实验四报告

程序语言：Python

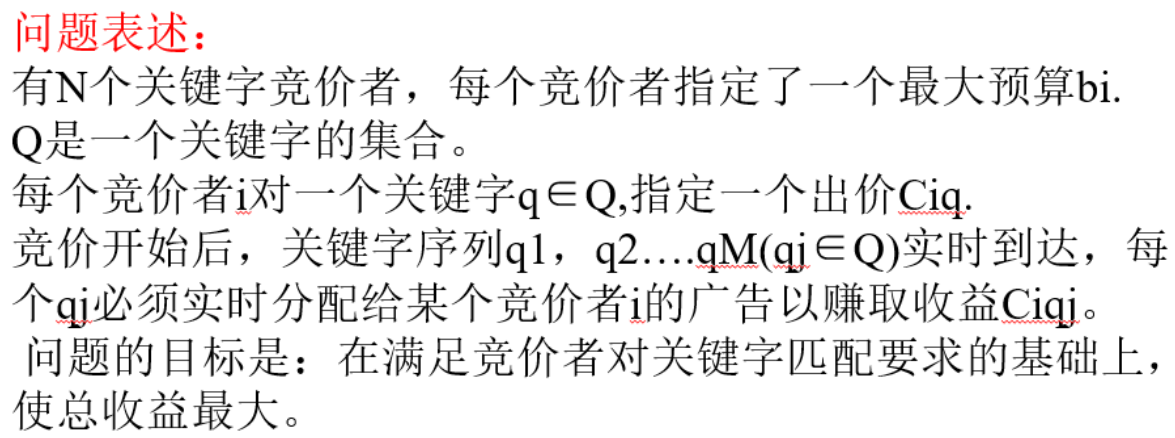
姓名：郑子睿

学号：22920212204317

专业：计算机科学与技术

内容：在线广告

一、问题描述



二、算法思想

本实验是对一篇论文中的方法的复现

[论文下载地址]

算法描述如下：

```
最新到来的关键字，应该匹配给V值最大的那个广告
其中：
    V = c(i)* CTR * ph(T(i))
    其中：
    c(i) = 该广告为该关键字i的出价
    CTR：该广告历史点击量 / 该广告历史显示总次数
    ph(x) = 1 - exp(x - 1)
    T(i) = mi / bi
    其中：
    mi：目前为止该广告的竞价者当日已经实际用掉多少钱
    bi：该广告的竞价者当日总预算
```

三、描述算法

函数传入两个参数：

1. `bidders`

   储存了所有竞价者的：

- 姓名
- 每个关键字的报价
- 当日总预算
- 当日已使用的钱数
- CTR

2. `adwords`

   随机生成的关键字序列

```
def calculateGrossIncome(bidders, adwords):
    grossIncome = 0 # 总收益
    for adword in adwords:
        pqueue = queue.PriorityQueue() # 定义一个优先队列，便于查找最大的V
        for name, info in bidders.items():
            # 用公式算出每个竞价者的V，存入优先队列中
            V = info[adword] * info['CTR'] * (1 - math.exp(info['当日已使用金额'] / info['当日总预算'] - 1))
            pqueue.put((-V, (name, info[adword])))
        winner = pqueue.get()
        print(f'关键字 "{adword}" 分配给了{winner[1][0]}，收益{winner[1][1]}元')
        grossIncome += winner[1][1]
        bidders[winner[1][0]]['当日已使用金额'] -= winner[1][1]
    return grossIncome
```

四、验证算法

具体代码

```
# 导入模块
import openpyxl as xlsx
import queue
import random
import math

# 读取excel中的数据
def load():
    book = xlsx.load_workbook('data.xlsx')
    bidderSheet = book['竞价者信息表']
    # 创建bidderDict，用于储存每个竞价者的信息和出价
    bidders = {}
    for bidder in bidderSheet[2:bidderSheet.max_row]:
        name = ''
        for col in bidder:
            header = bidderSheet.cell(1, col.col_idx).value
            if header == '竞价者':
                bidders[col.value] = {}
                name = col.value
            else:
                bidders[name][header] = col.value
    bidSheet = book['关键词出价表']
    adwords = set([])
    for bid in bidSheet[2:bidSheet.max_row]:
        name = ''
        for col in bid:
```

```
            header = bidSheet.cell(1, col.col_idx).value
            if header == '竞价者':
                name = col.value
            else:
                adwords.add(header)
                bidders[name][header] = col.value
    return bidders, list(adwords)

# 计算总收益
def calculateGrossIncome(bidders, adwords):
    grossIncome = 0
    for adword in adwords:
        pqueue = queue.PriorityQueue()
        for name, info in bidders.items():
            V = info[adword] * info['CTR'] * (1 - math.exp(info['当日已使用金额'] / info['当日总预算'] - 1))
            pqueue.put((-V, (name, info[adword])))
        winner = pqueue.get()
        print(f'关键字 "{adword}" 分配给了{winner[1][0]}，收益{winner[1][1]}元')
        grossIncome += winner[1][1]
        bidders[winner[1][0]]['当日已使用金额'] -= winner[1][1]
    return grossIncome

# 主函数
if __name__ == '__main__':
    data, adwords = load()
    random.shuffle(adwords)
    grossIncome = calculateGrossIncome(data, adwords)
    print(f'最大总收益{grossIncome}元')
```

结果展示

竞价者信息表

	A	B	C	D	E
1	竞价者	当日已使用金额	当日总预算	CTR	V
2	A	10.00	100.00	50.00%	
3	B	30.00	100.00	50.00%	
4	C	10.00	60.00	20.00%	
5	D	66.00	100.00	50.00%	
6	E	10.00	100.00	20.00%	
7	F	0.10	1,000.00	50.00%	
8	G	2.00	100.00	50.00%	
9	H	80.00	300.00	40.00%	

关键词出价表

	A	B	C	D	E
1	竞价者	手机	照相机	笔记本电脑	平板
2	A	1.00	1.50	1.00	1.00
3	B	1.00	0.10	1.00	3.00
4	C	1.05	2.00	1.00	0.99
5	D	1.00	1.00	1.00	0.10
6	E	1.50	1.00	1.00	1.00
7	F	0.10	1.00	0.10	1.00
8	G	2.00	0.50	1.00	0.10
9	H	1.00	0.10	1.00	2.00

代码运行结果

```
关键字 "笔记本电脑" 分配给了G，收益1元
关键字 "手机" 分配给了G，收益2元
关键字 "平板" 分配给了B，收益3元
关键字 "照相机" 分配给了A，收益1.5元
最大总收益7.5元
```

五、结论

本文参考了现有论文的做法，做了代码的复现。

改进措施：可以结合离线算法