演算法作業說明

Algorithm Assignment Instructions

劉恭銘 (M133040015@student.nsysu.edu.tw)
趙浚智 (M133040117@student.nsysu.edu.tw)
吳少棠 (M143040047@student.nsysu.edu.tw)
江凱逸 (M143040067@student.nsysu.edu.tw)
王昱人 (M143040090@student.nsysu.edu.tw)

Artificial Intelligence Lab@NSYSU

Algorithm Homework

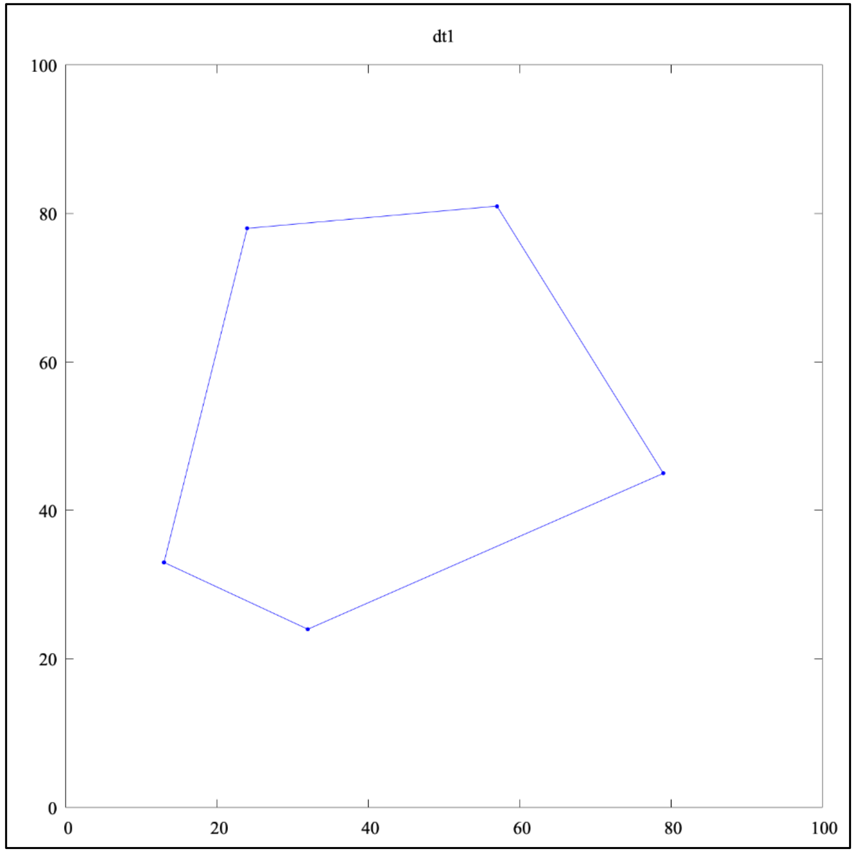
Travelling Salesman Problem introduction

Artificial Intelligence Lab@NSYSU

Travelling Salesman Problem (1/1)

- Travelling Salesman Problem (TSP)
  - Given a set of data points (cities) and the distances between them, find the shortest closed loop (tour) that visits every point exactly once.

City ID	[1, 2, 3, 4, 5]
X-coordinate	[13, 57, 79, 32, 24]
Y-coordinate	[33, 81, 45, 24, 78]



- Time Complexity O(N!)
- NP-complete

Homework3

Solving TSP with Dynamic Programming

Artificial Intelligence Lab@NSYSU

HW3 - Dynamic Programming (1/1)

- Suppose there are four points : 1, 2, 3, 4
- Distance : (1,4) = 10, (2,4) = 5, (3,4) = 4,

  (1,2) = 8, (1,3) = 6, (2,3) = 7

	1	**2**	**3**	**4**
1	0	8	6	10
2		0	7	5
3			0	4
4				0

Assume Node 1 is the default starting point. Each row represents the current start node, and each column represents the remaining nodes to be visited. e.g. (001) = (XX2)

	Assume starting from Node 1	**Visited 2**	**Visited 3**	**Visited 2,3**	**Visited 4**	**Visited 2,4**	**Visited 3,4**	**Visited 2,3,4**
	000	001	010	011	100	101	110	111
1								answer
2								
3								
4								

Howework4

Solving TSP with Ant colony optimization

Artificial Intelligence Lab@NSYSU

HW4 - Ant colony optimization (1/11)

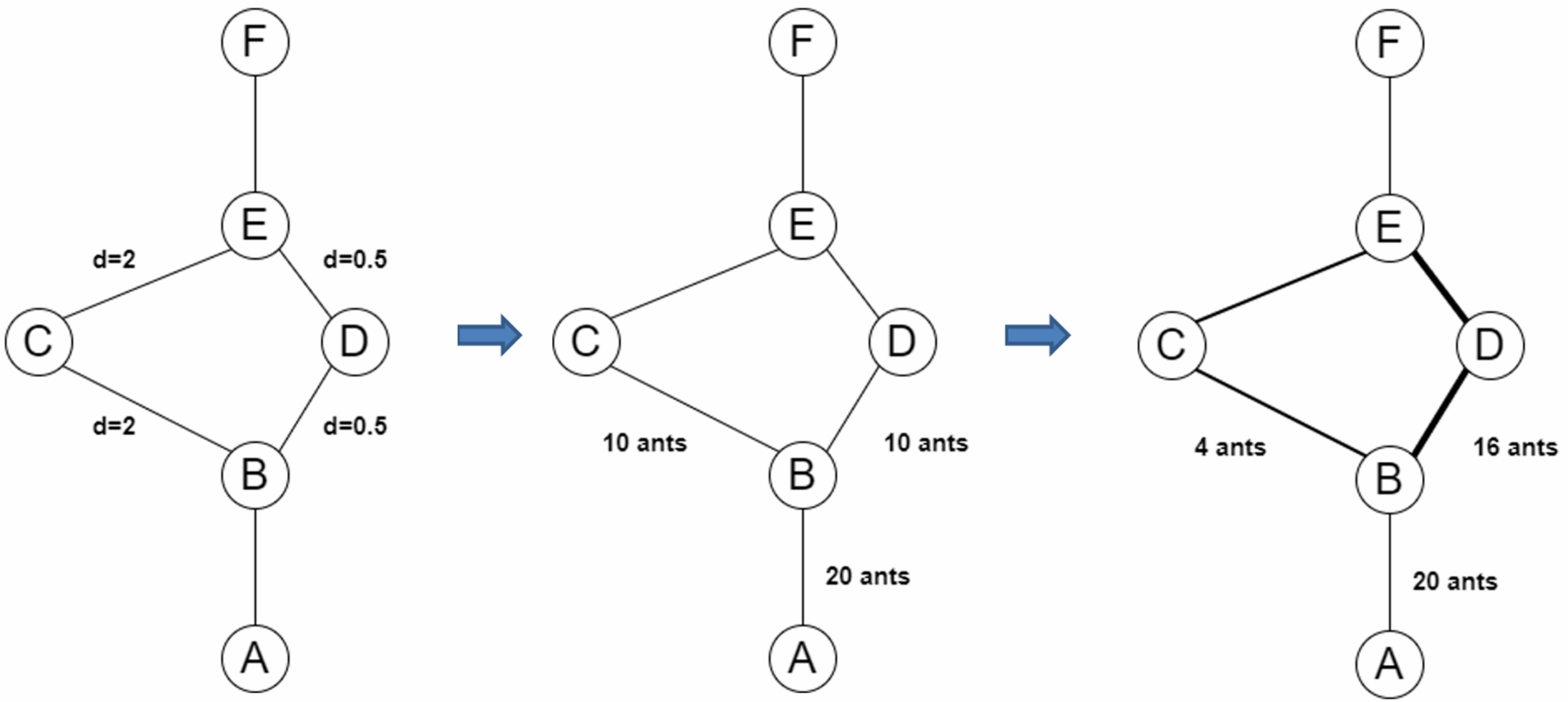
- What is Ant Colony Optimization[1] ?
  - Ant colony optimization takes inspiration from **ants behaviors**
  - Applicable to combinatorial optimization problems
    - ✓ Travelling Salesman Problem
    - ✓ Routing Problem

[1] M. Dorigo, M. Birattari and T. Stutzle, "Ant colony optimization," *IEEE Computational Intelligence Magazine*, vol. 1, no. 4, pp. 28-39, 2006

HW4 - Ant colony optimization (2/11)

- Inspiration : How ants find the shortest path
  - When ants find food, they deposit pheromones along their path.
  - Subsequent ants tend to choose paths with higher pheromone concentrations.
  - If a path is shorter, pheromones accumulate faster.
  - Over time, the entire colony naturally converges on the shortest path.
- Core Idea
  - Using pheromone table(τ) and route distance to construct solutions

HW4 - Ant colony optimization (3/11)



HW4 - Ant colony optimization (4/11)

- Ant Colony Optimization

Algorithm 1 The Ant Colony Optimization Metaheuristic

Set parameters, initialize pheromone trails

while termination condition not met **do**

  $ConstructAntSolutions$

  $ApplyLocalSearch\ (optional)$

  $UpdatePheromones$

endwhile

HW4 - Ant colony optimization (5/11)

- Set parameters, initialize pheromone trails
  - Parameter configuration before running ACO

Parameter	Definitions
run_times	Total number of algorithm runs
iteration	Maximum iterations per run
population_size	Number of ants (Population size)
alpha (α)	Pheromone importance factor
beta (β)	Heuristic Factor (1/distance)
rho (ρ)	Pheromone evaporation rate
Q	Constant

  - Each solution (or each ant) represents a specific TSP tour/path.
  - Each search performed by a single ant counts as one evaluation.
  - In each iteration, a colony composed of multiple ants performs the search.

HW4 - Ant colony optimization (6/11)

- $ConstructAntSolutions$

  Solutions are constructed step-by-step based on the pheromone table using the Roulette Wheel Selection method.

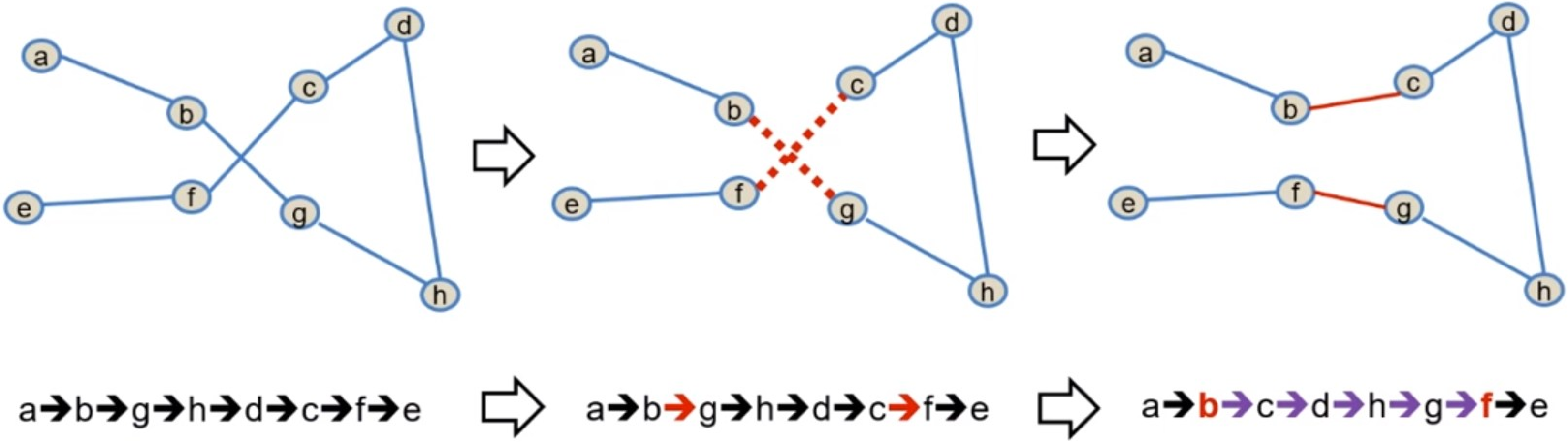
$$p_{ij}^{k} = \begin{cases} \dfrac{\tau_{ij}^{\alpha} \cdot \eta_{ij}^{\beta}}{\sum_{c_{il} \in \mathbf{N}(s^p)} \tau_{il}^{\alpha} \cdot \eta_{il}^{\beta}} & \text{if } c_{ij} \in \mathbf{N}(s^p), \\ 0 & \text{otherwise,} \end{cases}$$

$\tau: phermone,$
$\eta: heuristic\ information,$
$d: distance,$

$$\eta_{ij} = \frac{1}{d_{ij}},$$

HW4 - Ant colony optimization (7/11)

- $ApplyLocalSearch\ (optional)$: 2-OPT



HW4 - Ant colony optimization (8/11)

- $UpdatePheromones$

$$\tau_{ij} \leftarrow (1-\rho) \cdot \tau_{ij} + \sum_{k=1}^{m} \Delta\tau_{ij}^{k} ,$$

ρ: $pheromone\ evaporation\ rate$.

$$\Delta\tau_{ij}^{k} = \begin{cases} Q/L_k & \text{if ant } k \text{ used edge } (i, j) \text{ in its tour,} \\ 0 & \text{otherwise,} \end{cases}$$

HW4 - Ant colony optimization (9/11)

- Run vs. iteration vs. Evaluation:
  - Each "Run" needs initializing all ACO parameters (e.g., population, pheromone).
  - Each "Iteration" consists of every ant in the population finding a path once.
  - Each evaluation refers to assessing one ant’s complete path.
  - Evaluating one path increases the evaluation count by one.

```
for  0 ≤ r < run_times
    Initialization()
    for  0 ≤ i < iterations
        for  0 ≤ j < population_size
            ConstructAntSolution()
            Evaluation() // eval_count ++
        Update_pheromones()
```

HW4 - Ant colony optimization (10/11)

- Execution parameter setting:

Parameter	Definitions	Value
run_times	Total number of algorithm runs	30
iteration	Maximum iterations per run	Set by yourself
max_evaluation	Maximum evaluation times per run	10000*(number of city)
population_size	Number of ants (Population size)	Set by yourself
alpha (α)	Pheromone importance factor	Set by yourself
beta (β)	Heuristic Factor (1/distance)	Set by yourself
rho (ρ)	Pheromone evaporation rate	Set by yourself
Q	Constant	Set by yourself

HW4 - Ant colony optimization (11/11)

- Execution & Output Requirements:
  - ➢ Execute 30 independent runs for all test datasets.
  - ➢ Each run must not exceed the maximum number of evaluations.
  - ➢ Each dataset has a specific evaluation limit. Can limit the algorithm's evaluations using the following methods:
    - ✓ Set an additional parameter limit (evaluation_max).
    - ✓ Set a limit based on (iterations × population_size).
  - ➢ Upon completion, the following must be output:
    - ➢ The average best solution distance obtained over 30 runs.
    - ➢ The best solution found across all 30 runs.

Bonus Assignment

Solving TSP with Elastic Net

Artificial Intelligence Lab@NSYSU

Bonus Assignment – Elastic Net

- Please refer to the Elastic Net [2] paper and implement the Elastic Net algorithm to solve the TSP problem.
- Execution & Output Requirements:
  - Execute 30 independent runs for all test datasets.
  - Each run must not exceed the maximum evaluation limit.
  - Each dataset has its own evaluation limit. You may limit the evaluations using:
    - ✓ Setting an explicit parameter limit (evaluation_max).
    - ✓ Setting a limit based on iterations × population_size.
  - Upon completion, output:
    - ✓ The average best distance over 30 runs
    - ✓ The single best solution found across 30 runs.
  - Record the best path for every 100 evaluation, then **draw GIF** by all records.

[2] R. Durbin and D. Willshaw, “An analogue approach to the travelling salesman problem using an elastic net method,” *Nature*, vol. 326, no. 6114, pp. 689–691, 1987.

作業繳交規則
Assignment Submission Guidelines

Guidelines

- Submission Format：
  - Files must be compressed into：StudentID_hwx.zip (ex :b093040000_hw1.zip)
  - The archive must include the following (Do not include .exe files)
    ./b093040000_hwx/
    1. Source code (c or c++)
    2. dataset/
       - Containing output files and plot images for each dataset (ans.txt, fig.png)
    3. Plotting source code (any language).
       (Optional if plotting logic is already included in the C/C++ code).

- Submission Method: Course Website
- Online Submission Deadline: Dec 26, 23:59.
- On-site Demo Location: EC5009-1 (Please bring your own laptop).

Grading Criteria

Is the program able to execute correctly?	30% (If it cannot execute, the entire assignment will receive 0 points)
Is the answer correct?	20% (If the answer is wrong, you can get a maximum of 50 points.)
Is the structure and logic of the program correct?	20%
Is the output complete?	10%
Clear explanation of the program flow?(Verbal explanation or Comments)	10% (If plagiarism is found or you cannot explain the program flow, the entire assignment receives 0 points.)
Is the submission format correct?(File name and file format)	5%
Can input file name be dynamically read?(Input file name is not hard-coded)	5%

※ Partial credit is given for all items

Grading Criteria

- Ability to dynamically read specific files (Examples below):
  - Example 1: Input arguments via command line at runtime.

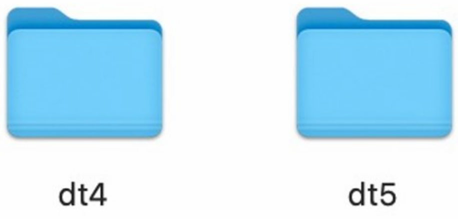
./main.exe <dataset> <output_name> <run_times> <evaluation_max> <population_size> <alpha> <beta> <rho> <Q>

  - Example 2: Prompted by the program after execution.

```
please enter file path:
```

Datasets & File Reading

- Total of 5 test datasets (each with a different number of cities). The 5th dataset (extra) is the bonus test data for ACO (HW4).
- The 5th dataset (extra) does not need to be executed for DP (HW3).
  - point.txt ：City Coordinates (ID, X, Y).
  - ans.txt ：Optimal solution (Provided only for the first two datasets).



Input Reading Example

1 13 33

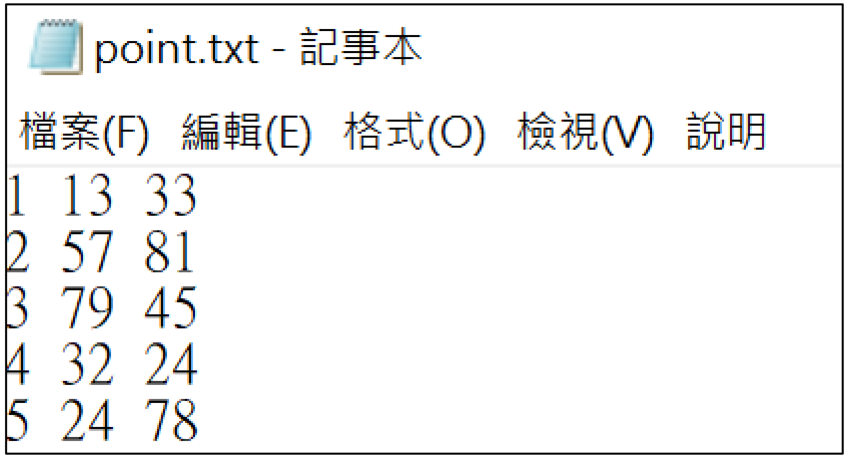
2 57 81

3 79 45

4 32 24

5 24 78

City ID	[1, 2, 3, 4, 5]
X-coordinate	[13, 57, 79, 32, 24]
Y-coordinate	[33, 81, 45, 24, 78]



point.txt

Output

- Output independently for each test dataset.
  - e.g. ans_dt01.txt, ans_dt02.txt, ans_dt03.txt
- Plot the found route map (e.g. fig.png)
- Output Regulations
  - Output the average best solution obtained from 30 runs (ACO).
  - The shortest path length found for that dataset.
  - Followed by the city indices in the order of the best path found.

Output

HW3

ans.txt - 記事本

檔案(F) 編輯(E) 格式(O) 檢視(V) 說明

```
distance: 194.153
1
4
3
2
5
```

HW4

```
mean distance: 194.153
distance: 194.153
1
4
3
2
5
```

ans.txt

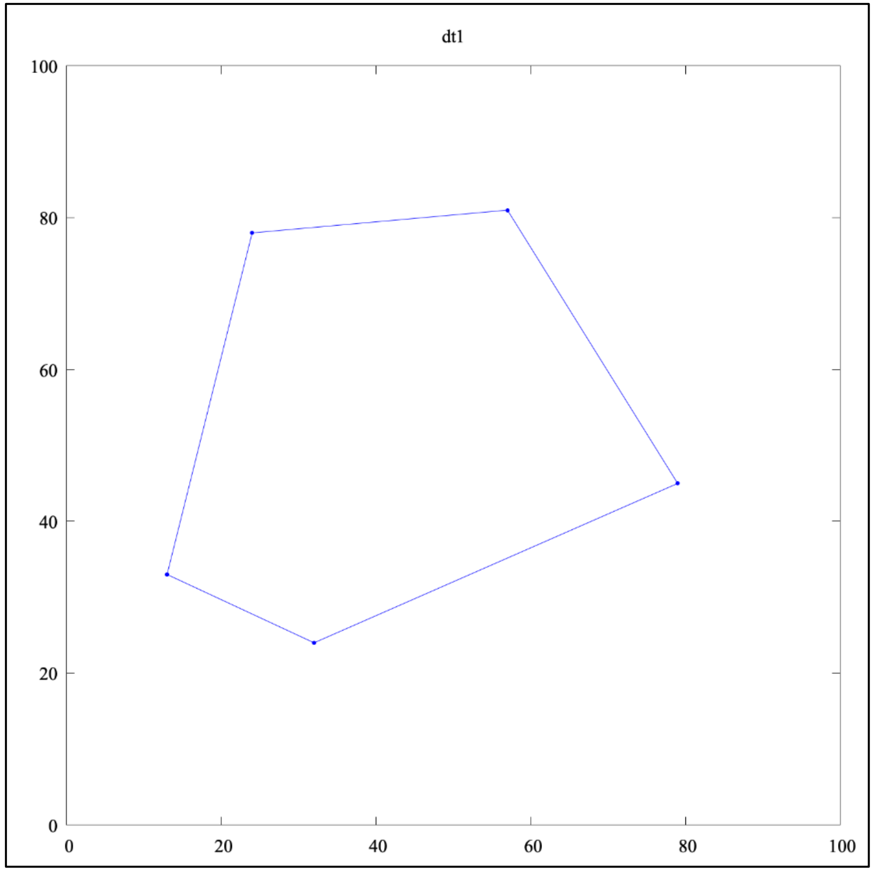


fig.png

Thank You ;-)