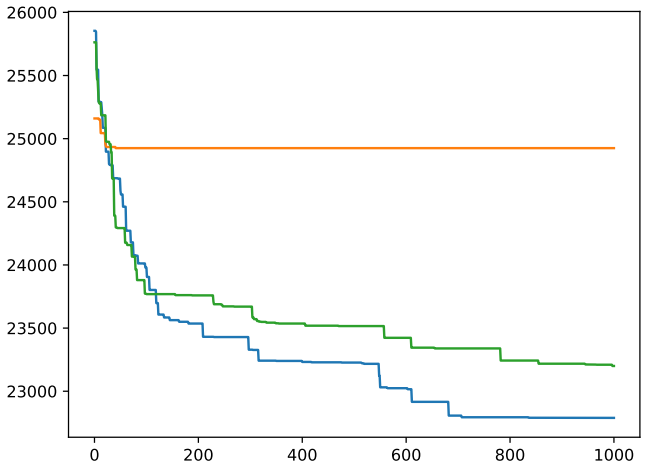

Figure 7: Comparaison for $\lambda = 5$, $\mu = 2$ and Boltzmann selection with constant parameter $T = 1000$ of mutation variation (in blue), crossover variation (in orange) and another variation consisting of a mix of both (in green).

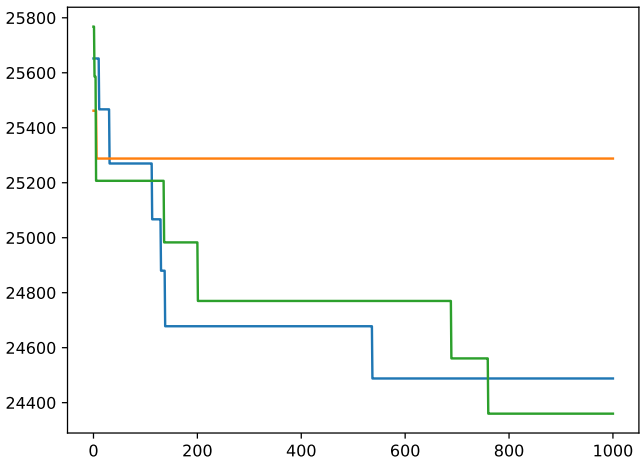

Figure 8: Comparaison for $\lambda = 5$, $\mu = 2$ and threshold selection with constant parameter $T = -150$ of mutation variation (in blue), crossover variation (in orange) and another variation consisting of a mix of both (in green).

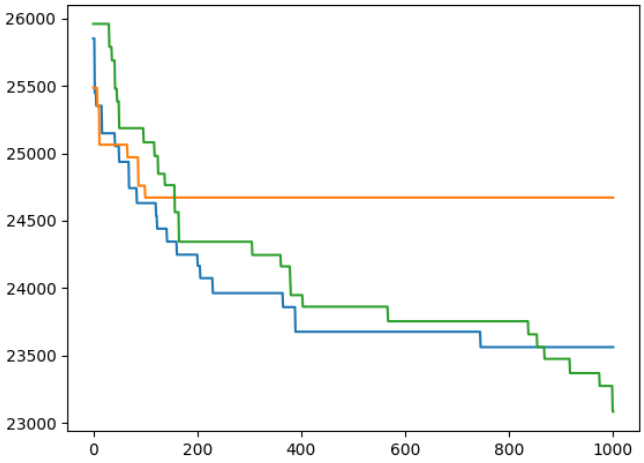

Figure 9: Comparaison for $\lambda = 5$, $\mu = 2$ and threshold selection with constant parameter $T = -80$ of mutation variation (in blue), crossover variation (in orange) and another variation consisting of a mix of both (in green).

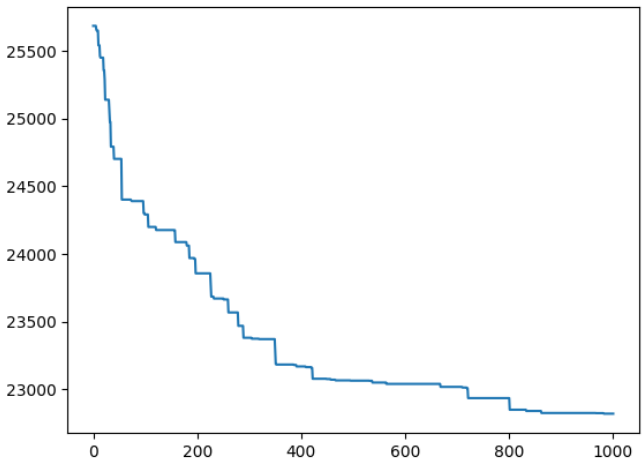

Figure 10: Comparaison for $\lambda = 11$, $\mu = 3$ and Boltzmann selection with constant parameter $T = 1000$ and mutation variation (in blue)

8 Conclusion

For this problem, it looks like my fitness proportional and crossover techniques aren't working very well. I am not sure why my crossover technique isn't showing good results.

Techniques that seem to be working well are mutation variation and Boltzmann selection.

Let's perform one last try with mutation variation and Boltzmann for $\lambda = 11$ and $\mu = 3$. See figure 10