《算法分析与设计》

实 验 报 告

学 号

姓 名

年 级 2020

专 业 软件工程

院 系 计算机与人工智能学院

二O二二年三月

# 实验3.3穷举法解决任务分配问题

1. 实验目的

|  |
| --- |
| 1. 掌握算法时间效率的分析方法。理解穷举法典型算法的求解过程。 2. 学习穷举法的时间复杂度分析方法，并通过实验验证算法的执行效 率。 3. 学会如何利用穷举法求解具体问题，了解穷举法的局限性。 |

1. 实验任务

|  |
| --- |
| 1. 有n（n>=1）个任务需要分配给n个人执行，每个任务只能分配给 一个人，每个人只能执行一个任务，第i个人执行第j个任务的成本是（1<=i, j<=n）。求出总成本最小的一种分配方案。 |

1. 实验环境

|  |
| --- |
| * 1. 硬件环境  1. 计算机：拯救者R7000P 2020H 2. CPU: AMD Ryzen 7 4800H with Radeon Graphics 2.90 GHz 3. RAM：16GB    1. 软件环境 4. 操作系统：Windows11家庭中文版 5. 开发工具：Visual Studio Code |

1. 实验步骤及结果

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| * 1. 实验预习      1. 根据样例输入，采用穷举法求出一种总成本最小的分配方案，写出求解过程。   样例输入：  9 2 7 8  6 4 3 7  5 8 1 8  7 6 9 4  穷举法求解过程如表1所示。     |  |  |  |  | | --- | --- | --- | --- | | 分配方案  （依次为1、2、3、4号人所花费成本） | 总成本 | 分配方案  （依次为1、2、3、4号人所花费成本） | 总成本 | | 9 4 8 9 | 30 | 9 4 1 4 | 18 | | 9 3 8 4 | 24 | 9 3 8 6 | 26 | | 9 7 8 9 | 33 | 9 7 1 6 | 23 | | 2 6 1 4 | 13 | 2 6 8 9 | 25 | | 2 3 5 4 | 14 | 2 3 8 7 | 20 | | 2 7 5 9 | 23 | 2 7 1 7 | 17 | | 7 6 8 4 | 25 | 7 6 8 6 | 27 | | 7 4 5 4 | 20 | 7 4 8 7 | 26 | | 7 7 5 6 | 25 | 7 7 8 7 | 29 | | 8 6 8 9 | 31 | 8 6 1 6 | 21 | | 8 4 5 9 | 26 | 8 4 1 7 | 20 | | 8 3 5 6 | 22 | 8 3 8 7 | 26 |   表1 穷举法求解过程  依表可知，最小成本为13。   * + 1. 程序代码   *//解空间为子集树*  #include <cstring>  #include <iostream>  using namespace std;  #define INF 0x3f3f3f3f  #define MAXN 10  int n = 4; *//任务数量*  int c[MAXN][MAXN]; *//第i个人执行第j个任务的成本*  int x[MAXN]; *//临时解*  int cost = 0; *//临时解的成本*  int bestx[MAXN]; *//最优解*  int mincost = INF; *//最优解的成本*  bool worker[MAXN]; *// worker[j]表示任务j是否已分配人员*  void dfs(int *i*)  {      if (*i* > n)      {          if (cost < mincost)          {              mincost = cost;              for (int k = 1; k <= n; k++)                  bestx[k] = x[k];          }      }      else      {          for (int j = 1; j <= n; j++)          { *//为人员i试探任务j，从1到n*              if (!worker[j])              { *//任务j还未分配*                  worker[j] = true;                  cost += c[*i*][j];                  x[*i*] = j;                  dfs(*i* + 1);  *//回溯*                  worker[j] = false;                  cost -= c[*i*][j];                  x[*i*] = 0;              }          }      }  }  int main()  {      memset(worker, false, sizeof(worker)); *//任务还未分配*      for (int i = 1; i <= n; i++)          for (int j = 1; j <= n; j++)              cin >> c[i][j];      dfs(1);      cout << "最小总成本为：" << mincost << endl;      system("pause");      return 0;  }  根据测试样例运行结果如图1。    图1 对应的程序执行结果  4.2上机实验  4.2.1算法测试  新测试数据  1 5 3  4 2 6  8 7 9  4.2.2测试结果及其分析  对应新的测试数据，其程序执行过程中间变量的值如下截图所示；    图2 程序执行第一个子树（第一个人）  为第一个人分配第一个任务，总成本  Cost=1 （1）    图3 程序执行第一个子树（第二个人）  为第二个人分配第二个任务，总成本  Cost=3 （2）    图4 程序执行第一个子树（第三个人）  为第二个人分配第二个任务，总成本  Cost=12 （3）    图5 程序执行第二个子树  第二个子树执行完毕，此时总成本为  Cost=14 （4）  此时最小成本为  mincost=12 （5）    图6 程序执行第三个子树  第三个子树执行完毕，此时总成本为  Cost=18 （6）  此时最小成本为  mincost=12 （7）  程序执行结果如图6所示，    图7 程序执行结果  程序对应的树图如图7所示，    图8 树图及中间变量  由树图分析可知，最小成本花费为12，与程序运行结果一致，验证正确。  5.实验总结  通过本次实验，理解了穷举法典型算法的求解过程，学习了穷举法的时间复杂度分析方法，并通过实验验证算法的执行效率，学会如何利用穷举法求解具体问题，了解了穷举法的局限性。 |