

Multi-Goal Planning and Data Collection Planning

Jan Faigl

Department of Computer Science
Faculty of Electrical Engineering
Czech Technical University in Prague

Lecture 06

B4M36UIR – Artificial Intelligence in Robotics

Overview of the Lecture

- Part 1 – Multi-Goal Planning and Data Collection Planning
 - Multi-Goal (Data Collection) Planning
 - Traveling Salesman Problem (TSP)
 - Traveling Salesman Problem with Neighborhoods (TSPN)
 - Generalized Traveling Salesman Problem (GTSP)
 - Orienteering Problem (OP)
 - Orienteering Problem with Neighborhoods (OPN)
 - Combined Profit with Shortest Path

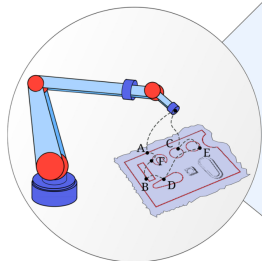
Part I

Part 1 – Multi-Goal Planning and Data Collection Planning

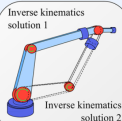
Multi-Goal Planning

- Having a set of locations to be visited, determine the cost-efficient path to visit them and return to a starting location.
 - Locations where a robotic arm or mobile robot performs some task
- The problem is called **robotic task sequencing problem** within the context of robotic manipulators

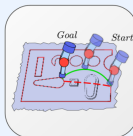
Robotic Task Sequencing Problem



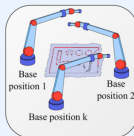
Multiple IK



Obstacle avoidance



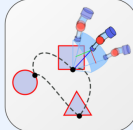
Robot base layout



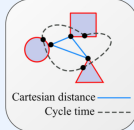
Partial order



Task specification



Objective

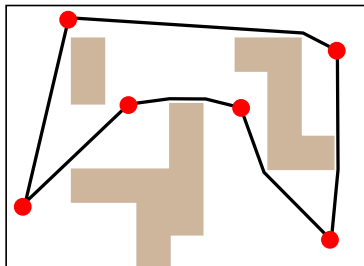
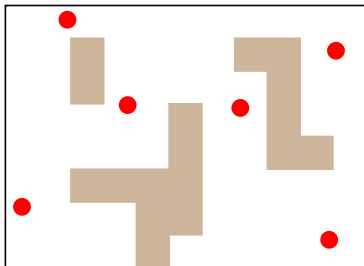


Alatartsev, S., Stellmacher, S., Ortmeier, F. (2015): **Robotic Task Sequencing Problem: A Survey**. Journal of Intelligent & Robotic Systems.

- It is also called **Multi-Goal Path Planning (MTP)** problem

Multi-Goal Path Planning (MTP)

- Multi-goal path planning problem is a problem to determine how to visit the given set of locations
- It consists of point-to-point planning problems how to reach one location from another
- The main “added” challenge to the path planning is a determination of the optimal sequence of the visits to the locations (with respect to the cost-efficient solution to visit all the given locations)



- Determining the sequence of visits is a **combinatorial optimization problem** that can be formulated as the **Traveling Salesman Problem**

Traveling Salesman Problem (TSP)

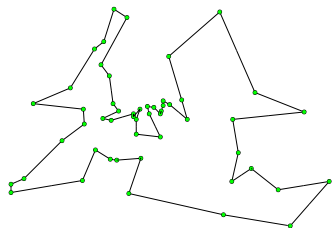
Given a set of cities and the distances between each pair of cities, what is the shortest possible route that visits each city exactly once and returns to the origin city.

- The TSP can be formulated for a graph $\mathbf{G}(V, E)$, where V denotes a set of locations (cities) and E represents edges connecting two cities with the associated travel cost c (distance), i.e., for each $v_i, v_j \in V$ there is an edge $e_{ij} \in E$, $e_{ij} = (v_i, v_j)$ with the cost c_{ij} .
- If the associated cost of the edge (v_i, v_j) is the Euclidean distance $c_{ij} = |(v_i, v_j)|$, the problem is called the **Euclidean TSP** (ETSP).
In our case, $v \in V$ represents a point in \mathbb{R}^2 and solution of the ETSP is a path in the plane.
- It is known, the TSP is NP-hard (its decision variant) and several algorithms can be found in literature.

William J. Cook (2012) – In Pursuit of the Traveling Salesman: Mathematics at the Limits of Computation

Existing Approaches to the TSP

- Efficient heuristics from the Operational Research have been proposed
- LKH – K. Helsgaun efficient implementation of the Lin-Kernighan heuristic (1998)
<http://www.akira.ruc.dk/~keld/research/LKH/>
- Concorde – Solver with several heuristics and also optimal solver
<http://www.math.uwaterloo.ca/tsp/concorde.html>

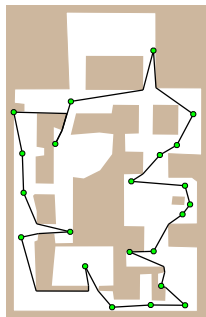


Problem Berlin52 from the TSPLIB

Beside the heuristic and approximations algorithms (such as Christofides 3/2-approximation algorithm), other („soft-computing”) approaches have been proposed, e.g., based on genetic algorithms, and memetic approaches, ant colony optimization (ACO), and [neural networks](#).

Multi-Goal Path Planning (MTP) Problem

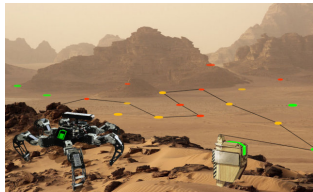
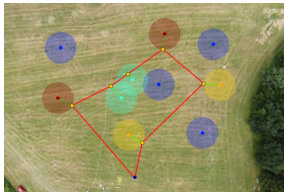
- **MTP problem** is a **robotic variant of the TSP** with the edge costs as the length of the *shortest* path connecting the locations
- In comparison to the traditional TSP, variants of **the robotic TSP respect selected constraints arising from limitations of real robotic systems**
E.g., obstacles, curvature-constraints, sensing range, location precision, etc.
- For n locations, we need to compute up to n^2 shortest paths
- The paths can be found as the shortest path in a graph (roadmap), from which the $G(V, E)$ for the TSP can be constructed



Visibility graph as the roadmap for a point robot provides a straight forward solution, but such a shortest path may not be necessarily feasible for more complex robots

Data Collection Planning

- Provide cost-efficient path to collect **all** (TSP) or the **most valuable data** (measurements) with **shortest possible path/time** or under **limited travel budget**



- Traveling Salesman Problem (TSP)** – visit all locations with minimal cost
 - TSP with Neighborhoods (TSPN)** – exploit non-zero sensing range
 - Close Enough TSP (CETSP)** – disk shaped neighborhoods
 - Generalized TSP (GTSP)** – “a discrete variant of the TSPN”
 - TSP and GTSP are pure combinatorial optimizations, while TSPN also includes continuous optimization
- Orienteering Problem (OP)** – maximize reward under limited travel budget
 - OP with Neighborhoods (OPN)** – exploit non-zero sensing range
 - Close Enough (CEOP)** – disk shaped neighborhoods
 - Set OP (SOP)** – “a discrete variant of the OPN”
 - OP and SOP are pure combinatorial optimization problems to select a subset of locations to visit and sequence to visit the locations

Traveling Salesman Problem (TSP)

- Let S be a set of n sensor locations $S = \{\mathbf{s}_1, \dots, \mathbf{s}_n\}$, $\mathbf{s}_i \in \mathbb{R}^2$ and $c(\mathbf{s}_i, \mathbf{s}_j)$ is a cost of travel from \mathbf{s}_i to \mathbf{s}_j
- Traveling Salesman Problem (TSP)** is a problem to determine a closed tour visiting each $\mathbf{s} \in S$ such that the total tour length is minimal, i.e.,
 - determine a **sequence of visits** $\Sigma = (\sigma_1, \dots, \sigma_n)$ such that

$$\text{minimize } \Sigma \quad L = \left(\sum_{i=1}^{n-1} c(\mathbf{s}_{\sigma_i}, \mathbf{s}_{\sigma_{i+1}}) \right) + c(\mathbf{s}_{\sigma_n}, \mathbf{s}_{\sigma_1}) \quad (1)$$

$$\text{subject to} \quad \Sigma = (\sigma_1, \dots, \sigma_n), 1 \leq \sigma_i \leq n, \sigma_i \neq \sigma_j \text{ for } i \neq j$$

- The TSP can be considered on a graph $G(V, E)$ where the set of vertices V represents sensor locations S and E are edges connecting the nodes with the cost $c(\mathbf{s}_i, \mathbf{s}_j)$
 - For simplicity we can consider $c(\mathbf{s}_i, \mathbf{s}_j)$ to be Euclidean distance; otherwise, we also need to address the path/motion planning problem
- Euclidean TSP**
- If $c(\mathbf{s}_i, \mathbf{s}_j) \neq c(\mathbf{s}_j, \mathbf{s}_i)$ it is the **Asymmetric TSP**
 - The TSP is known to be NP-hard unless $P=NP$

Traveling vs Travelling – <http://www.math.uwaterloo.ca/tsp/history/travelling.html>

Existing solvers to the TSP

■ Exact solutions

- Branch and Bound, Integer Linear Programming (ILP)

E.g., Concorde solver – <http://www.tsp.gatech.edu/concorde.html>

■ Approximation algorithms

- Minimum Spanning Tree (MST) heuristic with $L \leq 2L_{opt}$
- Christofides's algorithm with $L \leq \frac{3}{2} L_{opt}$

■ Heuristic algorithms

- Constructive heuristic – Nearest Neighborhood (NN) algorithm
- 2-Opt – local search algorithm proposed by Croes 1958
- Lin-Kernighan (LK) heuristic (e.g., **LKH**)

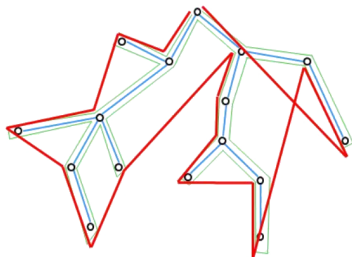
E.g., Helsgaun's implementation of the LK heuristic
<http://www.akira.ruc.dk/~keld/research/LKH>

■ Soft-computing techniques, e.g.,

- Variable Neighborhood Search (**VNS**), Greedy Randomized Adaptive Search Procedures (**GRASP**)
- Evolutionary approaches
- **Unsupervised learning**

MST-based Approximation Algorithm to the TSP

- Minimum Spanning Tree heuristic
 1. Compute the MST (denoted T) of the input graph G
 2. Construct a graph H by doubling every edge of T
 3. Shortcut repeated occurrences of a vertex in the tour



- For the triangle inequality, the length of such a tour L is

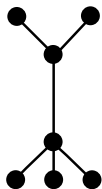
$$L \leq 2L_{optimal},$$

where $L_{optimal}$ is the cost of the optimal solution of the TSP

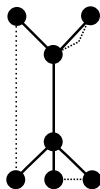
Christofides's Algorithm to the TSP

■ Christofides's algorithm

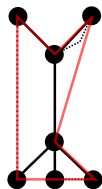
1. Compute the MST of the input graph G
2. Compute the minimal matching on the odd-degree vertices
3. Shortcut a traversal of the resulting Eulerian graph



MST



Matching



Final tour

- For the triangle inequality, the length of such a tour L is

$$L \leq \frac{3}{2} L_{optimal},$$

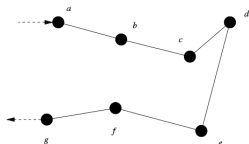
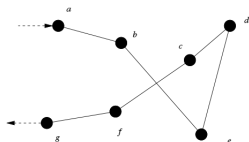
where $L_{optimal}$ is the cost of the optimal solution of the TSP

Length of the MST is $\leq L_{optimal}$

Sum of lengths of the edges in the matching $\leq \frac{1}{2} L_{optimal}$

2-Opt Heuristic

- Use a construction heuristic to create an initial route
 - NN algorithm, cheapest insertion, farther insertion
- Repeat until no improvement is made
 - Determine swapping that can shorten the tour (i, j) for $1 \leq i \leq n$ and $i + 1 \leq j \leq n$
 - route[0] to route[i-1]
 - route[i] to route[j] in reverse order
 - route[j] to route[end]
 - Determine length of the route
 - Update the current route if the length is shorter than the existing solution

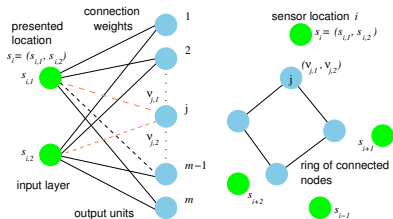


Unsupervised Learning based Solution of the TSP

- Sensor locations $S = \{s_1, \dots, s_n\}$, $s_i \in \mathbb{R}^2$; Neurons $\mathcal{N} = (\nu_1, \dots, \nu_m)$, $\nu_i \in \mathbb{R}^2$, $m = 2.5n$
 - Learning gain σ ; epoch counter i ; gain decreasing rate $\alpha = 0.1$; learning rate $\mu = 0.6$
1. $\mathcal{N} \leftarrow$ init ring of neurons as a small ring around some $s_j \in S$, e.g., a circle with radius 0.5
 2. $i \leftarrow 0$; $\sigma \leftarrow 12.41n + 0.06$;
 3. $I \leftarrow \emptyset$ // clear inhibited neurons
 4. **foreach** $s \in \Pi(S)$ (a permutation of S)
 - 4.1 $\nu^* \leftarrow \operatorname{argmin}_{\nu \in \mathcal{N} \setminus I} \|\nu, s\|$
 - 4.2 **foreach** ν in d neighborhood of ν^*

$$\nu \leftarrow \nu + \mu f(\sigma, d)(s - \nu)$$

$$f(\sigma, d) = \begin{cases} e^{-\frac{d^2}{\sigma^2}} & \text{for } d < 0.2m, \\ 0 & \text{otherwise,} \end{cases}$$
 - 4.3 $I \leftarrow I \cup \{\nu^*\}$ // inhibit the winner
 5. $\sigma \leftarrow (1 - \alpha)\sigma$; $i \leftarrow i + 1$;
 6. If (**termination condition** is not satisfied) Goto Step 3; Otherwise retrieve solution

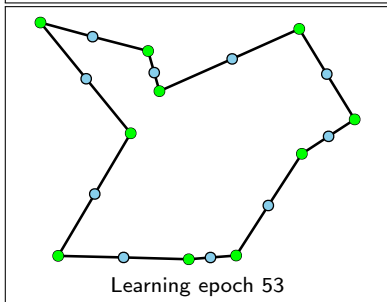
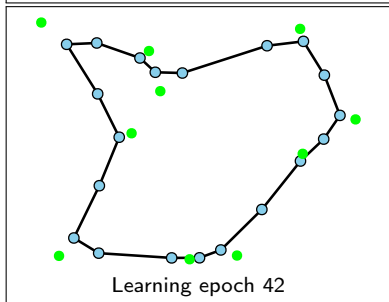
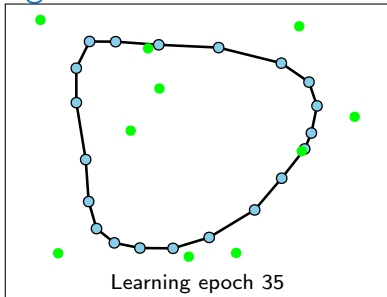
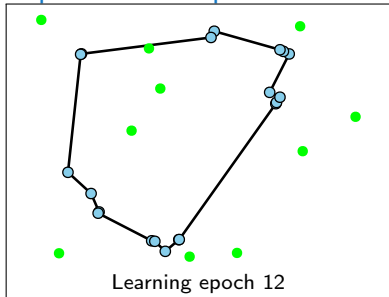


Termination condition can be

- Maximal number of learning epochs $i \leq i_{max}$, e.g., $i_{max} = 120$
- Winner neurons are negligibly close to sensor locations, e.g., less than 0.001

Somhom, S., Modares, A., Enkawa, T. (1999): [Competition-based neural network for the multiple travelling salesmen problem with minmax objective](#). Computers & Operations Research.
 Faigl, J. et al. (2011): [An application of the self-organizing map in the non-Euclidean Traveling Salesman Problem](#). Neurocomputing.

Example of Unsupervised Learning for the TSP



Traveling Salesman Problem with Neighborhoods (TSPN)

- Instead visiting a particular location $\mathbf{s} \in S$, $\mathbf{s} \in \mathbb{R}^2$ we can request to visit, e.g., a region $r \subset \mathbb{R}^2$ to save travel cost, i.e., visit regions $R = \{r_1, \dots, r_n\}$
- The TSP becomes the **TSP with Neighborhoods (TSPN)** where it is necessary, in addition to the determination of the order of visits Σ , determine suitable locations $P = \{\mathbf{p}_1, \dots, \mathbf{p}_n\}$, $\mathbf{p}_i \in r_i$, of visits to R
- The problem is a combination of combinatorial optimization to determine Σ with continuous optimization to determine P

$$\text{minimize}_{\Sigma, P} \quad L = \left(\sum_{i=1}^{n-1} c(\mathbf{p}_{\sigma_i}, \mathbf{p}_{\sigma_{i+1}}) \right) + c(\mathbf{p}_{\sigma_n}, \mathbf{p}_{\sigma_1})$$

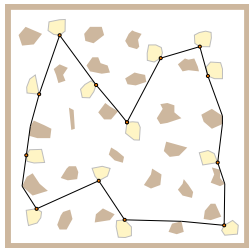
subject to

$$\begin{aligned} R &= \{r_1, \dots, r_n\}, r_i \subset \mathbb{R}^2 \\ P &= \{\mathbf{p}_1, \dots, \mathbf{p}_n\}, \mathbf{p}_i \in r_i \\ \Sigma &= (\sigma_1, \dots, \sigma_n), 1 \leq \sigma_i \leq n, \\ &\sigma_i \neq \sigma_j \text{ for } i \neq j \\ \text{Foreach } r_i \in R &\text{ there is } \mathbf{p}_i \in r_i \end{aligned}$$

(2)

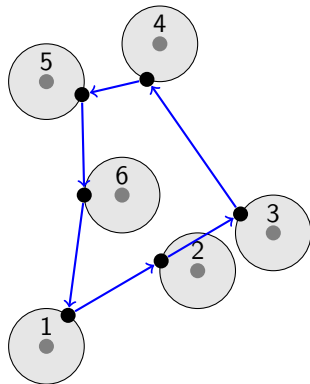
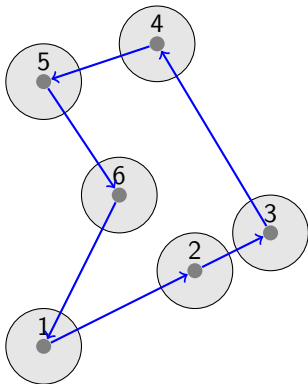
In general, TSPN is APX-hard, and cannot be approximated to within a factor $2 - \epsilon$, $\epsilon > 0$, unless P=NP.

Safra, S., Schwartz, O. (2006)



Traveling Salesman Problem with Neighborhoods (TSPN)

- Euclidean TSPN with disk shaped δ -neighborhoods – **Close Enough TSP (CETSP)**
- Sequence of visits to the regions with particular locations of the visit

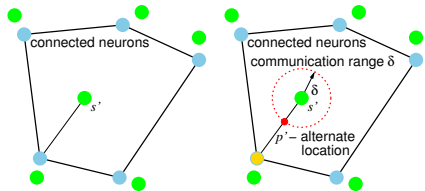
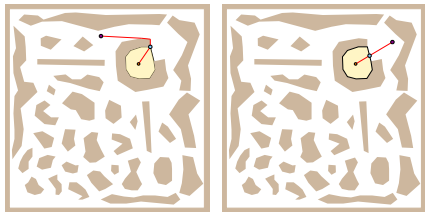


Approaches to the TSPN

- A direct solution of the TSPN – approximation algorithms and heuristics
- **Decoupled approach**
 - E.g., using evolutionary techniques or unsupervised learning
 - 1. Determine sequence of visits Σ independently on the locations P
 - E.g., as the TSP for centroids of the (convex) regions R
 - 2. For the sequence Σ determine the locations P to minimize the total tour length, e.g.,
 - Touring polygon problem (TPP)
 - Sampling possible locations and use a forward search for finding the best locations
 - Continuous optimization such as hill-climbing
 - E.g., **Local Iterative Optimization** (LIO), Váňa & Faigl (IROS 2015)
- Sampling-based approaches
 - For each region, sample possible locations of visits into a discrete set of locations for each region
 - The problem can be then formulated as the **Generalized Traveling Salesman Problem (GTSP)**
- Euclidean TSPN with, e.g., disk-shaped δ neighborhoods (**CETSP**)
 - Simplified variant with regions as disks with radius δ – remote sensing with the δ communication range

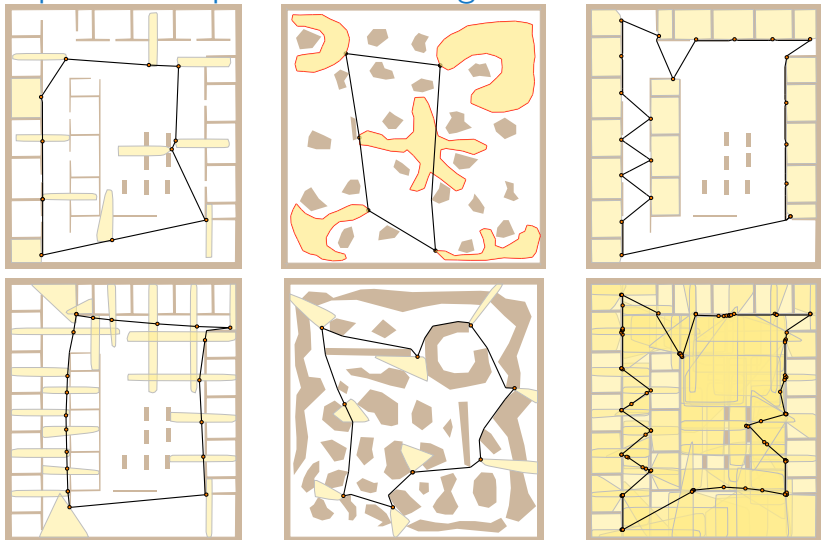
Unsupervised Learning for the TSPN

- In the unsupervised learning for the TSP, we can sample suitable sensing locations during winner selection
- We can use the centroid of the region for the shortest path computation from ν to the region r presented to the network
- Then, an intersection point of the path with the region can be used as an alternate location
- For the Euclidean TSPN with disk-shaped δ neighborhoods, we can compute the alternate location directly from the Euclidean distance



Faigl, J. et al. (2013): [Visiting convex regions in a polygonal map](#). Robotics and Autonomous Systems.

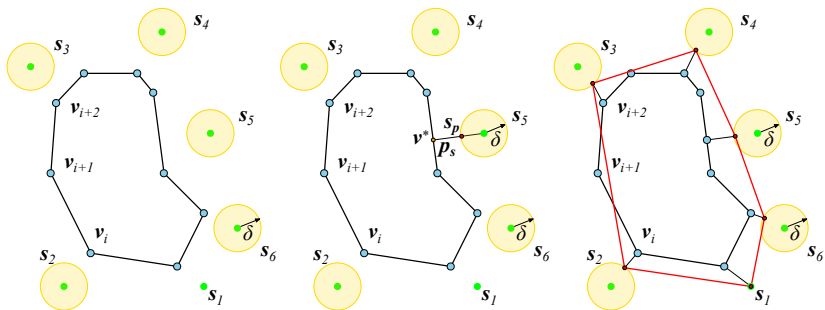
Example of Unsupervised Learning for the TSPN



It also provides solutions for non-convex regions, overlapping regions, and coverage problems.

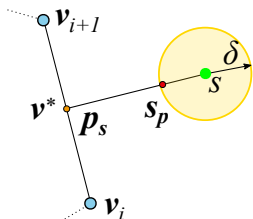
Growing Self-Organizing Array (GSOA)

- The GSOA is an array of nodes $\mathcal{N} = \{\nu_1, \dots, \nu_M\}$ that evolve in the problem space using unsupervised learning
- The array adapts to each $s \in S$ (in a random order) and for each s a new winner node ν^* is determined together with the corresponding s_p , such that $\|(s_p, \mathbf{s})\| \leq \delta$
- The winner and its neighborhoods are adapted (moved) towards s_p
- After the adaptation to all $s \in S$, each s has its ν and s_p , and the array defines the sequence Σ and the requested waypoints P

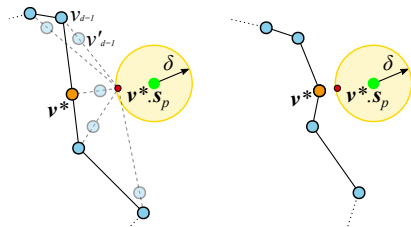


GSOA – Winner Selection and Its Adaptation

- Selecting winner node ν^* for s and its waypoint s_p



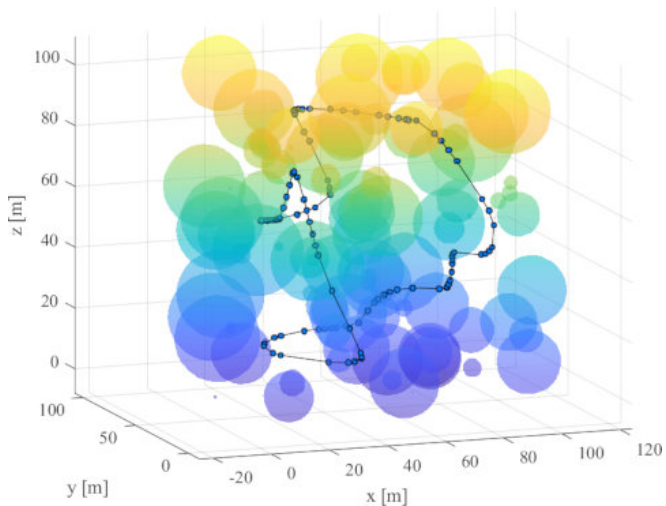
- Adaptation of the winner and its neighborhoods



- For each $s \in S$, we create new node ν^* , and therefore, all not winning nodes are removed after processing all locations in S (one learning epoch) to balance the number of nodes in the GSOA
- After each learning epoch, the GSOA encodes a feasible solution of the CETSP
- The power of adaptation is decreasing using a cooling schedule after each learning epoch
- The GSOA converges to a stable solution in tens of epochs

Faigl, J. (2018): [GSOA: Growing Self-Organizing Array - Unsupervised learning for the Close-Enough Traveling Salesman Problem and other routing problems](#). *Neurocomputing* 312: 120-134 (2018).

GSOA Evolution in solving the 3D CETSP



Solving the TSPN as the TPP – Iterative Refinement

- Let the sequence of n polygon regions be $R = (r_1, \dots, r_n)$

Li, F., Klette, R.: Approximate algorithms for touring a sequence of polygons. 2008

- Sampling the polygons into a discrete set of points and determine all shortest paths between each sampled points in the sequence of the regions visits
E.g., using visibility graph

- Initialization:* Construct an initial touring polygons path using a sampled point of each region

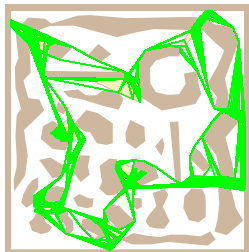
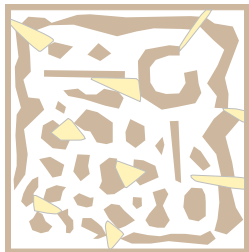
Let the path be defined by $P = (p_1, p_2, \dots, p_n)$, where $p_i \in r_i$ and $L(P)$ be the length of the shortest path induced by P

- Refinement:* For $i = 1, 2, \dots, n$

- Find $p_i^* \in r_i$ minimizing the length of the path $d(p_{i-1}, p_i^*) + d(p_i^*, p_{i+1})$, where $d(p_k, p_l)$ is the path length from p_k to p_l , $p_0 = p_n$, and $p_{n+1} = p_1$
- If the total length of the current path over point p_i^* is shorter than over p_i , replace the point p_i by p_i^*

- Compute path length L_{new} using the refined points
- Termination condition:* If $L_{new} - L < \epsilon$ Stop the refinement. Otherwise $L \leftarrow L_{new}$ and go to Step 3

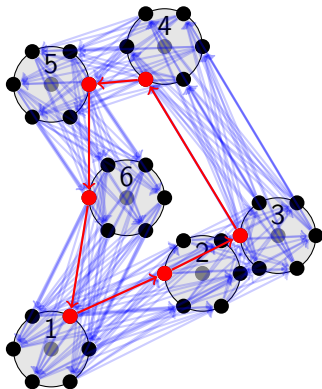
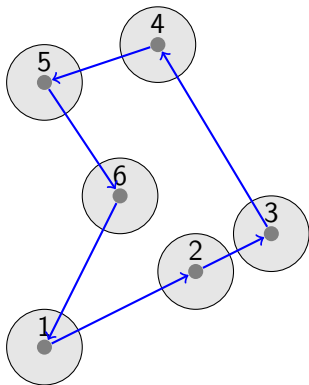
- Final path construction:* use the last points and construct the path using the shortest paths among obstacles between two consecutive points



Sampling-based Decoupled Solution of the TSPN

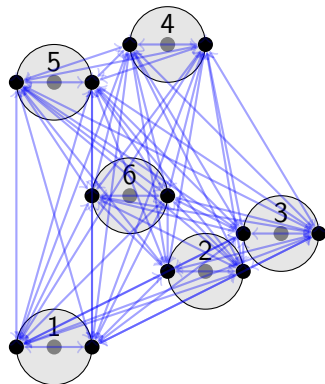
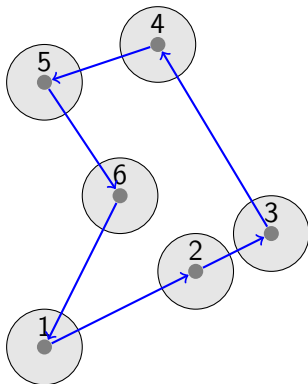
- Sample each neighborhood with, e.g., $k = 6$ samples
- Determine sequence of visits, e.g., by a solution of the ETSP for the centroids of the regions
- Finding the shortest tour takes in a forward search graph $\mathcal{O}(nk^3)$ for nk^2 edges in the sequence

Trying each of the k possible starting locations



Sampling-based Solution of the TSPN

- For an unknown sequence of the visits to the regions, there are $\mathcal{O}(n^2k^2)$ possible edges
- Finding the shortest path is NP-hard, we need to determine the sequence of visits, which is the solution of the TSP



The discrete variant of the TSPN can be formulated as the GTSP

Generalized Traveling Salesman Problem (GTSP)

- For sampled neighborhoods into discrete sets of locations, we can formulate the problem as the **Generalized Traveling Salesman Problem (GTSP)**

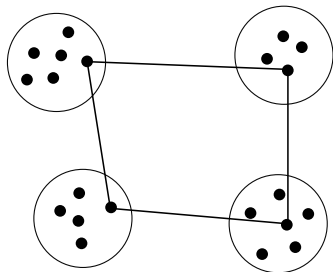
Also known as the **Set TSP** or **Covering Salesman Problem**, etc.

- For a set of n sets $S = \{S_1, \dots, S_n\}$, each with particular set of locations (nodes) $S_i = \{s_1^i, \dots, s_{n_i}^i\}$
- The problem is to determine the shortest tour visiting each set S_i , i.e., determining the order Σ of visits to S and a particular locations $s^i \in S_i$ for each $S_i \in S$

$$\text{minimize } \Sigma \quad L = \left(\sum_{i=1}^{n-1} c(s^{\sigma_i}, s^{\sigma_{i+1}}) \right) + c(s^{\sigma_n}, s^{\sigma_1})$$

$$\text{subject to} \quad \Sigma = (\sigma_1, \dots, \sigma_n), 1 \leq \sigma_i \leq n, \sigma_i \neq \sigma_j \text{ for } i \neq j$$

$$s^{\sigma_i} \in S_{\sigma_i}, S_{\sigma_i} = \{s_1^{\sigma_i}, \dots, s_{n_{\sigma_i}}^{\sigma_i}\}, S_{\sigma_i} \in S$$

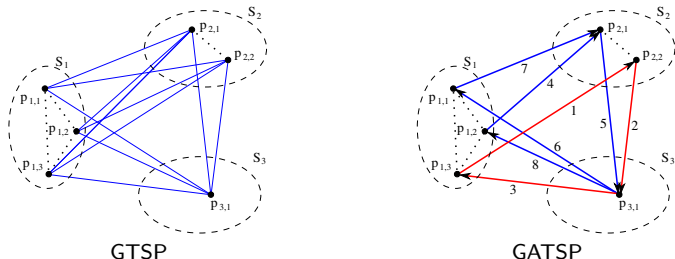


- In addition to exact, e.g., ILP-based, solution, heuristic algorithms are

- GLKH** – <http://akira.ruc.dk/~keld/research/GLKH/>
Helsgaun, K (2015), [Solving the Equality Generalized Traveling Salesman Problem Using the Lin-Kernighan-Helsgaun Algorithm](#). Mathematical Programming Computation.
- GLNS** – <https://ece.uwaterloo.ca/~s12smith/GLNS> (in Julia)
Smith, S. L., Imeson, F. (2017), [GLNS: An effective large neighborhood search heuristic for the Generalized Traveling Salesman Problem](#). Computers and Operations Research.

Transformation of the GTSP to the Asymmetric TSP

- The Generalized TSP can be transformed into the Asymmetric TSP that can be then solved, e.g., by LKH or exactly using Concorde with further transformation of the problem to the TSP



- A transformation of the GTSP to the ATSP has been proposed by Noon and Bean in 1993, and it is called as the **Noon-Bean Transformation**
 - Noon, C.E., Bean, J.C. (1993), [An efficient transformation of the generalized traveling salesman problem](#). *INFOR: Information Systems and Operational Research*.
 - Ben-Arieg, et al. (2003), [Transformations of generalized ATSP into ATSP](#). *Operations Research Letters*.

Noon-Bean Transformation

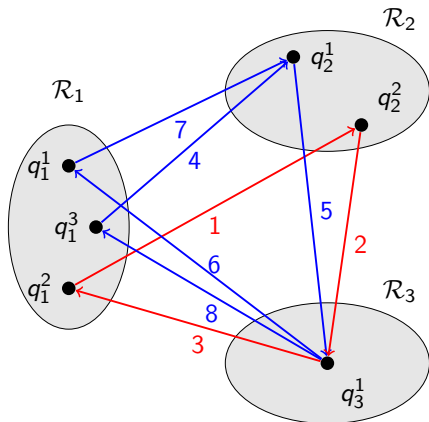
- Noon-Bean transformation to transfer GTSP to ATSP

- Modify weight of the edges (arcs) such that the optimal ATSP tour visits all vertices of the same cluster before moving to the next cluster

- Adding a large constant M to the weights of arcs connecting the clusters, e.g., a sum of the n heaviest edges
- Ensure visiting all vertices of the cluster in prescribed order, i.e., creating zero-length cycles within each cluster

- The transformed ATSP can be further transformed to the TSP

- For each vertex of the ATSP created 3 vertices in the TSP, i.e., it increases the size of the problem **three times**

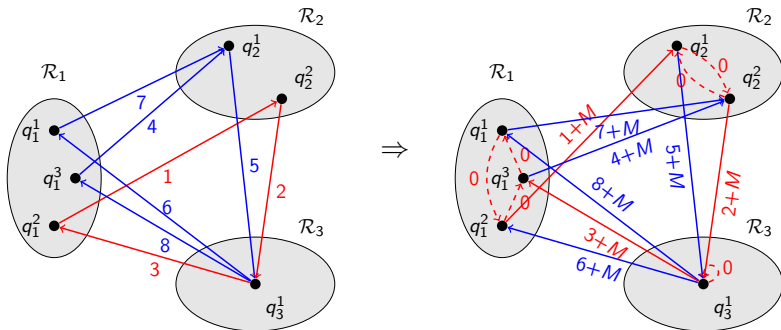


Noon, C.E., Bean, J.C. (1993), [An efficient transformation of the generalized traveling salesman problem](#). INFOR: Information Systems and Operational Research.

Example – Noon-Bean transformation (GATSP to ATSP)

1. Create a zero-length cycle in each set and set all other arcs to ∞ (or $2M$)
 To ensure all vertices of the cluster are visited before leaving the cluster

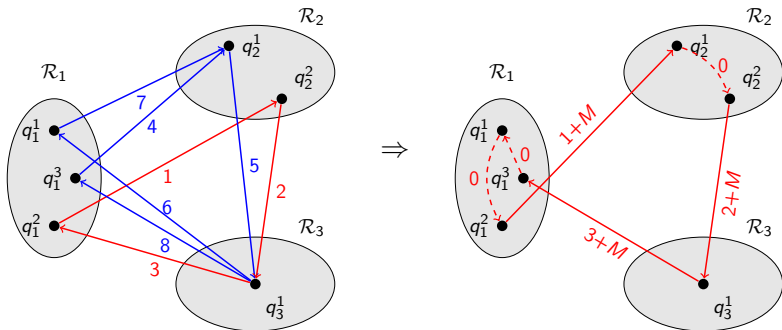
2. For each edge (q_i^m, q_j^n) create an edge (q_i^m, q_j^{n+1}) with a value increased by sufficiently large M
 To ensure visit of all vertices in a cluster before the next cluster



Example – Noon-Bean transformation (GATSP to ATSP)

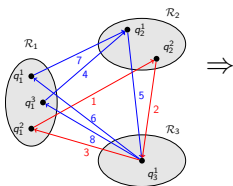
1. Create a zero-length cycle in each set and set all other arcs to ∞ (or $2M$)
 To ensure all vertices of the cluster are visited before leaving the cluster

2. For each edge (q_i^m, q_j^n) create an edge (q_i^m, q_j^{n+1}) with a value increased by sufficiently large M
 To ensure visit of all vertices in a cluster before the next cluster



Noon-Bean transformation – Matrix Notation

- 1. Create a zero-length cycle in each set; and 2. for each edge (q_i^m, q_j^n) create an edge (q_i^m, q_j^{n+1}) with a value increased by sufficiently large M



	q_1^1	q_1^2	q_1^3	q_2^1	q_2^2	q_3^1
q_1^1	∞	∞	∞	7	–	–
q_1^2	∞	∞	∞	–	1	–
q_1^3	∞	∞	∞	4	–	–
q_2^1	–	–	–	∞	∞	5
q_2^2	–	–	–	∞	∞	2
q_3^1	6	3	8	–	–	∞

∞ represents there are not edges inside the same set; and '–' denotes unused edge

Original GATSP

	q_1^1	q_1^2	q_1^3	q_2^1	q_2^2	q_3^1
q_1^1	∞	∞	∞	7	–	–
q_1^2	∞	∞	∞	–	1	–
q_1^3	∞	∞	∞	4	–	–
q_2^1	–	–	–	∞	∞	5
q_2^2	–	–	–	∞	∞	2
q_3^1	6	3	8	–	–	∞

Transformed ATSP

	q_1^1	q_1^2	q_1^3	q_2^1	q_2^2	q_3^1
q_1^1	∞	0	∞	–	$7+M$	–
q_1^2	∞	∞	0	$1+M$	–	–
q_1^3	0	∞	∞	–	$4+M$	–
q_2^1	–	–	–	∞	0	$5+M$
q_2^2	–	–	–	0	∞	$2+M$
q_3^1	$8+M$	$6+M$	$3+M$	–	–	0

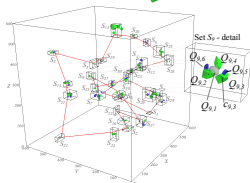
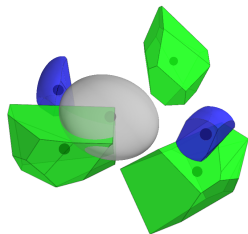
Noon-Bean Transformation – Summary

- It transforms the GATSP into the ATSP which can be further
 - Solved by existing solvers, e.g., the Lin-Kernighan heuristic algorithm (LKH) <http://www.akira.ruc.dk/~keld/research/LKH>
 - The ATSP can be further transformed into the TSP and solve it optimally, e.g., by the Concorde solver <http://www.tsp.gatech.edu/concorde.html>
- It runs in $\mathcal{O}(k^2n^2)$ time and uses $\mathcal{O}(k^2n^2)$ memory, where n is the number of sets (regions) each with up to k samples
- The transformed ATSP problem contains kn vertices

Noon, C.E., Bean, J.C. (1993), [An efficient transformation of the generalized traveling salesman problem](#). INFOR: Information Systems and Operational Research.

Generalized TSP with Neighborhoods (GTSPN)

- GTSPN is a multi-goal path planning problem to determine a cost-efficient path to visit a set of 3D regions
- Motivated by instances with redundant robotic manipulators, inspection tasks with multiple possible views
- GTSPN stands to determine a **shortest path** visiting each of the given n neighborhood sets $\mathcal{S} = \{S_1, \dots, S_n\}$ where each set $S_i \in \mathcal{S}$ consists of m_i regions $S_i = \{Q_{i,1}, \dots, Q_{i,m_i}\}$, $Q_{i,k} \subset \mathbb{R}^3$
- Region $Q_{i,j}$ can be ellipsoid, polyhedron, hybrid
- For mutually distance sets, the problem can be addressed as an instance of the **GTSP** using centroids of the regions with **continuous postprocessing optimization** (Centroids-GTSP+)
- Other methods are genetic algorithms and unsupervised learning

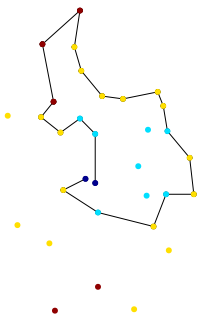


Jan Faigl, Petr Váňa and Jindřiška Deckerová, RA-L 2019

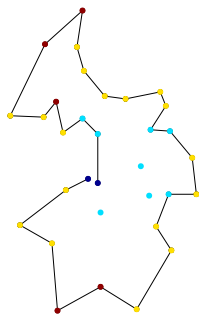
The Orienteering Problem (OP)

- The problem is to collect as many rewards as possible within the given **travel budget** (T_{max}), which is especially suitable for robotic vehicles such as multi-rotor Unmanned Aerial Vehicles (UAVs)
- The starting and termination locations are prescribed and can be different

The solution may not be a closed tour as in the TSP



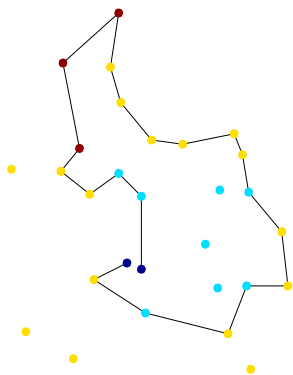
Travel budget $T_{max} = 50$,
Collected rewards $R = 190$



Travel budget $T_{max} = 75$,
Collected rewards $R = 270$

Orienteering Problem – Specification

- Let the given set of n sensors be located in \mathbb{R}^2 with the locations $S = \{\mathbf{s}_1, \dots, \mathbf{s}_n\}$, $\mathbf{s}_i \in \mathbb{R}^2$
- Each sensor \mathbf{s}_i has an associated score ζ_i characterizing the reward if data from \mathbf{s}_i are collected
- The vehicle is operating in \mathbb{R}^2 , and the travel cost is the Euclidean distance
- Starting and final locations are prescribed
- We aim to determine a subset of k locations $S_k \subseteq S$ that maximizes the sum of the collected rewards while the travel cost to visit them is below T_{max}



The **Orienteering Problem (OP)** combines two NP-hard problems:

- **Knapsack problem** in determining the most valuable locations $S_k \subseteq S$
- **Travel Salesman Problem (TSP)** in determining the shortest tour

Orienteering Problem – Optimization Criterion

- Let $\Sigma = (\sigma_1, \dots, \sigma_k)$ be a permutation of k sensor labels, $1 \leq \sigma_i \leq n$ and $\sigma_i \neq \sigma_j$ for $i \neq j$
- Σ defines a tour $T = (\mathbf{s}_{\sigma_1}, \dots, \mathbf{s}_{\sigma_k})$ visiting the selected sensors S_k
- Let the start and end points of the tour be $\sigma_1 = 1$ and $\sigma_k = n$
- The **Orienteering problem (OP)** is to determine the number of sensors k , the subset of sensors S_k , and their sequence Σ such that

$$\begin{aligned}
 & \underset{k, S_k, \Sigma}{\text{maximize}} & R &= \sum_{i=1}^k \zeta_{\sigma_i} \\
 & \text{subject to} & \sum_{i=2}^k |(\mathbf{s}_{\sigma_{i-1}}, \mathbf{s}_{\sigma_i})| &\leq T_{\max} \text{ and} \\
 & & \mathbf{s}_{\sigma_1} &= \mathbf{s}_1, \mathbf{s}_{\sigma_k} = \mathbf{s}_n.
 \end{aligned} \tag{3}$$

The OP combines the problem of determining the most valuable locations S_k with finding the shortest tour T visiting the locations S_k . It is NP-hard, since for $\mathbf{s}_1 = \mathbf{s}_n$ and particular S_k it becomes the TSP.

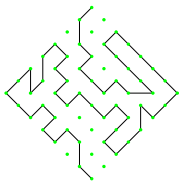
Existing Heuristic Approaches for the OP

- The **Orienteering Problem** has been addressed by several approaches, e.g.,

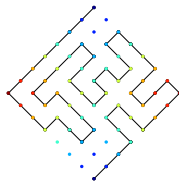
RB	4-phase heuristic algorithm proposed in [3]
PL	Results for the method proposed by Pillai in [2]
CGW	Heuristic algorithm proposed in [1]
GLS	Guided local search algorithm proposed in [4]

-
- [1] I.-M. Chao, B. L. Golden, and E. A. Wasil.
A fast and effective heuristic for the orienteering problem.
European Journal of Operational Research, 88(3):475–489, 1996.
- [2] R. S. Pillai.
The traveling salesman subset-tour problem with one additional constraint (TSSP+ 1).
Ph.D. thesis, The University of Tennessee, Knoxville, TN, 1992.
- [3] R. Ramesh and K. M. Brown.
An efficient four-phase heuristic for the generalized orienteering problem.
Computers & Operations Research, 18(2):151–165, 1991.
- [4] P. Vansteenwegen, W. Souffriau, G. V. Berghe, and D. V. Oudheusden.
A guided local search metaheuristic for the team orienteering problem.
European Journal of Operational Research, 196(1):118–127, 2009.

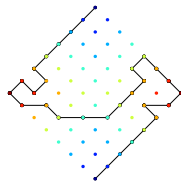
OP Benchmarks – Example of Solutions



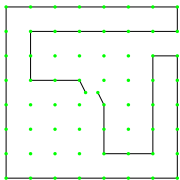
$T_{max}=80, R=1248$



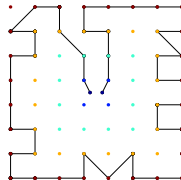
$T_{max}=80, R=1278$



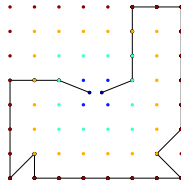
$T_{max}=45, R=756$



$T_{max}=95, R=1395$



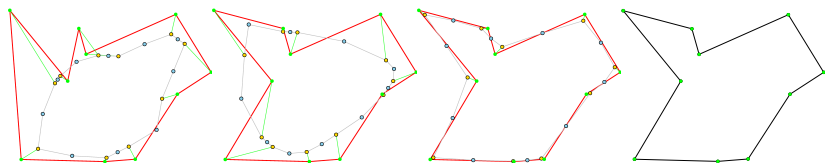
$T_{max}=95, R=1335$



$T_{max}=60, R=845$

Unsupervised Learning for the OP 1/2

- A solution of the OP is similar to the solution of the PC-TSP and TSP
- We need to satisfy the limited travel budget T_{max} , which needs the final tour over the sensing locations
- During the unsupervised learning, the winners are associated with the particular sensing locations, which can be utilized to determine the tour as a solution of the OP represented by the network:



Learning epoch 7

Learning epoch 55

Learning epoch 87

Final solution

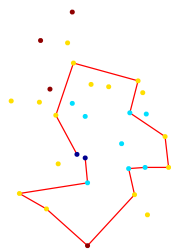
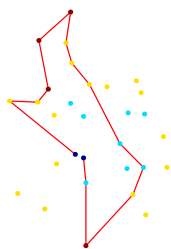
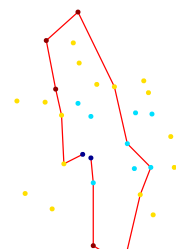
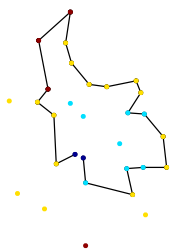
- This is utilized in the **conditional adaptation** of the network towards the sensing location and the adaptation is performed only if the tour represented by the network after the adaptation would satisfy T_{max}

Unsupervised Learning for the OP 2/2

- The winner selection for $s' \in S$ is conditioned according to T_{max}
 - The network is adapted only if the tour T_{win} represented by the current winners would be shorter or equal than T_{max}

$$\mathcal{L}(T_{win}) - |(s_{\nu_p}, s_{\nu_n})| + |(s_{\nu_p}, s')| + |(s', s_{\nu_n})| \leq T_{max}$$

- The unsupervised learning performs a *stochastic search* steered by the rewards and the length of the tour to be below T_{max}

Epoch 155, $R=150$ Epoch 201, $R=135$ Epoch 273, $R=125$ Final solution, $R=190$

Comparison with Existing Algorithms for the OP

- Standard benchmark problems for the Orienteering Problem represent various scenarios with several values of T_{max}
- The results (rewards) found by different OP approaches presented as the average ratios (and standard deviations) to the best-known solution

Instances of the Tsiligrides problems

Problem Set	RB	PL	CGW	Unsupervised Learning
Set 1, $5 \leq T_{max} \leq 85$	0.99/0.01	1.00/0.01	1.00/0.01	1.00/0.01
Set 2, $15 \leq T_{max} \leq 45$	1.00/0.02	0.99/0.02	0.99/0.02	0.99/0.02
Set 3, $15 \leq T_{max} \leq 110$	1.00/0.00	1.00/0.00	1.00/0.00	1.00/0.00

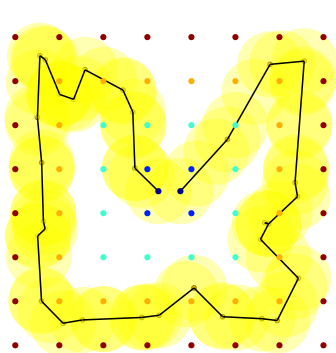
Diamond-shaped (Set 64) and Square-shaped (Set 66) test problems

Problem Set	RB [†]	PL	CGW	Unsupervised Learning
Set 64, $5 \leq T_{max} \leq 80$	0.97/0.02	1.00/0.01	0.99/0.01	0.97/0.03
Set 66, $15 \leq T_{max} \leq 130$	0.97/0.02	1.00/0.01	0.99/0.04	0.97/0.02

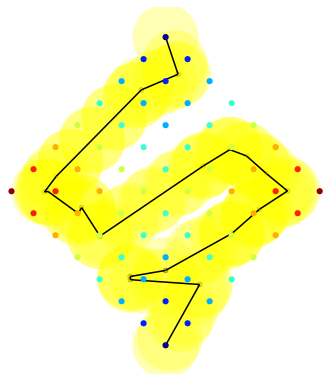
Required computational time is up to units of seconds, but for small problems tens or hundreds of milliseconds.

Orienteering Problem with Neighborhoods

- Similarly to the TSP with Neighborhoods and PC-TSPN we can formulate the **Orienteering Problem with Neighborhoods**.



$$T_{max}=60, \delta=1.5, R=1600$$

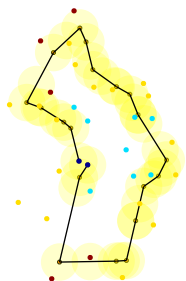


$$T_{max}=45, \delta=1.5, R=1344$$

Orienteering Problem with Neighborhoods

- Data collection using wireless data transfer allows to reliably retrieve data within some communication radius δ
 - Disk-shaped δ -neighborhood – **Close Enough OP (CEOP)**
- We need to determine the most suitable locations P_k such that

$$\begin{aligned} & \text{maximize}_{k, P_k, \Sigma} && R = \sum_{i=1}^k \zeta_{\sigma_i} \\ & \text{subject to} && \sum_{i=2}^k |(\mathbf{p}_{\sigma_{i-1}}, \mathbf{p}_{\sigma_i})| \leq T_{\max}, \\ & && |(\mathbf{p}_{\sigma_i}, \mathbf{s}_{\sigma_i})| \leq \delta, \quad \mathbf{p}_{\sigma_i} \in \mathbb{R}^2, \\ & && \mathbf{p}_{\sigma_1} = \mathbf{s}_1, \mathbf{p}_{\sigma_k} = \mathbf{s}_n. \end{aligned}$$



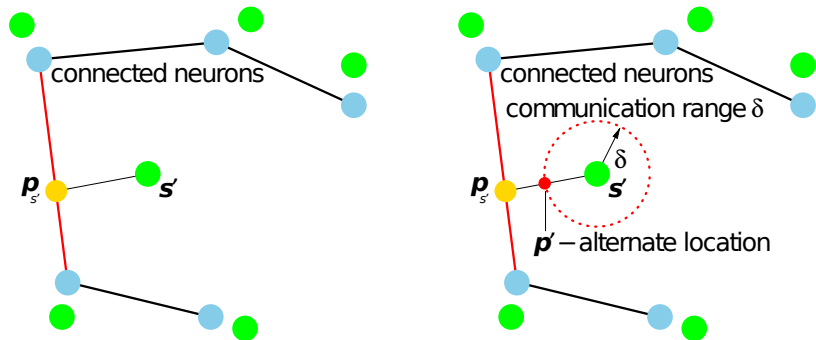
$$T_{\max} = 50, R = 270$$

Introduced by Best, Faigl, Fitch (IROS 2016, SMC 2016, IJCNN 2017)

- More rewards can be collected than for the OP formulation with the same travel budget T_{\max}

Generalization of the Unsupervised Learning to the Orienteering Problem with Neighborhoods

- The same idea of the alternate location as in the TSPN



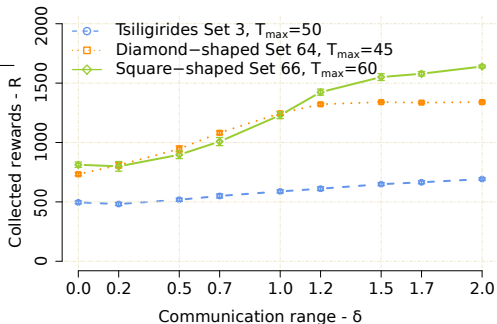
- The location p' for retrieving data from s' is determined as the alternate goal location during the conditioned winner selection

Influence of the δ -Sensing Distance

- Influence of increasing communication range to the sum of the collected rewards

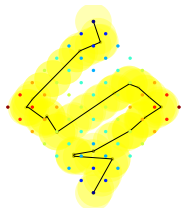
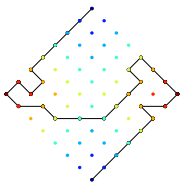
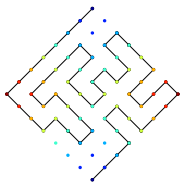
Problem	Solution of the OP	
	R_{best}	R_{SOM}
Set 3, $T_{max}=50$	520	510
Set 64, $T_{max}=45$	860	750
Set 66, $T_{max}=60$	915	845

- *Allowing to data reading within the communication range δ may significantly increases the collected rewards, while keeping the budget under T_{max}*



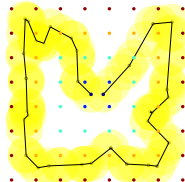
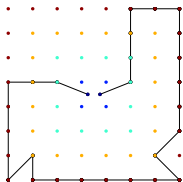
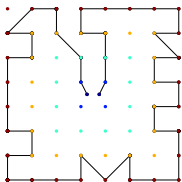
OP with Neighborhoods (OPN) – Example of Solutions

- Diamond-shaped problem Set 64 – SOM solutions for T_{max} and δ



$T_{max}=80, \delta=0.0, R=1278$ $T_{max}=45, \delta=0.0, R=756$ $T_{max}=45, \delta=1.5, R=1344$

- Square-shaped problem Set 66 – SOM solutions for T_{max} and δ



$T_{max}=95, \delta=0.0, R=1335$ $T_{max}=60, \delta=0.0, R=845$ $T_{max}=60, \delta=1.5, R=1600$

In addition to unsupervised learning, **Variable Neighborhood Search (VNS)** for the OP has been generalized to the OPN

Autonomous (Underwater) Data Collection

- Having a set of sensors (sampling stations), we aim to determine a cost-efficient path to retrieve data by autonomous underwater vehicles (AUVs) from the individual sensors

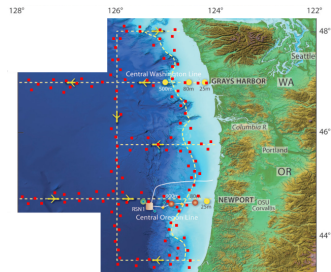
E.g., Sampling stations on the ocean floor

- The planning problem is a variant of the **Traveling Salesman Problem**

Two practical aspects of the data collection can be identified

1. Data from particular sensors may be of different importance
2. Data from the sensor can be retrieved using wireless communication

These two aspects (of general applicability) can be considered in the Prize-Collecting Traveling Salesman Problem (PC-TSP) and Orienteering Problem (OP) and their extensions with neighborhoods



Prize-Collecting Traveling Salesman Problem with Neighborhoods (PC-TSPN)

- Let n sensors be located in \mathbb{R}^2 at the locations $S = \{\mathbf{s}_1, \dots, \mathbf{s}_n\}$
- Each sensor has associated penalty $\xi(\mathbf{s}_i) \geq 0$ characterizing additional cost if the data are not retrieved from \mathbf{s}_i
- Let the data collecting vehicle operates in \mathbb{R}^2 with the motion cost $c(\mathbf{p}_1, \mathbf{p}_2)$ for all pairs of points $\mathbf{p}_1, \mathbf{p}_2 \in \mathbb{R}^2$
- The data from \mathbf{s}_i can be retrieved within δ distance from \mathbf{s}_i

PC-TSPN – Optimization Criterion

The **PC-TSPN** is a problem to

- **Determine a set of unique locations** $P = \{\mathbf{p}_1, \dots, \mathbf{p}_k\}$, $k \leq n$, $\mathbf{p}_i \in \mathbb{R}^2$, at which data readings are performed
- **Find a cost efficient tour** T visiting P such that the total cost $C(T)$ of T is minimal

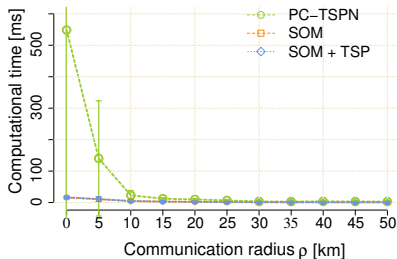
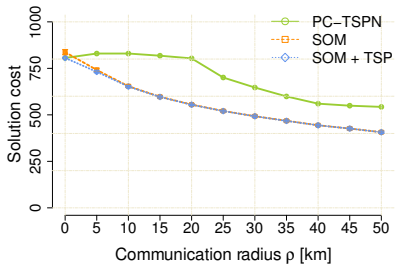
$$C(T) = \sum_{(\mathbf{p}_{l_i}, \mathbf{p}_{l_{i+1}}) \in T} c(\mathbf{p}_{l_i}, \mathbf{p}_{l_{i+1}}) + \sum_{\mathbf{s} \in S \setminus S_T} \xi(\mathbf{s}), \quad (4)$$

where $S_T \subseteq S$ are sensors such that for each $\mathbf{s}_i \in S_T$ there is \mathbf{p}_{l_j} on $T = (\mathbf{p}_{l_1}, \dots, \mathbf{p}_{l_{k-1}}, \mathbf{p}_{l_k})$ and $\mathbf{p}_{l_j} \in P$ for which $|(\mathbf{s}_i, \mathbf{p}_{l_j})| \leq \delta$.

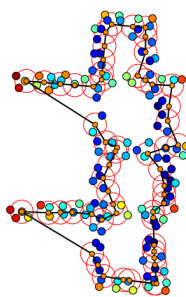
- PC-TSPN includes other variants of the TSP
 - for $\delta = 0$ it is the PC-TSP
 - for $\xi(\mathbf{s}_i) = 0$ and $\delta \geq 0$ it is the TSPN
 - for $\xi(\mathbf{s}_i) = 0$ and $\delta = 0$ it is the ordinary TSP

PC-TSPN – Example of Solution

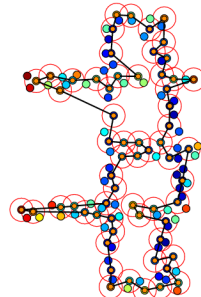
Ocean Observatories Initiative (OOI) scenario



SOM PC-TSPN



PC-TSPN



Faigl and Hollinger – (IROS 2014, TNNLS 2017)

Summary of the Lecture

Topics Discussed

- Data collection planning can be formulated as variants of
 - **Traveling Salesman Problem (TSP)**
 - **Orienteering Problem (OP)**
 - *Prize-Collecting Traveling Salesman Problem with Neighborhoods (PC-TSPN)*
- Exploiting the non-zero sensing range can be addressed as
 - **TSP with Neighborhoods (TSPN)** or specifically as the **Close Enough TSP (CETSP)** for disk shaped neighborhoods
 - **OP with Neighborhoods (OPN)** or the **Close Enough OP (CEOP)**
- Problems with continuous neighborhoods include **continuous optimization** that can be addressed by **sampling** the neighborhoods into discrete sets
 - **Generalized TSP** and **Set OP**
- Existing solutions include
 - Approximation algorithms and heuristics (combinatorial, unsupervised learning, evolutionary methods)
 - Sampling-based and decoupled approaches
 - ILP formulations for discrete problem variants (sampling-based approaches)
 - Transformation based approaches (GTSP→ATSP) / Noon-Bean transformation
 - Combinatorial heuristics such as **VNS** and **GRASP**
 - TSP can be solved by efficient heuristics such as **LKH**
- **Next: Data-collection planning with curvature-constrained vehicles**