Teoria dos Grafos e Computabilidade

— Overview —

Silvio Jamil F. Guimarães

Graduate Program in Informatics – PPGINF
Laboratory of Image and Multimedia Data Science – IMScience
Pontifical Catholic University of Minas Gerais – PUC Minas

Feb 2025

Objetivos

Levar o aluno a compreender os principais conceitos relacionados ao projeto e análise de algoritmos; auxiliar o aluno no desenvolvimento das habilidades de desenvolver soluções computacionais para problemas por meio da modelagem usando estratégias de projeto de algoritmo; fornecer subsídios para que os alunos aperfeiçoem suas habilidades de desenvolvimento de sistemas, levando-os a reconhecer a importância da abstração e da redução de problemas

Algorithm Design

Objetivos

Levar o aluno a compreender os principais conceitos relacionados ao projeto e análise de algoritmos; auxiliar o aluno no desenvolvimento das habilidades de desenvolver soluções computacionais para problemas por meio da modelagem usando estratégias de projeto de algoritmo; fornecer subsídios para que os alunos aperfeiçoem suas habilidades de desenvolvimento de sistemas, levando-os a reconhecer a importância da abstração e da redução de problemas

Ementa

Notações para complexidade de algoritmos. Crescimento assintótico de funções e classes de complexidade. Limite inferior para classes de problemas. Análise de algoritmos recursivos. Técnicas de Projeto de Algoritmos: redução, divisão e conquista, programação dinâmica, método guloso, retrocesso e branch and bound. Tratabilidade de problemas. Teoria da Complexidade: classes de problemas P, NP e NP-Completo. Teorema de Cook.

Pedagogical Strategy

- Syncronous lectures
- Slides
- Forum
- Exercises
- Some tools for interactions (like Kahoot)

Algorithm Design

Pedagogical Strategy

- Syncronous lectures
- Slides
- Forum
- Exercises
- Some tools for interactions (like Kahoot)

Evaluation

- Exams
- Homeworks

Expected knowledge (a priori)

- Data structure
- Graph
- Discrete maths

Sorting

INSTANCE A list $L = x_1, x_2, \ldots, x_n$ of numbers.

SOLUTION A permutation (reordering) of the input sequence such $x_{i_1} \leq x_{i_2} \leq \cdots x_{i_n}$

Sorting

INSTANCE A list $L = x_1, x_2, \ldots, x_n$ of numbers.

SOLUTION A permutation (reordering) of the input sequence such $x_{i_1} \leq x_{i_2} \leq \cdots x_{i_n}$

It's expected to develop methods which are CORRECT and EFFICIENT.

Robot tour optimization

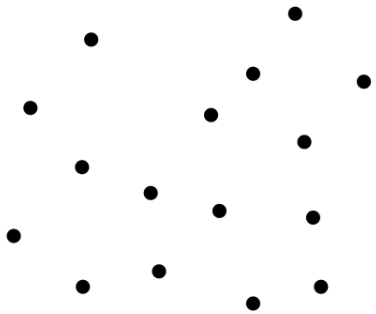
INSTANCE A robot arm equipped with a tool, say a soldering iron. To enable the robot arm to do a soldering job, we must construct an ordering of the contact points, so the robot visits (and solders) the points in order.

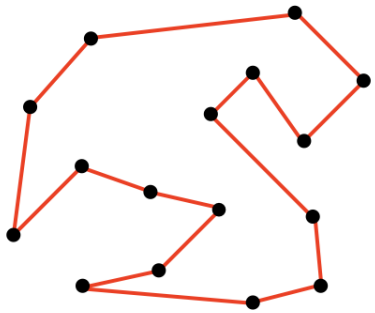
SOLUTION An order which minimizes the testing time (i.e. travel distance) it takes to assemble the circuit board.

Robot tour optimization

INSTANCE A robot arm equipped with a tool, say a soldering iron. To enable the robot arm to do a soldering job, we must construct an ordering of the contact points, so the robot visits (and solders) the points in order.

SOLUTION An order which minimizes the testing time (i.e. travel distance) it takes to assemble the circuit board.

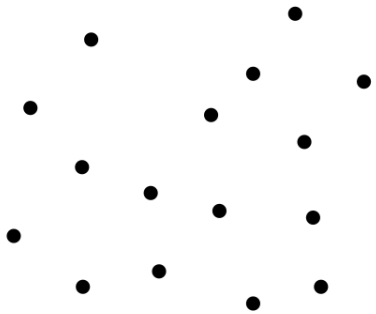


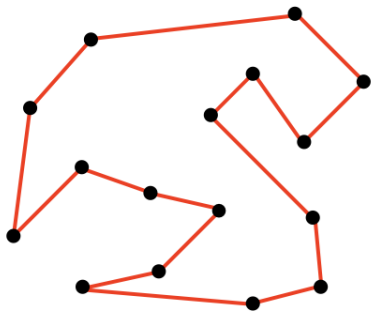


Robot tour optimization

INSTANCE A robot arm equipped with a tool, say a soldering iron. To enable the robot arm to do a soldering job, we must construct an ordering of the contact points, so the robot visits (and solders) the points in order.

SOLUTION An order which minimizes the testing time (i.e. travel distance) it takes to assemble the circuit board.





You are given the job to program the robot arm. **Develop algorithms** to find the most efficient tour!

Robot Tour Optimization

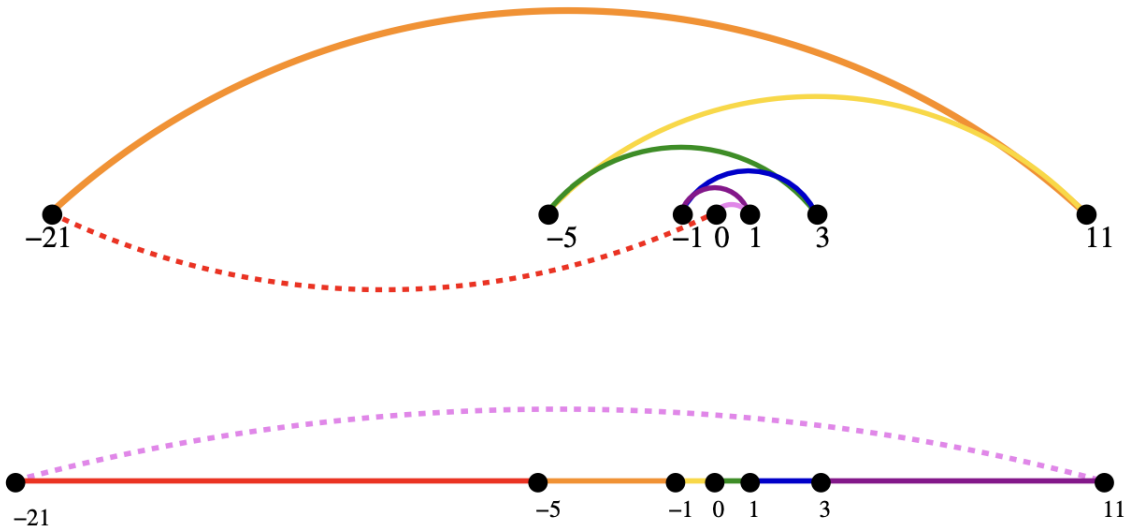
Nearest Neighbor Tour *A popular solution starts at some point p_0 and then walks to its nearest neighbor p_1 first, then repeats from p_1, etc. until done.*

Closest Pair Tour *Another idea is to repeatedly connect the closest pair of points whose connection will not cause a cycle or a three-way branch, until all points are in one tour.*

Robot Tour Optimization

Nearest Neighbor Tour *A popular solution starts at some point p_0 and then walks to its nearest neighbor p_1 first, then repeats from p_1, etc. until done.*

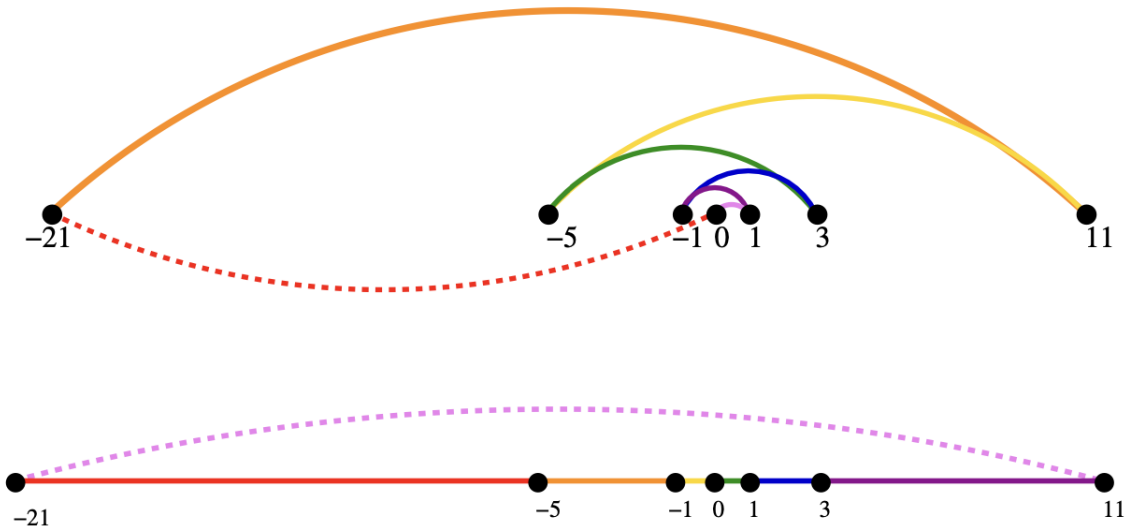
Closest Pair Tour *Another idea is to repeatedly connect the closest pair of points whose connection will not cause a cycle or a three-way branch, until all points are in one tour.*



Robot Tour Optimization

Nearest Neighbor Tour *A popular solution starts at some point p_0 and then walks to its nearest neighbor p_1 first, then repeats from p_1, etc. until done.*

Closest Pair Tour *Another idea is to repeatedly connect the closest pair of points whose connection will not cause a cycle or a three-way branch, until all points are in one tour.*



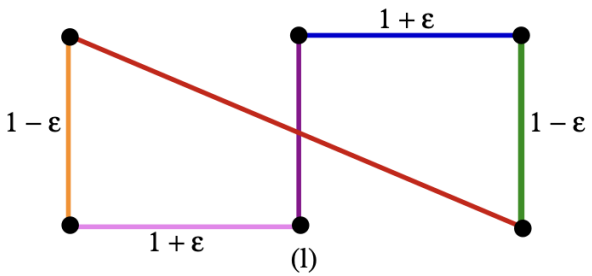
Nearest Neighbor Tour is

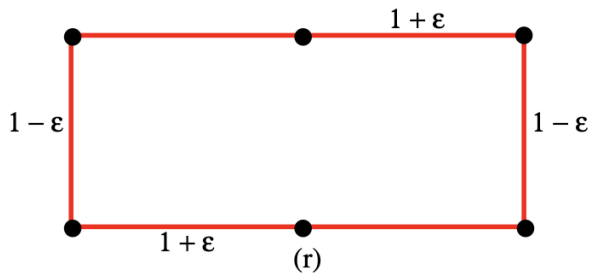
WRONG

Robot Tour Optimization

Nearest Neighbor Tour *A popular solution starts at some point p_0 and then walks to its nearest neighbor p_1 first, then repeats from p_1, etc. until done.*

Closest Pair Tour *Another idea is to repeatedly connect the closest pair of points whose connection will not cause a cycle or a three-way branch, until all points are in one tour.*

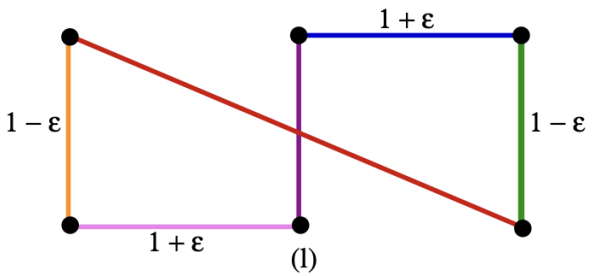


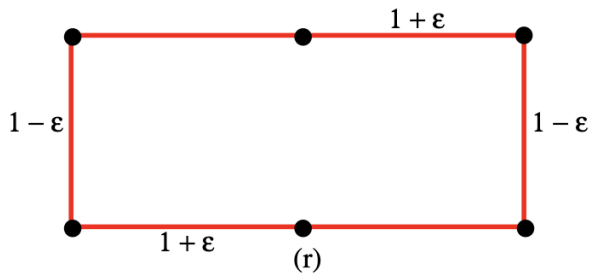


Robot Tour Optimization

Nearest Neighbor Tour *A popular solution starts at some point p_0 and then walks to its nearest neighbor p_1 first, then repeats from p_1, etc. until done.*

Closest Pair Tour *Another idea is to repeatedly connect the closest pair of points whose connection will not cause a cycle or a three-way branch, until all points are in one tour.*





Closest Pair Tour is

WRONG

Robot Tour Optimization

Nearest Neighbor Tour *A popular solution starts at some point p_0 and then walks to its nearest neighbor p_1 first, then repeats from p_1, etc. until done.*

Closest Pair Tour *Another idea is to repeatedly connect the closest pair of points whose connection will not cause a cycle or a three-way branch, until all points are in one tour.*

A Correct Algorithm: Exhaustive Search

Robot Tour Optimization

Nearest Neighbor Tour *A popular solution starts at some point p_0 and then walks to its nearest neighbor p_1 first, then repeats from p_1, etc. until done.*

Closest Pair Tour *Another idea is to repeatedly connect the closest pair of points whose connection will not cause a cycle or a three-way branch, until all points are in one tour.*

A Correct Algorithm: Exhaustive Search

*We could try **all possible orderings** of the points, then select the one which minimizes the total length. Since all possible orderings are considered, we are **guaranteed** to end up with the shortest possible tour.*

Searching elements

INSTANCE A list $L = x_1, x_2, \ldots, x_n$ of distinct integers and an element x.

SOLUTION The position i of the element x in the list L.

Searching elements

INSTANCE A list $L = x_1, x_2, \ldots, x_n$ of distinct integers and an element x.

SOLUTION The position i of the element x in the list L.

There are two different cases:

Searching elements

INSTANCE A list $L = x_1, x_2, \ldots, x_n$ of distinct integers and an element x.

SOLUTION The position i of the element x in the list L.

There are two different cases:

- The element x is in the list L – $x \in L$

Searching elements

INSTANCE A list $L = x_1, x_2, \ldots, x_n$ of distinct integers and an element x.

SOLUTION The position i of the element x in the list L.

There are two different cases:

- The element x is in the list L – $x \in L$

1	2	3	4	5	6	7	8
9	5	7	1	3	8	2	6

$x = 6$

Searching elements

INSTANCE A list $L = x_1, x_2, \ldots, x_n$ of distinct integers and an element x.

SOLUTION The position i of the element x in the list L.

There are two different cases:

- The element x is in the list L – $x \in L$

1	2	3	4	5	6	7	8
9	5	7	1	3	8	2	6

$x = 6$

Searching elements

INSTANCE A list $L = x_1, x_2, \ldots, x_n$ of distinct integers and an element x.

SOLUTION The position i of the element x in the list L.

There are two different cases:

- The element x is in the list L – $x \in L$
- The element x may not be in the list L – $x \in L$ or $x \notin L$

Searching elements

INSTANCE A list $L = x_1, x_2, \ldots, x_n$ of distinct integers and an element x.

SOLUTION The position i of the element x in the list L.

There are two different cases:

- The element x is in the list L – $x \in L$
- The element x may not be in the list L – $x \in L$ or $x \notin L$

1	2	3	4	5	6	7	8
9	5	7	1	3	8	2	6

$x = 4$

Searching elements

INSTANCE A list $L = x_1, x_2, \ldots, x_n$ of distinct integers and an element x.

SOLUTION The position i of the element x in the list L.

Searching elements

INSTANCE A list $L = x_1, x_2, \ldots, x_n$ of distinct integers and an element x.

SOLUTION The position i of the element x in the list L.

Design, at least, two different solutions for solving the searching element when the element x is in L.

Searching elements

INSTANCE A list $L = x_1, x_2, \ldots, x_n$ of distinct integers and an element x.

SOLUTION The position i of the element x in the list L.

Design, at least, two different solutions for solving the searching element when the element x is in L.

Solution 1

1. The list is organized as an array $[0, n-1]$
2. Go through L from 0 to $n-1$ checking the element at position i

Searching elements

INSTANCE A list $L = x_1, x_2, \ldots, x_n$ of distinct integers and an element x.

SOLUTION The position i of the element x in the list L.

Design, at least, two different solutions for solving the searching element when the element x is in L.

Solution 1

1. The list is organized as an array $[0, n-1]$
2. Go through L from 0 to $n-1$ checking the element at position i

Solution 2

1. The list is organized as an array $[0, n-1]$
2. Sort L
3. Go through L from 0 to $n-1$ checking the element at position i

Searching element: a solution

Searching element: a solution

Searching element: a solution

Searching elements

INSTANCE A list $L = x_1, x_2, \ldots, x_n$ of distinct integers and an element x.

SOLUTION The position i of the element x in the list L.

Searching elements

INSTANCE A list $L = x_1, x_2, \ldots, x_n$ of distinct integers and an element x.

SOLUTION The position i of the element x in the list L.

Number of instructions of two different solutions for solving the searching element.

Searching elements

INSTANCE A list $L = x_1, x_2, \ldots, x_n$ of distinct integers and an element x.

SOLUTION The position i of the element x in the list L.

Number of instructions of two different solutions for solving the searching element.

$x \in L$

$T(n) = 2 \times n + 1$

Searching elements

INSTANCE A list $L = x_1, x_2, \ldots, x_n$ of distinct integers and an element x.

SOLUTION The position i of the element x in the list L.

Number of instructions of two different solutions for solving the searching element.

$x \in L$	$x \in L$ or $x \notin L$
$T(n) = 2 \times n + 1$	$T(n) = 4 \times n + 4$

Searching elements

INSTANCE A list $L = x_1, x_2, \ldots, x_n$ of distinct integers and an element x.

SOLUTION The position i of the element x in the list L.

Number of instructions of two different solutions for solving the searching element.

$x \in L$	$x \in L$ or $x \notin L$
$T(n) = 2 \times n + 1$	$T(n) = 4 \times n + 4$

How to solve the general problem decreasing the number of instructions?

Searching elements

INSTANCE A list $L = x_1, x_2, \ldots, x_n$ of distinct integers and an element x.

SOLUTION The position i of the element x in the list L.

Number of instructions of two different solutions for solving the searching element.

$x \in L$	$x \in L$ or $x \notin L$
$T(n) = 2 \times n + 1$	$T(n) = 4 \times n + 4$

9	5	7	1	3	8	2	6

Searching elements

INSTANCE A list $L = x_1, x_2, \ldots, x_n$ of distinct integers and an element x.

SOLUTION The position i of the element x in the list L.

Number of instructions of two different solutions for solving the searching element.

$x \in L$	$x \in L$ or $x \notin L$
$T(n) = 2 \times n + 1$	$T(n) = 4 \times n + 4$

Searching element: a solution