# Improvements for mlrose applied to the Traveling Salesperson Problem

Michael Lehenauer Stefan Wintersteller Martin Uray Stefan Huber

Salzburg University of Applied Sciences
Salzburg, Austria

EUROCAST 2022 — Gran Canaria, Spain
Feb 20–25, 2022

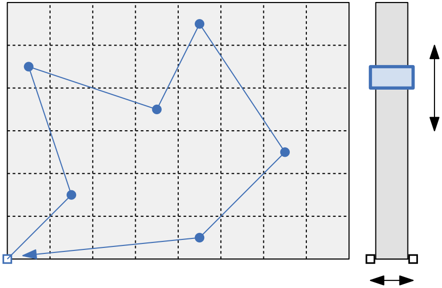
FH Salzburg

Interreg Österreich–Bayern 2014–2020
Europäische Union – Europäischer Fonds für Regionale Entwicklung

# Motivation: Material flow in high-bay storage

## Commissioning problem

Given a finite set of object locations, what is the most efficient path to visit them all?



Mathematical model:

- Efficiency of a path is given by its length $\ell$.
- In a simple setting we consider Euclidean metric $d$ on $\mathbb{R}^2$.
- Disparity in motion direction modeled through anisotropy in the metric $d$ of a metric space $(X, d)$.

# Notation

Given a finite point set $p_0, \ldots, p_{n-1}$ in the metric space $(X, d)$. A tour $\pi$ corresponds to a permutation

$$\pi\colon \{0, \ldots, n-1\} \to \{0, \ldots, n-1\}$$

over the index set $\{0, \ldots, n-1\}$ and its length $\ell$ is defined as

$$\ell(\pi) = \sum_{i=0}^{n-1} d(p_{\pi(i)}, p_{\pi(i+1 \bmod n)}).$$

## Commissioning problem

What is the optimal tour, i.e., what is $\arg\min_\pi \ell(\pi)$?

This is the Traveling Salesperson Problem, which asks for the minimum-weight Hamiltonian cycle in an edge-weighted graph in the more general setting of graphs.

# Background

- Classical optimization problem in operations research and algorithm theory.
- Given some $L > 0$, deciding whether a TSP tour $\pi$ with $\ell(\pi) \leq L$ exists, is NP-complete. (Also in case of Euclidean metric.)

Approximation algorithms:

- Euclidean plane provides additional structure that can be exploited.
- Christofides algorithm is a 1.5-approximation that runs in $O(n^3)$ time.
- Based on the Euclidean minimum spanning tree, which is a subgraph of the Delaunay triangulation.
- Mitchell and Arora independently found a polynomial-time approximation scheme.

TSP attracted AI research, as many NP-hard problems:

- Logic-based methods, e.g., through Singular Modulo Theories (phrased as a ILP problem).
- Machine learning and heuristic search methods, e.g., ant colony, genetic algorithms, particle swarm, simulated annealing, hill climbing, reinforcement learning, et cetera.

# TSP in mlrose

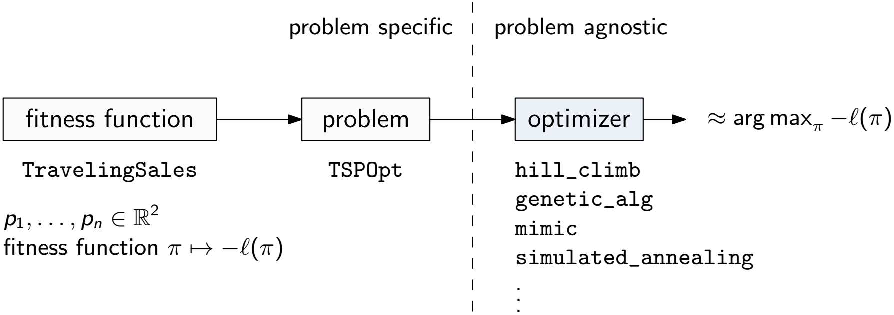
## Question

How can we apply AI methods form an engineering perspective?

The library mlrose stands for *machine learning, random optimization and search*:

- Implemented in Python, mainly by Genevieve Hayes
- Moderately active: 174 forks on github, 10 developers, but last commit from end 2019
- Provides hill climbing, simulated annealing, genetic algorithm, and MIMIC
- Various optimization problems already integrated, including TSP

# mlrose architecture




In the following, we restrict investigations to two optimization methods:

- Genetic algorithm
- Hill climbing

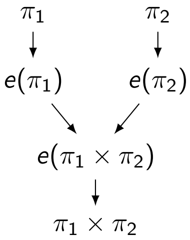
# Genetic Algorithm

## Basic idea of GA is “survival of the fittest”

- Consider a population of individuals (TSP candidate solutions)
- Apply two genetic operators: (i) mutation (random alteration) and (ii) crossover (recombination of two parents) to form offsprings
- Apply a selection to find the fittest individuals to form a new generation of the population.

A suitable genetic encoding $e(\pi)$ of $\pi$ is paramount:

- The encoding $e(\pi)$ is a string over some alphabet.
- Crossover: Form $e(\pi_1 \times \pi_2)$ from $e(\pi_1)$ and $e(\pi_2)$.
- Single-point crossover $\pi_1 \times \pi_2$:
  Take a random prefix of $e(\pi_1)$ and complete it with $e(\pi_2)$.
- Care needs to be taken:
  $e(\pi_1 \times \pi_2)$ needs to be a valid encoding of an individual $\pi_1 \times \pi_2$.



# GA in mlrose

In mlrose, a tour $\pi$ is encoded as the sequence of visited locations, i.e., $e(\pi) = (\pi(0), \ldots, \pi(n-1))$.

- Crossover: random prefix of $e(\pi_1)$ concatenated with the missing locations as they occur in $e(\pi_2)$.
- Hence, $e(\pi_1 \times \pi_2)$ is guaranteed to encode a permutation $\pi_1 \times \pi_2$.

In mlrose, a tour $\pi$ is encoded as the sequence of visited locations, i.e., $e(\pi) = (\pi(0), \ldots, \pi(n-1))$.

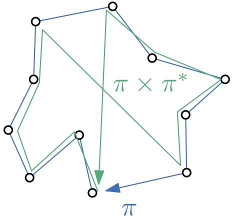
- Crossover: random prefix of $e(\pi_1)$ concatenated with the missing locations as they occur in $e(\pi_2)$.
- Hence, $e(\pi_1 \times \pi_2)$ is guaranteed to encode a permutation $\pi_1 \times \pi_2$.

Symmetry of TSP:

- Take a tour $\pi$ and consider its reversed tour $\pi^*$.
- Since $\ell(\pi) = \ell(\pi^*)$ they have equal fitness.

The crossover operator does not respect this:

- If $\pi$ has good fitness, or is even optimal, so is $\pi^*$.
- But $\pi \times \pi^*$ has most likely bad fitness.
- The offspring of fit parents has bad fitness.



# An improved crossover operator

## Essence of the problem

- We seek for a direction-conforming recombination, which respects "traversal direction".
- However, there is no natural mathematical notation of such a traversal direction of a tour $\pi$ suitable for our problem. Hence, we mathematically "factor out" the two possible traversal directions.

# An improved crossover operator

## Essence of the problem

- We seek for a direction-conforming recombination, which respects "traversal direction".
- However, there is no natural mathematical notation of such a traversal direction of a tour $\pi$ suitable for our problem. Hence, we mathematically "factor out" the two possible traversal directions.

We propose this crossover operator $\otimes$:

$$\pi_1 \otimes \pi_2 = \begin{cases} \pi_1 \times \pi_2 & \text{if } \ell(\pi_1 \times \pi_2) \leq \ell(\pi_1 \times \pi_2^*) \\ \pi_1 \times \pi_2^* & \text{otherwise} \end{cases}$$

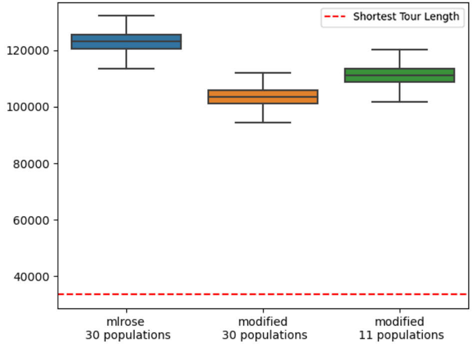
Note that $\otimes$ is reversal-invariant, i.e.,

$$\pi_1 \otimes \pi_2 = \pi_1 \otimes \pi_2^*$$

# Experimental results

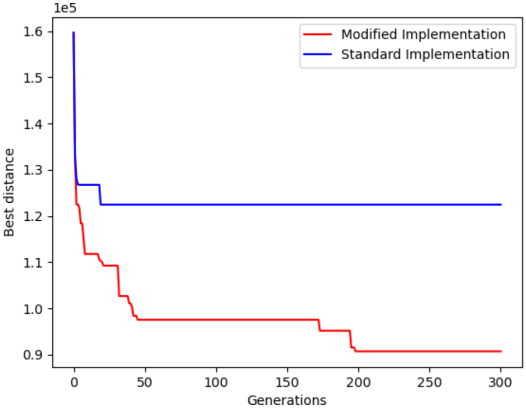
## Experimental setup

- 1000 runs to obtain TSP on the att48 dataset of TSPLIB (48 cities)
- 30 generations, population size of 200, zero mutation rate (for the sake of comparison)

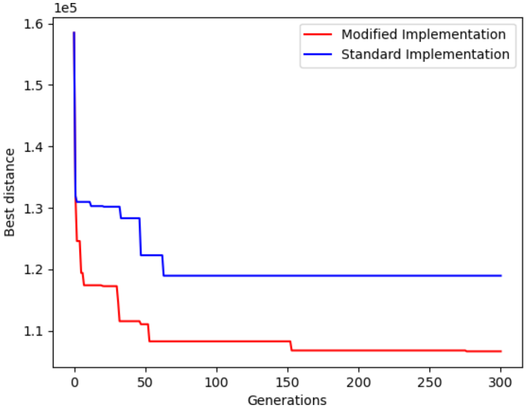



- New crossover operator leads to 16 % shorter tour lengths: $103278 \pm 3370$ versus $122714 \pm 4003$.
- New crossover operator comes at higher runtime costs: $14505 \pm 666$ ms versus $5281 \pm 350$ ms. (Python implementation!)
- We rerun with a reduced number of generations (11) to obtain similar runtime, and still get lower tour lenghts (110888 ±3689).

# Experimental results




(a) Mutation rate 0.0




(b) Mutation rate 0.1

Figure: Decline of best $\ell(\pi)$ over generations. Population size 100.

# Next steps for Genetic Algorithms

## Hypothesis

In early generations, the individuals are close to random.

- We therefore expect that $\times$ and $\otimes$ give similarly fit offsprings.
- The advantage of $\otimes$ kicks in at later generations.

# Next steps for Genetic Algorithms

## Hypothesis

In early generations, the individuals are close to random.

- We therefore expect that $\times$ and $\otimes$ give similarly fit offsprings.
- The advantage of $\otimes$ kicks in at later generations.

Let us define the reversal discrepancy $\Delta$ as

$$\Delta(\pi_1, \pi_2) = |\ell(\pi_1 \times \pi_2) - \ell(\pi_1 \times \pi_2^*)|$$

## Motivation

Large $\Delta$ means $\otimes$ is better than $\times$. Investigating the evolution of $\Delta$ gives an understanding on how $\otimes$ unfolds its advantage.

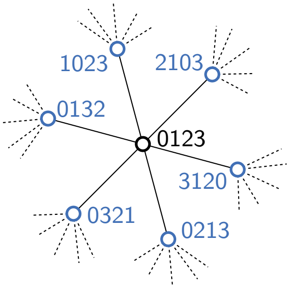
- At the first generation, $\Delta$ it is the (absolute) difference of the random variable $\ell(\pi)$ for random $\pi$.
- Hypothesis: At later generations the expectation of $\Delta$ grows, and variance declines

# Hill Climbing

## Basic idea of HC

- Consider a fitness function $f : D \to \mathbb{R}$. Starting at a random position, follow steepest ascent until a local maximum has been reached.
- Then possibly restart to find a better ascending path.

- We have fitness $f = -\ell$.
- The domain $D$ is the transposition graph $G = (V, E)$ with $V$ being the set of permutations $\pi$ and $\{\pi, \pi'\} \in E$ if $\pi$ and $\pi'$ differ by a transposition.
- Each $\pi$ has a neighborhood $N(\pi)$ of $\binom{n}{2}$ permutations.
- In each step, HC proceeds from $\pi$ to the $\ell$-minimizing $\pi' \in N(\pi)$.



# Shortcomings of vanilla HC

Shortcomings have been extensively studied:

- Plateaus, regions where fitness is constant, are an issue of HC.
  - Mitigation exists, but not in mlrose.
  - However, unlikely that $\ell$ would be be constant in $N(\pi)$ for a $\pi$.
- HC easily gets stuck in local optimum.
  - Restarts shall mitigate this.

## Our proposal

- We aim to prolong descending paths on the $\ell$-landscape over $G$.
- Idea: Escape “insignificant” local minima via “easy to pass” shoulders.
- Idea similar to momentum-based gradient descent optimizers.

# Escaping insignificant local optima

## Basic idea

- Allow for making one upward step when stuck in a local minimum.
- If the following step would lead us back, we terminate. Otherwise we were able to prolong the descending path.

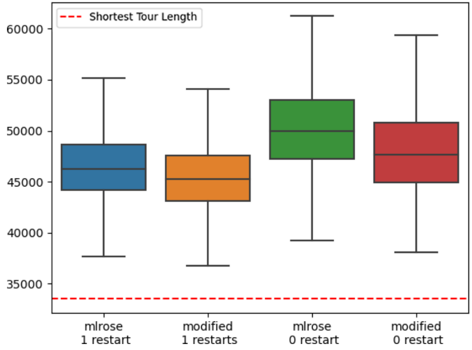
Adds another improvement en passant:

- We now need to check whether we already visited vertices of $G$.
- Keep this information over restarts to allow for early outs.

# Experimental results

## Experimental setup

- 1000 runs to obtain TSP on the att48 dataset of TSPLIB (48 cities)




- With 0 restarts (default), the tour length was reduced by 4.6 % to $47944 \pm 4348$ from $50263 \pm 4498$.
- With 1 restart, the tour length was reduced by 2.1 % to $45420 \pm 3340$ from $46438 \pm 3427$.
- Increasing restart to 1 slowed down runtime by a factor of 1.9. For 0 restarts the modified version an in $18273 \pm 2980$ ms, the original version in $16701 \pm 2551$ ms.
- Summary: Modification with 0 restarts leads to similar results than original version with 1 restart, but is factor of 1.9 faster.

# Experimental results

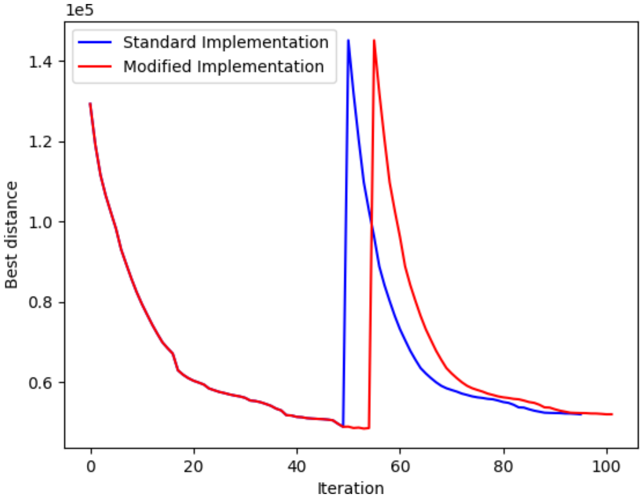



Figure: Decline of tour length with one restart.

# Next steps for Hill Climbing

## Topological interpretation

Consider 0-th persistent homology on the $\ell$-sub-levelset filtration over $G$.

- We aim to escape local optima of low persistence, e.g., $k \geq 1$.

- Consider $V_L = \{\pi \in V \colon \ell(\pi) \leq L\}$ with $L$ growing from 0 to $\infty$.
- The topology $V_L$ changes over time:
  - At local minima connected components are created (birth).
  - At shoulders connected components merge. Persistent homology: The "younger" component joins (death) the "older" one.
  - Peristence is the difference between time of death and time of birth.

## Open questions

- What is the distribution of the persistence of local minima of the $\ell$-landscape?
- How large are their basins in the sense of [HML18]?

# Conclusion

## KI-Net

The KI-Net research project is about

- AI methods for manufacturing and production
- with a particular focus on the small/medium sized economy.

Central theme: Reduce access barrier to AI for SMEs.

# Conclusion

## KI-Net

The KI-Net research project is about

- AI methods for manufacturing and production
- with a particular focus on the small/medium sized economy.

Central theme: Reduce access barrier to AI for SMEs.

Reflection based on this talk:

- Methods of certain domains can be used like fire & forget, whereas others don't.
- For instance, software library for numerical mathematics are fire & forget solutions.
- For instance, control theory does not.

Are there inherent reasons why AI and machine learning cannot provide "fire & forget"?

# Thank you

Questions?

# Bibliography I

[HML18] Leticia Hernando, Alexander Mendiburu, and Jose A. Lozano. "Hill-Climbing Algorithm: Let's Go for a Walk Before Finding the Optimum." In: *2018 IEEE Congress on Evolutionary Computation (CEC)*. Rio de Janeiro: IEEE, July 2018, pp. 1–7. ISBN: 978-1-5090-6017-7. DOI: 10.1109/CEC.2018.8477836.