《算法分析与设计》
上机实验选题及要求

1 实验目的

- 掌握利用算法进行问题求解的实验流程
- 强化对课堂所学算法原理的理解
- 提升利用算法原理进行编码实践的能力
- 了解算法研究的前沿动态，拓展算法改进的思路

2 实验内容及要求

实验采取自由分组的方式完成，每组至多3人，编程语言推荐但不限于$c/c++$。要求每组从下列主题中选择1 项作为实验内容，完成实验设计、过程实现、结果分析、实验总结及实验报告撰写。

2.1 算法改进类

针对货箱装船问题、0/1 背包问题、子集和数问题、多段图最段路径问题、所有点对最短路径问题、矩阵乘法链问题、最大非交叉子网问题、最长公共子序列问题、带罚款额的作业调度问题、TSP 问题的、最近点对问题、数逆序对问题 其中之一:

- 指出现有求解算法的局限，提出改进方案或新的算法解决方案
- 融合多种算法策略，针对问题特点，提出改进方案
- 放松约束条件的限制，提出改进方案
- 加强约束条件的限制，提出改进方案

要求：改进方案有针对性、思路清晰；需要对比实验结果。

2.2 算法验证类

- 利用贪心、分治、动态规划、回溯及分支限界中某一算法，选择附件 1 中至少两个问题的至少三个数据集 进行实验验证

要求：分析数据集分布特点对算法性能的影响，归纳算法解决不同问题的优缺点。

2.3 问题求解类

- 针对附件 1 中某一问题的至少 3 个数据集，利用利用贪心、分治、动态规划、回溯及分支限界中至少三种算法进行求解。

要求：分析不同算法解决同一问题的复杂度，归纳不同求解算法的优缺点。

2.4 前沿算法实现类

- 从附件 2 中选择1 篇文献，编码实现其中的核心算法, 并进行实验验证

要求：必须以 PPT 的方式展示算法的基本原理、核心算法的实现细节。

3 实验报告要求

- 题目：60 字以内，如：“动态规划算法问题求解性能比较”
- 摘要：300 字左右，如：“本实验针对... 问题，利用... 算法，对.... 数据集进行性能分析。实验结果表明...”
- 正文：5-8 页，包括实验目的、实验设计流程、代码实现、实验结果及复杂性分析
- 总结：200 字左右，包括实验过程总结和得出的主要结论。

附件 1：问题求解数据集

0/1 包问题

- Florida State University：8 个数据集。Link
- Unicauca University：31 个数据集。Link

子集和数问题

- Florida State University：7 个数据集。Link

矩阵乘法链问题

- github：9 个数据集。link

所有点对最短短路径问题

- gitlab：利用 Ggen.cpp 随机生成：Link

最大集团问题

- DIMACS-benchmark, 37 个数据集，Link

TSP 问题

- Florida State University: 6 个数据集, Link
- TSP LIB: 总计 143 个数据集, Link

最近点对问题

- Git hub: 4 个数据集 Link

数逆序对问题

- Git hub: 3 个数据集 Link

附件 2：文献阅读列表

1. Dynamic programming approaches for the traveling salesman problem with drone [1]

   Abstract: A promising new delivery model involves the use of a delivery truck that collaborates with a drone to make deliveries. Effectively combining a truck and a drone gives rise to a new planning problem that is known as the traveling salesman problem with drone (TSP-D). This paper presents exact solution approaches for the TSP-D based on dynamic programming and provides an experimental comparison of these approaches. Our numerical experiments show that our approach can solve larger problems than the mathematical programming approaches that have been presented in the literature thus far. Moreover, we show that restrictions on the number of locations the truck can visit while the drone is away can help significantly reduce the solution times while having relatively little impact on the overall solution quality.

2. Fast Dynamic Programming on Graph Decompositions [2]

   Abstract: In this paper, we consider three types of graph decompositions, namely tree decompositions, branch decompositions, and clique decompositions. We improve the running time of dynamic programming algorithms on these graph decompositions for a large number of problems as a function of the treewidth, branchwidth, or cliquewidth, respectively. Such algorithms have many practical and theoretical applications.

   The main techniques used in this paper are a generalization of fast subset convolution, as introduced by Bj¨orklund et al., now applied in the setting of graph decompositions and augmented such that multiple states and multiple ranks can be used. In the case of branch decompositions, we combine this approach with fast matrix multiplication, as suggested by Dorn. Recently, Lokshtanov et al. have shown that some of the algorithms obtained in this paper have running times in which the base in the exponents is optimal, unless the Strong Exponential-Time Hypothesis fails.

3. Robust Shortest Path Planning and Semicontractive Dynamic Programming [3]

Abstract: In this article, we consider shortest path problems in a directed graph where the transitions between nodes are subject to uncertainty. We use a minimax formulation, where the objective is to guarantee that a special destination state is reached with a minimum cost path under the worst possible instance of the uncertainty. Problems of this type arise, among others, in planning and pursuit-evasion contexts, and in model predictive control. Our analysis makes use of the recently developed theory of abstract equation, existence of optimal paths, and the validity of various algorithms patterned after the classical methods of value and policy iteration, as well as a Dijkstra-like algorithm for problems with nonnegative arc lengths.

4. Subset Sum Made Simple [4]

   Abstract: SubsetSum is a classical optimization problem taught to undergraduates as an example of an NP-hard problem, which is amenable to dynamic programming, yielding polynomial running time if the input numbers are relatively small. Formally, given a set S of n positive integers and a target integer t, the SubsetSum problem is to decide if there is a subset of S that sums up to t. Dynamic programming yields an algorithm with running time $O(nt)$. Recently, the authors improved the running time to $\tilde{O}(\sqrt{n}t)$. it was further improved to $\tilde{O}(n+t)$ by a somewhat involved randomized algorithm by Bringmann, where $\tilde{O}$ hides polylogarithmic factors. Here, we present a new and significantly simpler algorithm with running time $\tilde{O}(\sqrt{n}t)$. While not the fastest, we believe the new algorithm and analysis are simple enough to be presented in an algorithms class, as a striking example of a divide-and-conquer algorithm that uses FFT to a problem that seems (at first) unrelated. In particular, the algorithm and its analysis can be described in full detail in two pages.

5. A New Exact Algorithm for Traveling Salesman Problem with Time Complexity Interval $(O(n^4), O(n^3 * 2^n))$ [5]

   Abstract: Traveling salesman problem is a NP-hard problem. Until now, researchers have not found a polynomial time algorithm for traveling salesman problem. Among the existing algorithms, dynamic programming algorithm can solve the problem in time$O(n^2 \cdot 2^n)$ where n is the number of nodes in the graph. The branch-and-cut algorithm has been applied to solve the problem with a large number of nodes. However, branch-and-cut algorithm also has an exponential worst-case running time. In this paper, a new exact algorithm for traveling salesman problem is proposed. The algorithm can be used to solve an arbitrary instance of traveling salesman problem in real life and the time complexity interval of the algorithm is $(O(n^4), O(n^3 * 2^n))$. It means that for some instances, the algorithm can find the optimal solution in polynomial time although the algorithm also has an exponential worst-case running time. In other words, the algorithm tells us that not all the instances of traveling salesman problem need exponential time to compute the optimal solution. The algorithm of this paper can not only assist us to solve traveling salesman problem better, but also can assist us to deepen the comprehension of the relationship between NP-complete and P. Therefore, it is considerable in the further research on traveling salesman problem and NP-hard problem.

6. Efficient algorithms for the longest common subsequence in k-length substrings [6]

Abstract: Finding the longest common subsequence in k-length substrings (LCSk) is a recently proposed problem motivated by computational biology. This is a generalization of the well-known LCS problem in which matching symbols from two sequences A and B are replaced with matching non-overlapping substrings of length k from A and B. We propose several algorithms for LCSk, being non-trivial incarnations of the major concepts known from LCS research (dynamic programming, sparse dynamic programming, tabulation). Our algorithms make use of a linear-time and linear-space preprocessing finding the occurrences of all the substrings of length k from one sequence in the other sequence.

References

[1] Paul Bouman, Niels Agatz, and Marie Schmidt. Dynamic programming approaches for the traveling salesman problem with drone. Networks, 72(4):528–542, December 2018.

[2] Johan M. M. van Rooij, Hans L. Bodlaender, Erik Jan van Leeuwen, Peter Rossmanith, and Martin Vatshelle. Fast Dynamic Programming on Graph Decompositions. arXiv:1806.01667 [cs], June 2018. arXiv: 1806.01667.

[3] Dimitri P. Bertsekas. Robust shortest path planning and semicontractive dynamic programming: Robust Shortest Path Planning. Naval Research Logistics (NRL), 66(1):15–37, February 2019.

[4] Konstantinos Koiliaris and Chao Xu. Subset Sum Made Simple. arXiv:1807.08248 [cs], July 2018. arXiv: 1807.08248.

[5] Yunpeng Li. A new exact algorithm for traveling salesman problem with time complexity interval $(O(n^4), O(n^3 * 2^n))$. arXiv, page 22, 2014.

[6] Sebastian Deorowicz and Szymon Grabowski. Efficient algorithms for the longest common subsequence in k-length substrings. arXiv:1311.4552 [cs], November 2013. arXiv: 1311.4552.