南京航空航天大学

# 带约束的单源最短路径的一些快速算法

# faster algorithms on constrained single-source shortest path


| | |
|---|---|
| 学生姓名 | 李想 |
| 学　　号 | 161610323 |
| 学　　院 | 计算机科学与技术学院 |
| 专　　业 | 计算机软件培优班 |
| 班　　级 | 1616001 |
| 指导教师 | 陈松灿教授 |


二〇一八年八月

# 目 录

# 第一章　引言

在 Dijkstra 算法及其扩展专题中，我们小组的研讨方向是的对边权大小种类较少的一类优化，基数堆优化，多重基数堆优化三种优化方法。在对比实验中，我们主要就如何优化算法，优化的目标以及不同优化算法之间的关系进行了讨论与总结。

# 第二章 算法简介

## 2.1 解决有少量不同正边权的单源最短路问题的快速算法

该算法是由赵子渊同学讲述的，他所参考的论文是《A faster algorithm for the single source shortest path problem with few distinct positive lengths》[1]。这种算法是基于有 $n$ 个顶点，$m$ 条边。，$K$ 个不同的正边权的图，这种算法，当 $nK \leq 2m$ 时，时间复杂度为 $O(m)$, 否则为 $O(m\log(\frac{nK}{m}))$。

## 2.2 解决正整数边权的单源最短路问题的快速算法

该算法是由张天凡同学讲述的，他所参考的论文是《Faster Algorithms for the Shortest Path Problem》[2] 这种算法将基数堆与 Dijkstra 相结合，在满足在 $n$ 个顶点，$m$ 条边，且其中所有权都小于上界 $C$ 的非负整数权图上，一阶基数堆可以优化到 $O(m + n\log C)$。

## 2.3 一种基于桶，堆，表的单调优先队列及其对 Dijsktra 算法的优化应用

该算法是由我讲述的，我所参考的论文是《Buckets, Heaps, Lists, and Monotone Priority Queues》[3] 这种算法在普通基数堆的基础上，将桶与优先队列相结合成新的数据结构 multilevel bucket 作为新的桶来优化堆，使得在满足在 $n$ 个顶点，$m$ 条边，且其中所有权都小于上界 $C$ 的非负整数权图上，时间复杂度可降至 $O(m + n(\log C)^{\frac{1}{3}+\epsilon})$，其中 $\epsilon$ 为 Thorup's heaps[4] 的一个常数。

# 第三章 实验仿真

## 3.1 数据设置

我们使用 GRIDGEN 生成器[5] 生成，其两种图的类型及参数如表3.1所示。

表 3.1 Graph Types

| Graph Type | Graph Family | Range of values | Other values |
|---|---|---|---|
| long grid | Modifying $C$ | $C = 1\,to\,1,000,000$ | $x = 8192$ |
| | Modifying $x$ | $x = 512\,to\,32,768$ | $C = 16$ |
| | | | $C = 10,000$ |
| | | | $C = 100,000,000$ |
| | Modifying $C$ and $x$ | $x = 512\,to\,32,678$ | $C = x$ |
| wide grid | Modifying $C$ | $C = 1\,to\,1,000,000$ | $y = 8192$ |
| | Modifying $y$ | $y = 512\,to\,32,768$ | $C = 16$ |
| | | | $C = 10,000$ |
| | | | $C = 100,000,000$ |
| | Modifying $C$ and $y$ | $y = 512\,to\,32,678$ | $C = y$ |

## 3.2 测试环境

系统：Windows 10 Pro 1803 17134.112

处理器：Intel(R) Core(TM) i5-7300HQ CPU @ 2.50GHz 2.50GHz

运行内存：8.00 GB

编译环境：TDM64-GCC 4.9.2

## 3.3 实验结果

其中 h1 为算法 2，h3 为算法 3 的 heap-on-top priority queue，算法一由于其适用场景与 2,3 不同，所以未进行比较。

### 3.3.1 Long grids

表 3.2 运行时间 ($C = 16$)

| n | 8193 | 16385 | 32769 | 65537 | 131073 | 262145 | 524289 |
|---|---|---|---|---|---|---|---|
| h1 | 0.03s | 0.05s | 0.10s | 0.20s | 0.41s | 0.81s | 1.62s |
| h3 | 0.03s | 0.06s | 0.11s | 0.22s | 0.45s | 0.89s | 1.78s |

表 3.3 运行时间 ($C = 100,000,000$)

| n | 8193 | 16385 | 32769 | 65537 | 131073 | 262145 | 524289 |
|---|---|---|---|---|---|---|---|
| h1 | 0.04s | 0.08s | 0.15s | 0.26s | 0.51s | 1.03s | 2.05s |
| h3 | 0.03s | 0.06s | 0.12s | 0.23s | 0.46s | 0.91s | 1.80s |

表 3.4 运行时间 ($b = 131,073$)

| C | 100 | 1000 | 10000 | 100000 | 1000000 | 9999994 | 99999937 |
|---|---|---|---|---|---|---|---|
| h1 | 0.44s | 0.48s | 0.47s | 0.48s | 0.49s | 0.52s | 0.50s |
| h3 | 0.47s | 0.48s | 0.48s | 0.46s | 0.47s | 0.46s | 0.46s |

### 3.3.2 Wide grids

表 3.5 运行时间 ($C = 100,000,000$)

| n | 8193 | 16385 | 32769 | 65537 | 131073 | 262145 | 524289 |
|---|---|---|---|---|---|---|---|
| h1 | 0.07s | 0.12s | 0.24s | 0.42s | 0.85s | 1.85s | 3.77s |
| h3 | 0.03s | 0.07s | 0.14s | 0.31s | 0.68s | 1.45s | 3.05s |

表 3.6 运行时间 ($n = 131,073$)

| C | 100 | 1000 | 10000 | 100000 | 1000000 | 9999994 | 99999937 |
|---|---|---|---|---|---|---|---|
| h1 | 0.56s | 0.58s | 0.63s | 0.74s | 0.83s | 0.85s | 0.85s |
| h3 | 0.61s | 0.63s | 0.64s | 0.66s | 0.68s | 0.67s | 0.68s |

# 第四章 结论和总结

## 4.1 对 Dijkstra 优化方向的分析

很容易分析，限制 Dijkstra 算法时间复杂度瓶颈在于从 dist[] 中选择未确定的顶点的值中最小的值, 我们可以通过普通的二叉堆，将插入，删除操作复杂度从 $O(N)$ 降低至 $O(\log N)$，将查询复杂度从 $O(\log N)$ 降低至 $O(1)$，而二叉堆并非在线查询最值的最优数据结构，所以我们想到可以去运用不同的数据结构来优化甚至代替二叉堆从而降低 Dijkstra 算法的复杂度。

## 4.2 论文提出算法的操作复杂度比较

表 4.1 时间复杂度

| Algorithm | Insert | DeleteMin | DecreaseKey |
|---|---|---|---|
| Radix Heap | $O(1)$ | $O(\log C)$ | $O(1)$ |
| HOT queue | $O(\log^{\frac{1}{3}} C)$ | $O(\log^{\frac{1}{3}+\epsilon} C)$ | $O(1)$ |

## 4.3 适用情况分析

通过理论分析易得，算法 2 适用于稀疏图（m 较小），算法 3 适用于稠密图，实验也证明了这点，在小数据或是十分稀疏的图中算法 3 的表现甚至不如算法 2，但在大多数数据中算法 3 的表现要优于算法 2。

## 4.4 对于算法优化的思考

学习了这三篇论文的优化方法，我了解到要对一个问题的现有算法进行优化就要先分析算法复杂度找到复杂度瓶颈与问题条件的关系，在这个关系上进行优化，而优化的方法包括但不限于适用高效数据结构，减少冗余计算等等。

# 致谢

感谢陈松灿教授在算法研讨课上的讲授，这让我更深的理解了算法优化的一些知识与理论，给予了我极大的启发。此外还要感谢赵子渊同学和张天凡同学在我阅读论文时提供的无私帮助，在他们的帮助下我才得以顺利完成此篇文章。

# 参考文献

[1] ORLIN J B, MADDURI K, SUBRAMANI K, et al. A faster algorithm for the single source shortest path problem with few distinct positive lengths[J]. Journal of Discrete Algorithms, 2010, 8(2): 189–198.

[2] AHUJA R K, MEHLHORN K, ORLIN J, et al. Faster algorithms for the shortest path problem[J]. Journal of the Acm, 1990, 37(2): 213–223.

[3] CHERKASSKY B V, GOLDBERG A V, SILVERSTEIN C. Buckets, heaps, lists, and monotone priority queues [C]. [S.l.: s.n.], 1997: 83–92.

[4] THORUP M. On ram priority queues[J]. Siam Journal on Computing, 1996, 30(1): 86–109.

[5] CHERKASSKY B V, GOLDBERG A V, RADZIK T. Shortest paths algorithms: Theory and experimental evaluation[C]. [S.l.: s.n.], 1994: 516–525.