Ai for TSP Competition

The Margaridinhas (@Federico, please put the real team picture hahaha!)

July 9, 2021

1 Introduction

Caroline

2 Surrogate-based tracker approach

Here is the complete description of the surrogate-based *tracker approach*.

2.1 Concept definitions

This section presents the technical terms used throughout the document:

- Simulation means one call to the black-box simulator with a route (namely, one call to the `check_solution` function of the `env.py` file);
- Solution means an output of the surrogate model (namely, a value for each decision variable after the optimization of the surrogate model);
- Route means a sequence of nodes starting at and going back to the depot;
- A solution represents a route, and a route may be represented as a solution. Thus, the terms route and solution may be used interchangeably.

2.2 Surrogate model

This section presents the surrogate model used in the tracker approach.

2.2.1 Data

The surrogate model takes into consideration the following deterministic data:

- $\mathcal{N}$ - set of nodes in the network;
- N - number of nodes in the network;
- T - maximum duration of the route;
- r_i - reward retrievable at node i;
- o_i - opening of time window at node i;
- c_i - closing of time window at node i;
- t_{ij} - worst travel time of arc (i, j).

The surrogate model takes into consideration the following stochastic data:

- p_{ij} - penalty associated with arc (i, j).

The stochastic data is estimated with increasing accuracy as previous simulations accumulate.

2.2.2 Decision

The surrogate model takes in consideration the following decision variables:

- x_{ij} - 1 if arc (i,j) is in the route, 0 otherwise;
- t_i - step of the route in which node i is visited.

We denote as $\mathcal{T}$ the space of the decision variables.
Let s denote the lenght of the route, the tracker approach solves the following surrogate model $\mathcal{M}(s)$:

$$\begin{aligned}
\mathcal{M}(s): \quad \max_{(x,t)\in\mathcal{T}} \quad & \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}}(r_j+p_{ij})\cdot x_{ij} && && \text{(1a)}\\
\text{subject to:} \quad & \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}} x_{ij}\le s && && \text{(1b)}\\
& \sum_{k\in\mathcal{N}} x_{1k}=1 && && \text{(1c)}\\
& \sum_{k\in\mathcal{N}} x_{k1}=1 && && \text{(1d)}\\
& \sum_{k\in\mathcal{N}} x_{ki}=\sum_{k\in\mathcal{N}} x_{ik} && \forall i\in\mathcal{N} && \text{(1e)}\\
& \sum_{k\in\mathcal{N}} x_{ik}\le 1 && \forall i\in\mathcal{N} && \text{(1f)}\\
& \sum_{k\in\mathcal{N}} x_{ki}\le 1 && \forall i\in\mathcal{N} && \text{(1g)}\\
& t_j-t_i\ge 1-N\cdot(1-x_{ij}) && \forall i\in\mathcal{N},\forall j\in\mathcal{N}/\{1\} && \text{(1h)}\\
& x_{ij}\in\{0,1\} && \forall i\in\mathcal{N},\forall j\in\mathcal{N} && \text{(1i)}\\
& t_i\ge 0 && \forall i\in\mathcal{N} && \text{(1j)}
\end{aligned}$$

The objective function (1a) maximizes the deterministic reward plus the estimated penalty of the route based on previous simulations. Constraint (1b) restricts the number of arcs a route can have to s, which is a parameter of the tracker approach. Constraints (1c)-(1h) define a valid route: Constraint (1c) guarantees that the route starts at the depot, whereas constraint (1d) guarantees that the route ends at the depot. Constraints (1e) preserve the flow at each node. Constraints (1g) require that the route can arrive at each node at most once, whereas constraints (1f) require that the route can depart from each node at most once. Constraints (1h) guarantee that the route does not contain cycles.

Note that the surrogate model $\mathcal{M}(s)$ does not explicitly handle time windows at nodes and the maximum duration of the route. The idea is to acquire this information

1. in the objective function by well estimating penalty p_{ij} (section 2.2.3) and
2. in the constraints by adding cuts (section 2.2.4).

2.2.3 Estimation

Let $\mathcal{H}$ be the set of pairs route-penalty (r, p) observed throughout the tracker approach, where r is a route and p is the penalty associated with route r in a particular implementations. In other words, set $\mathcal{H}$ stores part of the information from previous simulations. It is possible to estimate the value of the penalty p_{ij} per arc (i, j) by taking the average penalty equally distributed among arcs inside a route of routes with arc (i, j). More specifically, it is possible to calculate penalty p_{ij} using the following formula, where $occurrences((i, j), \mathcal{H})$ yields the number of routes in set $\mathcal{H}$ containing arc (i, j), $belongs((i, j), r)$ yields 1 if arc (i, j) is part of route r and 0 otherwise, and $size(r)$ yields the number of arcs in route r:

$$p_{ij} = \frac{1}{occurrences((i,j),\mathcal{H})} \cdot \sum_{(r,p)\in\mathcal{H}} \frac{belongs((i,j),r)\cdot p}{size(r)}$$

2.2.4 Cuts

In this section we expose the cuts that we employ in our tracker approach.

Cuts for infeasible solutions. Imagine that a solution x^k, found by the tracker approach, represents the route $[1, 2, 3, 1]$, and that this route is deemed infeasible. It follows by intuition that all routes starting with $[1, 2, 3]$ are likely to be infeasible as well. If infeasibility comes from violating a time window at a certain node, all routes starting with $[1, 2, 3]$ will still likely violate those time windows. If infeasibility comes from violating the maximum duration of the route, adding an extra stop to another node before returning to depot is also likely to violate the maximum duration of the route as the triangle inequality holds. Thus, constraint $x_{12} + x_{23} \leq 1$ could be added to the surrogate model $\mathcal{M}(s)$ to forbid the search for solutions that start with $[1, 2, 3]$.

More generally, let $\mathcal{A}(x^k)$ be the set of arcs in the route represented by solution x^k minus the return to the depot. In the example, $\mathcal{A}(x^k)$ would be $\{(1, 2), (2, 3)\}$. The following cut for an infeasible solution could be added to the surrogate model $\mathcal{M}(s)$ without harm for an infeasible solution x^k:

$$\sum_{(i,j)\in\mathcal{A}(x^k)} x_{ij} \leq |\mathcal{A}(x^k)| - 1$$

Cuts for feasible solutions. Imagine that a solution x^k, found by the tracker approach, represents the route $[1, 2, 3, 1]$, and that this route is deemed feasible. The tracker approach will run a standard number of simulations to identify what is the actual score (reward plus penalties) of the route and consider whether solution x^k should replace the best solution in the current iteration. However, nothing prevents solution x^k from appearing in the next iterations after the adjustment of the penalties in the objective function. In this sense, it is possible to add a cut for a feasible solution to the surrogate model $\mathcal{M}(s)$ to prevent it from considering solution x^k again in the next iterations.

More generally, let $\mathcal{B}(x^k)$ be the set of arcs in the route represented by solution x^k. Note that here the return to the depot is important. In the example, $\mathcal{B}(x^k)$ would be $\{(1, 2), (2, 3), (3, 1)\}$. The following cut for a feasible solution could be added to the surrogate model $\mathcal{M}(s)$ for a feasible solution x^k:

$$\sum_{(i,j)\in\mathcal{B}(x^k)} x_{ij} \leq |\mathcal{B}(x^k)| - 1$$

Impossible nodes. Apparently, some nodes cannot be part of attractive routes. An example would be a node with a time window that opens after the maximum duration of the route. These nodes form an impossible set $\mathcal{I}$. A node i is part of the impossible set $\mathcal{I}$ if a simple route that visits only node i has an average score (rewards plus penalties) smaller than zero. This set of impossibles nodes is generated at the beginning of the algorithm and these nodes are ignored in the surrogate model $\mathcal{M}(s)$ to decrease the size of the search space.

Risky arcs. There are some arcs that seem to be too risky to be part of attractive routes. More specifically, if it is not possible to reach node j before its closing time from node i while departing at the earliest time possible and considering the worst travel time (i.e., $o_i + t_{ij} > c_j$), then arc (i, j) is considered too risky to be part of a route. This set of risky arcs is generated at the beginning of the algorithm and these arcs are also ignored in the surrogate model $\mathcal{M}(s)$ to decrease the size of the search space.

2.3 Tracker approach

This section presents the components of the surrogate-based tracker approach.

The tracker approach takes into consideration the following parameters:

- K - maximum number of iterations;
- M - number of simulations per iteration;
- f - feasibility percentage threshold to consider a route as feasible or not (e.g., $f = 0.9$ means that from M simulations at least 90% of them must certify the route as feasible);
- g - gap threshold when considering the upper bound for a route size s.

The tracker approach has the following steps:

1. Eliminate impossible nodes and risky arcs;
2. Set the initial value for $s \leftarrow 2$;
3. Initiate the iteration counter $k \leftarrow 0$;
4. Run the surrogate model $\mathcal{M}(s)$ and obtain a solution x^k;
5. Identify the route r^k represented by solution x^k;
6. Simulate route r^k M times and assess the output;
7. If route r^k is feasible at least f percent of the time, store solution x^k and add a cut for feasible solutions. If not, add a cut for infeasible solutions;
8. Calculate the upper bound u^k for size s (namely, solve the surrogate model $\mathcal{M}(s)$ without considering estimated penalties in the objective function);

9. If the gap between the best objective and the upper bound u^k at iteration k is less than g, allow larger routes by setting $s \leftarrow s + 1$;
10. Update the historical observation set $\mathcal{H}$ with route r^k represented by solution x^k and the average penalty p^k over M simulations;
11. Update the penalties p_{ij} according to the updated set $\mathcal{H}$;
12. Return to step 4 until $k > K$ or the best objective meets the upper bound;
13. Output the feasible route r^k with the best objective.

2.4 Improvement step

The tracker approach generates different solutions according to the parameters. In this sense, a genetic algorithm has been written to work with the different solutions outputted by the tracker approach and come up with nicer solutions. The genetic algorithm is used as a random local search policy, and would be very difficult to exploit in stand-alone, as we run it for only a few generations on the warmed-up solutions of the surrogate steps to see if there exists a better solution in the neighborhood. More about it in Section 3.

2.5 Track qualification

The tracker approach qualifies for the surrogate-based track because it obeys the definition in the problem statement: *Given one instance, previously tried routes, and the reward for those routes, the goal is to learn a model that can predict the reward for a new route.* Given the historical observation set $\mathcal{H}$, the surrogate model $\mathcal{M}(s)$ is progressively updated to predict the reward for a new route of length s. *Then an optimizer finds the route that gives the best reward according to that model, and that route is evaluated, giving a new data point.* The mixed-integer programming solver, CPLEX, solves the current version of the surrogate model $\mathcal{M}(s)$ and outputs a solution that represents a route. The related route is simulated by the tracker approach M times to determine what is the actual score (reward plus penalties) of the route. *Then the model is updated, and this iterative procedure continues for a fixed number of steps. Over time, the model becomes more accurate, giving better and better routes.* The historical observation set $\mathcal{H}$ grows over time and the surrogate model $\mathcal{M}(s)$ yields progressively better solutions due to a better estimation of the penalties and the addition of constraints for visited feasible and infeasible solutions.

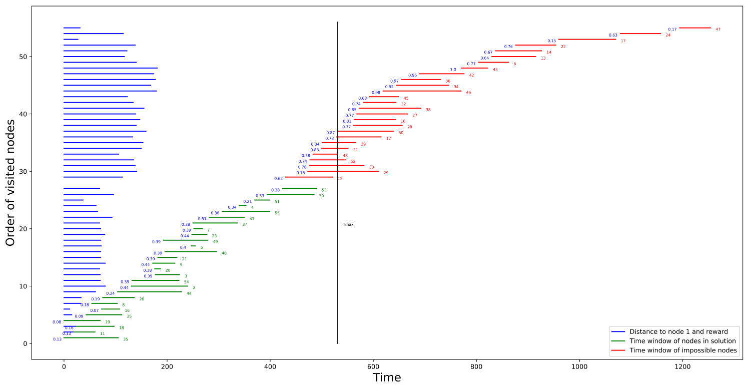
3 Genetic algorithm

Justine

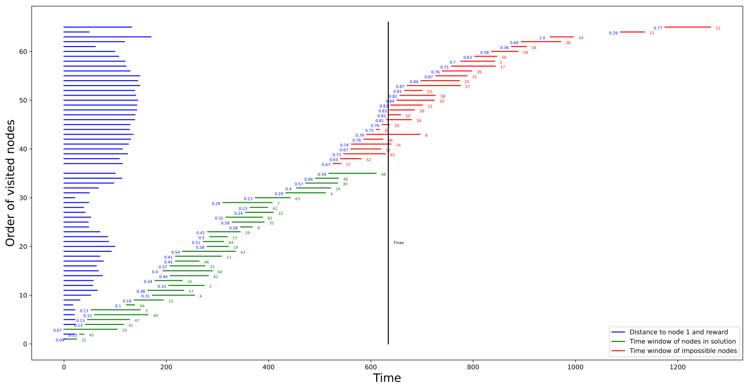
4 Why we can take the last weekend off

As described in Section 2.2.4, we were able to ignore many nodes that we marked as impossible nodes. Every impossible node n_i is defined as having a negative objective for the tour $[1, n_i, 1]$, mostly because of a late opening time o_i, forcing the tour to arrive later than time T. Those nodes were marked in red in Figures

1a and 1b which represent the best solutions we found on the validation and test instances. By summing the rewards of the remaining nodes (in green), we were able to obtain a lower bound on the value of the optimal tour. Those lower bounds were of value 8.52 and 11.32 for the validation and test instance respectively. Using the mathematical programming and genetic algorithm approaches, we were able to find solutions that achieve those values. From that point, the only thing that could improve our solution was to try to incorporate an impossible node to our solution. But this sounded difficult from the definition of the impossible nodes. We managed to find some nodes n_j such that the tour $[1, n_i, n_j, 1]$ was faster than $[1, n_i, 1]$ for some impossible nodes n_i, but due to the stochasticity of the problem, we always obtained a small degree of infeasibility, which made that we did not add any impossible node in the final solutions. We concluded that the lower bound was also the upper bound of the maximization problem, and that we had a solution reaching that bound. We could thus take the last weekend off.



(a)



(b)

Fig. 1: (a) Best solution found on validation instance. (b) Best solution found on test instance.