Computer Science and Information Engineering
National Chi Nan University

Combinatorial Optimization

Dr. Justie Su-Tzu Juan

Lecture 1. Introduction

Slides for a Course Based on the Text
Combinatorial Optimization
by Cook, Cunningham, Pulleyblank and Schrijver

Introduction

- **Q: What is Combinatorial Optimization?**
- **A: "What is a best arrangement?"**
- https://zh.wikipedia.org/wiki/%E7%BB%84%E5%90%88%E4%BC%98%E5%8C%96

- **教學目標:**

  以GRAPH為工具，學習一般組合學上遇到求解最佳化問題時，研究與解決的方法。

- **評分方式:**

  作業40% + 平時成績20% + 期末報告30% + 自主學習報告10%
  (平時作業七題 + 期中(考)作業)

- TA: ?

Introduction

- **課程進度及綱要:**
  1. Introduction[1]
  2. Mathematical Preliminaries[124]
  3. Domination Problem in Tree[7]
  4. The Domination Problems on Interval Graphs[7]
  5. The Domatic Number Problem on Interval Graphs[7]
  6. NP-Completeness[17]
  7. Applications of L. P. Duality :
     The Weighted Domination Problem on Strongly Chordal Graphs[7]
  8. Applications of L. P. Duality : The Independent Set on Chordal Graphs[7]
  9. Domatic Number Problem[7]
  10. More Class of Graphs[7]
  11. Minimum Spanning Trees[1]
  12. Shortest Paths 1[1]
  13. Shortest Paths 2[1]
  14. Maximum Flow Problem[1]
  15. Minimum-Cost Flow Problem[1]

將根據上課情形調整進度與方向

Introduction

- **課程進度及綱要:**

第1週: Introduction
第2週: Mathematical Preliminaries
第3週: Domination Problem in Tree
第4週: The Domatic Number Problem on Interval Graphs
第5週: The Domination Problems on Interval Graphs
第6週: NP-Completeness
第7週: Applications of L.P. Duality : The Weighted Domination Problem on Strongly Chordal Graphs
第8週: 民族掃墓節放假
第9週: Applications of L. P. Duality : The Independent Set on Chordal Graphs

Introduction

- **課程進度及綱要:**

第10週: Domatic Number Problem
第11週: More Class of Graphs
第12週: Minimum Spanning Trees
第13週: Shortest Paths 1
第14週: Shortest Paths 2
第15週: 期末報告
第16週: 期末報告
第17週: 自主學習(請閱讀相關論文並繳交報告)
第18週: 自主學習(請閱讀相關論文並繳交報告)

Introduction

- 參考書籍:

1. William J. Cook, William H. Cunningham, William R. Pulleyblank, Alexander Schrijver, Combinatorial Optimization, Wiley-Interscience, 1997 (2011).
2. Eugene Lawler, Combinatorial Optimization: Networks and Matroids, Dover Publications, 2001
3. B. H. Korte, Jens Vygen, Bernhard Korte, Combinatorial Optimization: Theory and Algorithms (Algorithms and Combinatorics), Springer-Verlag, 2002.
4. Christos H. Papadimitriou, Kenneth Steiglitz, Combinatorial Optimization : Algorithms and complexity, Dover Publications, 1998
5. Jon Lee, D. G. Crighton, A First Course in Combinatorial Optimization, (華通代理), 2004.
6. Laurence A. Wolsey, George L. Nemhauser, Integer and Combinatorial Optimization, Wiley - Interscience, 1999.
7. 相關論文

Computer Science and Information Engineering
National Chi Nan University

Combinatorial Optimization

Dr. Justie Su-tzu Juan

Lecture 1. Introduction

§ 1.1 Two Problems

Slides for a Course Based on the Text
Combinatorial Optimization
by Cook, Cunningham, Pulleyblank and Schrijver

1.1 Two Problems –
The Traveling Salesman Problem

- **Story: An oil company, 47 drilling platforms, visit platforms periodically.**
  - **Q: How to routing s.t. total helicopters' flying time is minimized?**

- **Def: Euclidean Traveling Salesman Problem**
  - **Input: A set V of points in the Euclidean plane.**
  - **Output: Find a simple circuit passing through the points for which the sum of the lengths of the edges is minimized.**

- **Method 1: Try all possible solutions, then select the best.**

  → if $|V| = n$, then $\exists$ $(n-1)!/2$ possible solutions.

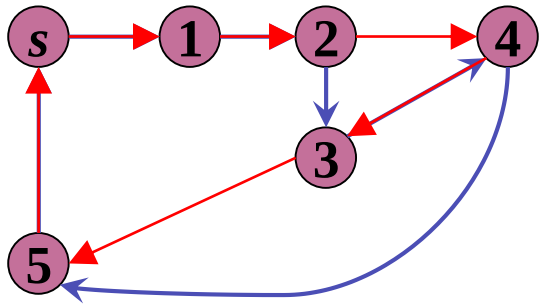
  → Use a computer, evaluating a single possibility in 10^{-9} sec, 23 platforms, need ≥ 178 centuries!!

1.1 Two Problems –
The Traveling Salesman Problem

- **Method 2: Nearest Neighbor Algorithm**

  Pick any starting point, go to the nearest point not yet visited. Continue from these to the nearest unvisited point. Until all points have been visited, then return to the starting point.

- **Ex: (1) Figure 1.1** **(2)**



- **Defect:**

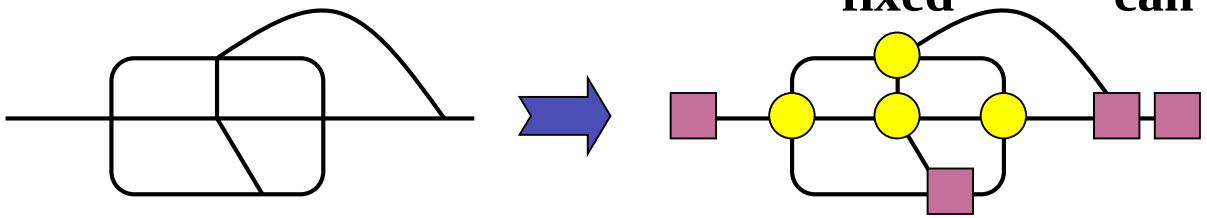
  1. **It is easy to omit a point, which must be visited later at great expense.**
  2. **Forced to make a long move to reach the nearest point where you can continue the tour.**

1.1 Two Problems –
The Matching Problem

- **Story: Design logic circuits, use a plotter to draw. Plotter: moving a pen back and forth; rolling paper forwards and backwards. Diff colors.**
  - **Q: Minimize the time required: "pen-down" / "pen-up"**

fixed **can be reduced**

- **Ex:**



- **Def: node (odd or even)**

- **Theorem:**
  1. **There will always be an even number of odd nodes.**
  2. **(Euler) The figure can be traced, returning to the starting node with no pen-up iff it is connected and ≠ odd nodes.**

1.1 Two Problems –
The Matching Problem

- **Solution: Add new lines between 2 odd nodes s.t. no odd node leave.**

- **Suppose: $t(p, q)$: the time required to draw a line from p to q and $\forall\ p, q, r, t(p, r) \leq t(p, q) + t(q, r)$. (triangle inequality)**

- **Def: Euclidean Matching Problem**
  - **Input: A set V of points in the Euclidean plane.**
  - **Output: Find a set of lines, such that each point is an end of exactly one line, and such that the sum of the lengths of the lines is minimized.**

- **Ex:**

1.1 Two Problems –
The Matching Problem

- **Note:**
  1. **If the original figure is not connected, some extra pen-up motion is needed.**
  2. **Euclidean traveling salesman problem is a special case of this problem (the case of not connected).**

1.1 Two Problems –
Some Similarities and Differences (of ETSP and EMP)

- **Similarities:**
  1. **selecting lines connecting points in the plane.**
  2. **the number of feasible solutions in far too large.**
  3. **most simple heuristics for the problems do not perform very well.**

- **Difference:**
  - **$\exists$ efficient algorithm (Edmonds) to find an optimal solution for the Euclidean matching problem.**
  - **No such algorithm known for the Euclidean traveling salesman problem, and most researchers believe $\not\exists$. (Chapter 9)**

- **Throughout the book: discuss between these 2 kinds.**
  - **Chapter 5, 6 $\rightarrow$ in-depth treatment of matching problem.**
  - **Chapter 7 $\rightarrow$ discussion why traveling salesman problem "bad"!**

Computer Science and Information Engineering
National Chi Nan University

Combinatorial Optimization

Dr. Justie Su-tzu Juan

Lecture 1. Introduction

§ 1.2 Measuring Running Times

Slides for a Course Based on the Text
Combinatorial Optimization
by Cook, Cunningham, Pulleyblank and Schrijver

1.2 Measuring Running Times

- **"efficient": "giving upper bounds on the number of steps"**

- **Before:**
  1. **It should not be taken as a hard and fast rule that "having a better bound" means better performance in practice.**
  2. **Only point out the advantages of one method over another.**
  3. **Led to discover many superior algorithms (both in theoretical sense and practice).**

- **Two models:**
  1. **Arithmetic model: +, ×, ÷, comparison → unit cost.**
  2. **Bit model: count the numbers of "bit operation". (ex: prime)**

1.2 Measuring Running Times

- **Note:**

  1. Ignore $3n$ in $5n^2+3n$
  2. $5n^2$ v.s. $17n^2$

  } describing $5n^2+3n$ as "order n^2"

- **Def:** If $f(n)$ and $g(n)$ are positive real-valued functions on the set of nonnegative integers. ($f, g: \mathbb{Z}^+ \cup \{0\} \rightarrow \mathbb{R}^+$), we say $f(n)$ is $\mathcal{O}(g(n))$ if $\exists$ constant $c > 0$ s.t. $f(n) \leq c \cdot g(n)$ $\forall$ large enough values of n. The notation $\mathcal{O}(g(n))$ is read "big oh of $g(n)$".

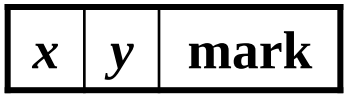
- **Ex:** 1. $5n^2+3n$ is $\mathcal{O}(n^2)$

  2. $35 \cdot 2^n+n^3$ is $\mathcal{O}(2^n)$

1.2 Measuring Running Times

- **Ex: 3. The Nearest Neighbor Algorithm for the E.T.S.P.:**
  - **In arithmetic model:**

    initial "0", visited "1"

    Input: $2n$;

    Use three fields n-element array

    Initialization: $3n$

    general pass: $n+6(n-1)$ (3: add., 2: multi., 1:comparison)

    $\Rightarrow 3n+(n-1)(n+5(n-1)) = \mathcal{O}(n^2)$
  - **In bit model:**

    Let $M = \max\{1+\lceil \log(|X|+1) \rceil, 1+\lceil \log(|Y|+1) \rceil\}$ ("log" = "$\log_2$")

    Input: $2nM$

    Initialization: $2nM+n$

    general pass: $\mathcal{O}(nM^2)$

    $\Rightarrow \mathcal{O}(n^2M^2)$

1.2 Measuring Running Times

- **Remark:**

  1. Our main goal will be to present algorithms that having running-time bounds of $\mathcal{O}(n^k)$ for small values of k, whenever possible, say "polynomial-time algorithm".

  2. Typically, we will work with the arithmetic model, but a polynomial-time bound in the arithmetic model $\Rightarrow$ a polynomial-time bound in the bit model when the sizes of the numbers appearing in the algorithm do not grow too fast. ($t \rightarrow \mathcal{O}(t^k)$ for some fixed k)

  3. In this text, "polynomial time" : it runs in the bit model.

  4. $\exists$ problems (ex: linear programming problem) satisfy polynomial-time algorithm in the bit model are known, but no algorithm is known that is polynomial-time in the arithmetic model.

  - **Chapter 9 $\rightarrow$ discuss the issue of computational complexity further.**