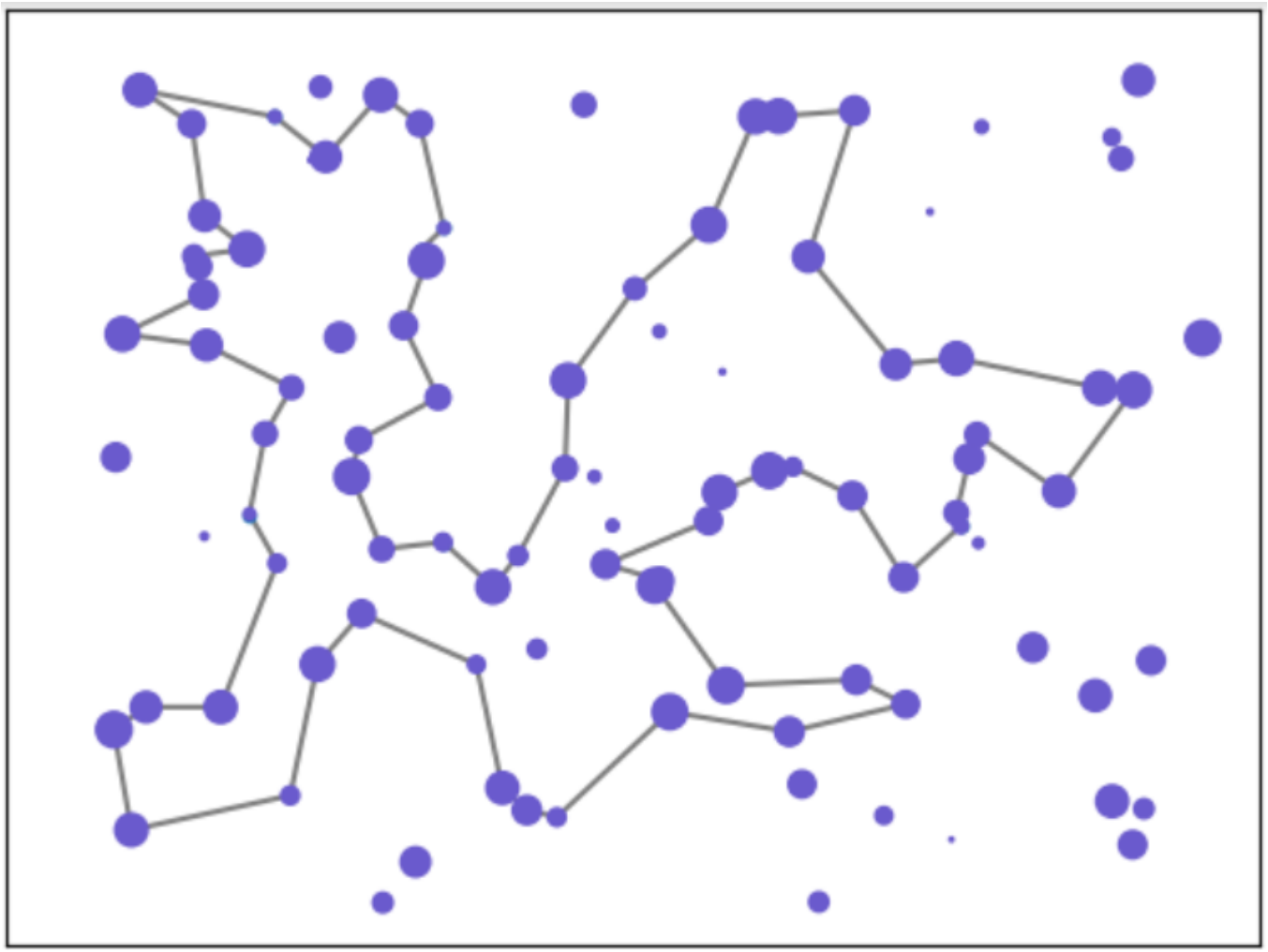
The Price-Collecting Traveling Salesman and Related Problems

Czirják Lilla
Supervisor: Király Tamás

09 January, 2025

Price-Collecting Traveling Salesman



Definition

Given a complete graph $G = (V,\ E)$, a root $r \in V$,
$c_e \geq 0\ \forall e \in E$ metric lengths of the edges and
$\pi_v \geq 0\ \forall v \in V \backslash \{r\}$ vertex weights,
the **price-collecting traveling salesman problem** is to find a cycle $C = (V_C,\ E_C)$ in G, so that $r \in V_C$, and
$\sum_{e \in E_C} + \sum_{v \in V \backslash V_C}$ is minimal.

LP Relaxation

$$min \sum_{e \in E} C_e x_e + \sum_{v \in V} \pi_v (1 - y_v)$$

$$\begin{aligned}
x(\delta(v)) &= 2y_v && \forall v \in V \backslash \{r\} \\
x(\delta(r)) &\leq 2 && \\
x(\delta(S)) &\geq 2y_v && \forall S \subseteq V \backslash \{r\},\ v \in S \\
y_r &= 1 && \\
x_e &\geq 0 && \forall e \in E \\
y_v &\geq 0 && \forall v \in V
\end{aligned}$$

Where

$\delta(S) := \{e \in E : \ |e \cap S| = 1\}$

$x(\delta(S)) \ := \ \sum_{e \in \delta(S)} x_e$

Implementation of the LP

- Language: Python
- Library: PuLP
- Solver: PULP-CBC-CMD

Issue: number of constraints is exponential

$$\hookrightarrow x(\delta(S)) \geq 2y_v \quad \forall S \subseteq V \backslash \{r\},\ v \in S$$

Code Optimization Strategy

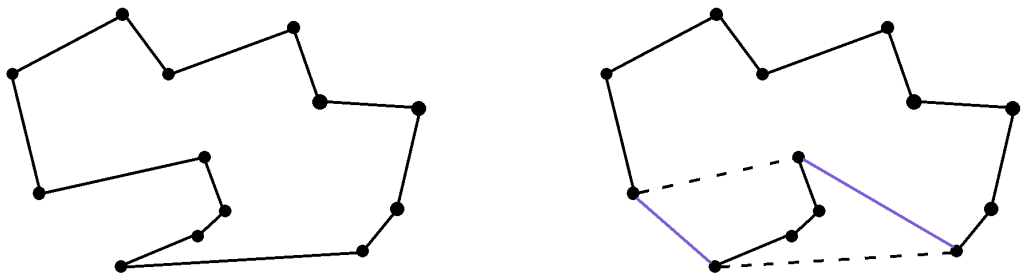
1. **Initial Model:** Exclude all set constraints of the form $x(\delta(S)) \geq 2y_v$.
2. **Iterative Refinement:**
   - 2.1 Solve initial model.
   - 2.2 Identify violated constraints:
     - For disjoint graphs: Add constraints for connected components.
     - For connected graphs: Check minimal cuts and add constraints for violated sets.
   - 2.3 Re-solve the updated model.
3. **Repeat** until no more constraints are added or time becomes impractical.

Heuristic Algorithm I

1. **Setup:**
   - Be given an n-vertex complete graph (metric edge lengths, vertex weights).
   - Begin with root node as a one-point tour.
2. **Vertex Insertion:** Add one vertex to the tour which leads to the biggest improvement.

Heuristic Algorithm II

3. **Improvement Steps:**
   - **Edge-Swapping:**

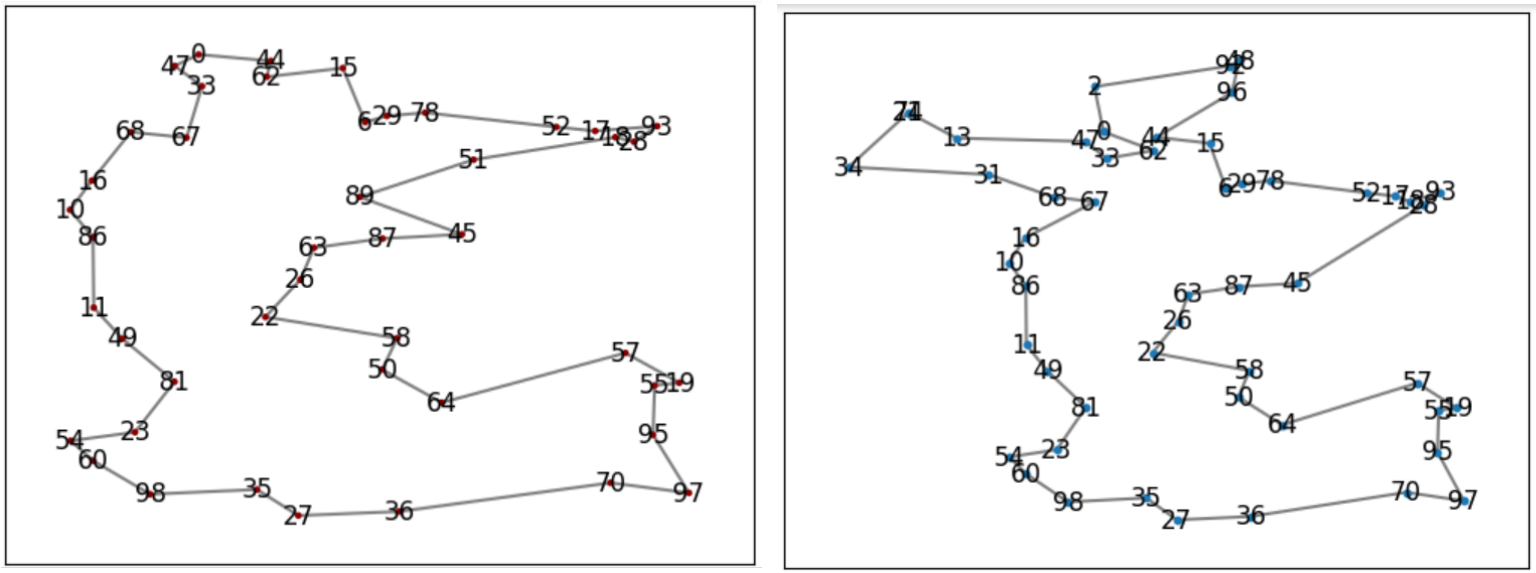


   - **Vertex-Dropping:** Check if removing a vertex reduces the objective value.
4. **Algorithm Termination:**
   - Iterate until all $n - 1$ vertices are added.
   - Track the best tour throughout and return as the final solution.

Setting Parameters

- Generate pseudorandom graphs from seeds using NumPy.
- **Number of vertices:** fixed $n = 100$
- **Vertex weights:** pseudorandom integers between 0 and 100
- **Edge lengths:** pseudorandom coordinates between 0 and 700, euclidean distances, rounding

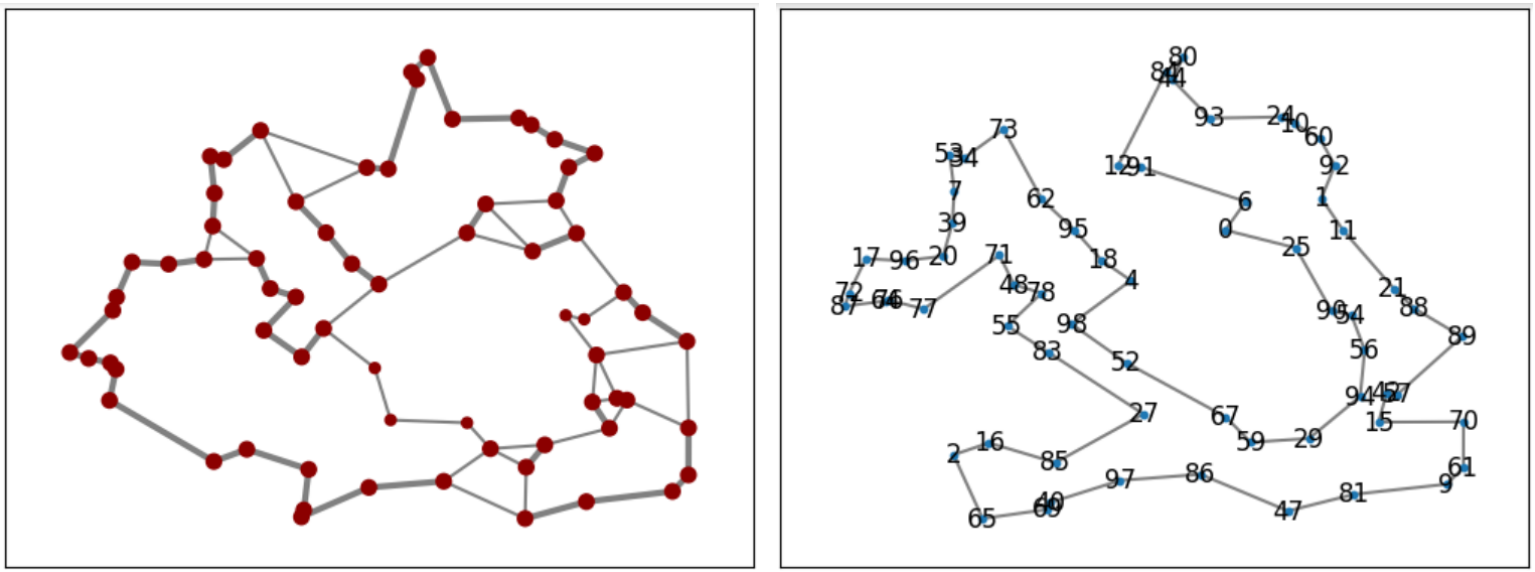
Some Tours Found by the Algorithms



Left: LP, Right: heuristic

Objective value by the heuristic algorithm: 4658
Lower bound found by the LP solution: 4494

Some Tours Found by the Algorithms



Left: LP, Right: heuristic

Objective value by the heuristic algorithm: 4879
Lower bound found by the LP solution: 4611.5

Some experimental results

seed	value of LP	value of heuristic	difference in values	solution size of LP	tour size of heuristic	difference in tour sizes
0	4564	4683	119	46	26	-20
1	4336	4664	328	52	63	11
2	4696	4925	229	42	1	-41
3	4295	4595	300	50	40	-10
4	4299,5	4580	280,5	64	70	6
5	4142	4660	518	50	51	1
6	4368	4711	343	61	3	-58
7	4568	4929	361	58	1	-57
8	4494	4658	164	45	52	7
9	4319	4493	174	64	43	-21
10	4198,75	4361	162,25	74	21	-53
11	4332,986	4578	245,014	70	29	-41
12	4553,348	5014	460,6518	94	55	-39
13	4267	4613	346	41	47	6
14	4740,5	5016	275,5	59	45	-14
15	4611,5	4879	267,5	69	66	-3
16	4091	4264	173	30	15	-15
17	4222	4380	158	40	21	-19
18	4277,875	4581	303,125	85	62	-23
19	4455	4760	305	59	29	-30
20	4334	4651	317	59	56	-3
21	4315	4451	136	60	16	-44
22	3957,652	4154	196,3485	77	29	-48
23	4482	4602	120	51	19	-32
24	4332	4887	555	59	1	-58
25	4765	5159	394	62	70	8
26	4386	4992	606	54	73	19
27	4541,5	4899	357,5	68	28	-40
28	4662	5089	427	57	77	20
29	4095	4397	302	71	37	-34

Thank you for your attention!