# 摘要

# Abstract

# 目录

[摘要 1](#_Toc420414921)

[Abstract 1](#_Toc420414922)

[目录 1](#_Toc420414923)

[1 绪论 2](#_Toc420414924)

[2 0-1背包问题的概述 2](#_Toc420414925)

[2.1 0-1背包问题的数学描述 2](#_Toc420414926)

[2.2 0-1背包问题的传统求解方法 2](#_Toc420414927)

[2.2.1 递归法 2](#_Toc420414928)

[2.2.2 动态规划法 3](#_Toc420414929)

[2.2.3 分支定界法 3](#_Toc420414930)

[3 蚁群算法的基本原理 3](#_Toc420414931)

[3.1 蚁群算法的简介 3](#_Toc420414932)

[3.2 基本蚁群算法的数学模型 3](#_Toc420414933)

[4 蚁群算法求解0-1背包问题 4](#_Toc420414934)

[4.1 KPACA蚁群算法的数学描述 4](#_Toc420414935)

[4.2 KPACA蚁群算法的编程实现 5](#_Toc420414936)

[4.3 仿真实验 5](#_Toc420414937)

[5 结论 5](#_Toc420414938)

[6 致谢 5](#_Toc420414939)

[7 参考文献 5](#_Toc420414940)

[附录A 部分核心代码 5](#_Toc420414941)

# 1 绪论

蚁群算法是一种基于模拟蚂蚁觅食行为的仿生学优化算法。该算法常常被用来解决路径优化等相关问题（如TSP问题、0-1背包问题）。本文将讨论蚁群算法在0-1背包问题中的应用，并给出相关的算法与结论。

# 2 0-1背包问题的概述

## 2.1 0-1背包问题的数学描述

背包问题(Knapsack problem)是由Merkel和Hellman于1978年提出的一种组合优化问题。该问题表述为：给定一组物品，每种物品的重量与价格已知。在不超过给定的总重量的前提下，如何选取物品的组合，使他们的总价值达到最大。

这里给出其数学描述：

给定种物品，、分别为第件物品的重量与价值。现有一个总容量为的背包。要求一个维0-1向量，使得满足的条件下，使达到最大。

即求解规划问题



## 2.2 0-1背包问题的传统求解方法

0-1背包问题已被证明为是一种NP完全问题。即无法找到一个有效的算法，能在多项式的时间复杂度内求出该问题的精确解。该问题有很多常用的优化算法，这里列举以下几种：

### 2.2.1 递归法

递归法是解决0-1背包问题的最常用的方法之一。该算法的描述如下：

设表示将前件物品放入容量为的背包中所能得到的最大价值。则有

|  |  |  |
| --- | --- | --- |
|  |  | (2-1) |

对上式的含义可以做出这样的解释：考虑第件物品放与不放的情形，若第件物品不被放入背包之中，则显然有。若第件物品放入背包中，则需腾出的空间来存放前件物品，故此时有，考虑到上述两种情况，便可得到式2-1。

### 2.2.2 动态规划法

动态规划是求解决策过程最优化的一种常用数学方法。该方法将多阶段的求解过程转化为一系列单阶段的问题，并通过各阶段之间的关系，依次求解。

### 2.2.3 分支定界法

分支定界法是一种求解整数规划问题的常用算法。该算法的主要步骤如下：

1. 放宽或取消原规划问题的限制条件，如所求变量必须取整数等条件。若此时求得的解为可行解，则该解为问题的最优解。否则该解为此问题的最优解上界。

2. 将被放宽条件的问题分解成若干个子问题，子问题解的集合的并集为原问题的可行解。对每个子问题进行求解。这些子问题的解中最优者若是原问题的可行解，则该解为原问题的最优解。否则，该解为原问题的一个新的最优解上界。另外，子问题的解中若存在原问题的可行解，则将这些解中的最优解作为原问题最优解的下界。

3. 若最优解的目标函数值已经低于最优解下界的子问题，可以肯定其中已不包含原问题的最优解，可以不作考虑。其余的子问题保留。

4. 在剩余的子问题中，选取具有最优的最优解的子问题。重复步骤1、2。指导求出原问题的最优解。

对于0-1背包问题，还有许多其他的算法如贪婪算法、图论法等等。本文将采用一种寻找优化路径的几率型算法——蚁群算法来求解0-1背包问题。

# 3 蚁群算法的基本原理

## 3.1 蚁群算法的简介

蚁群算法(Ant Colony Optimization, ACO)是一种模拟进化算法，由Marco Dorigo于1992年在他的博士论文中提出。

自然界中蚂蚁在寻找食物时，总能聚集到从蚁穴到食物之间最短的路径之上。这是因为蚂蚁在寻找路径的同时释放了一种信息素。蚂蚁经历的路径越长，被释放的信息素浓度将越低。如此，较短的路径上将聚集浓度较高的信息素。而其它蚂蚁在选择路径时，信息素浓度较高的路径具有较大的概率被选择。在这样一种正反馈的机制下，蚂蚁将趋于集中到最短的路径上。这就是蚁群算法的核心思想。

蚁群算法广泛应用于各类优化问题的求解。对于0-1背包问题，蚁群算法也具有较好的求解性能。本文将做出基于蚁群算法求解0-1背包问题的相关讨论。

## 3.2 基本蚁群算法的数学模型

Marco Dorigo在他的博士论文中提出了三种基本蚁群算法的模型，分别为And-Cycle模型、Ant-Quantity模型以及Ant-Density模型。这三者的区别在于模型中蚂蚁释放信息素的时机不同。

这里给出Ant-Cycle蚁群算法的数学模型（以求解TSP问题为例）：

设蚂蚁需搜寻的路径节点数为，蚂蚁数量为，时刻路径上的信息素含量为，路径的长度为，第只蚂蚁已经过的节点集合为，第只蚂蚁下一步可以选择的节点集合为，第只蚂蚁在时刻从节点转移到节点的概率为，则有

|  |  |  |
| --- | --- | --- |
|  |  | (3-1) |

上式中，、分别为信息启发因子、期望启发因子。信息启发因子表明了信息素的浓度对蚂蚁选择路径时的重要性，期望启发因子代表能见度（即路径长度）对蚂蚁选择路径的重要性。

蚂蚁在所有城市都被遍历一次之后，将更新所经过路径上的信息素，记为第只蚂蚁遍历所有城市后，在路径上所释放的信息素量

|  |  |  |
| --- | --- | --- |
|  |  | (3-2) |

所有蚂蚁遍历城市一次后，信息素总量将更新为

|  |  |  |
| --- | --- | --- |
|  |  | (3-3) |

公式2-3中，被称为信息素挥发系数，其值越大，信息素的挥发效果将越明显。

通过多次迭代上述搜索过程，由于信息素逐渐积累，蚁群将逐渐收敛至该TSP问题的某个局部最优解上。

# 4 蚁群算法求解0-1背包问题

## 4.1 KPACA蚁群算法的数学描述

前文所述的基本蚁群算法是基于求解TSP问题的。对于0-1背包问题，需要对该模型做适当的修正。这里采用KPACA算法进行求解。

下面给出KPACA算法的数学模型：

将0-1背包问题的可行解以0、1的形式存在于数组中，其中表示蚂蚁对物品的操作状态，该值取0表明未装入背包，取1表明已装入背包。

其中第件物品被蚂蚁装入背包的概率为

|  |  |  |
| --- | --- | --- |
|  |  | (4-1) |

上式中为时刻物品所携带的信息素量，为物品的启发函数，此处启发函数为物品的单位重量价值，即。为蚂蚁尚未搜索过的物品集合。

在所有蚂蚁搜索完成之后，对物品携带的信息素进行更新操作：

|  |  |  |
| --- | --- | --- |
|  |  | (4-2) |

上式中为信息素挥发系数。若物品被该代蚁群中总价值最大的蚂蚁选中，则，否则。其中为信息素强度，为该代蚁群中价值最大蚂蚁背包中物品的总价值。

## 4.2 KPACA蚁群算法的编程实现

上述的KPACA蚁群算法的实现步骤如下：

1. 初始化参数，令蚂蚁数量为m，最大迭代次数为MAX\_LOOP\_COUNT，当前代数i = 0，各物品的信息素浓度pheromone[i] = 1，进入步骤2。

2. 令蚂蚁索引号k = 0，重置该蚂蚁的禁忌表，进入步骤3。

3. 蚂蚁按照公式4-1的概率以轮盘赌规则选取一件物品，若背包尚有空间存放该物品，则将该物品放入背包，并更新禁忌表，重复执行步骤3。否则进入步骤4。

4. 蚂蚁索引号k = k + 1，若k < m，进入步骤3，否则进入步骤5。

5. 按照公式4-2更新物品携带的信息素。

6. 迭代次数i = i + 1,若i < MAX\_LOOP\_COUNT，则进入步骤2，否则进入步骤7。

7. 算法结束，得到价值最大的物品组合。

## 4.3 仿真实验

本文通过随机生成各种不同规模的0-1背包问题，

# 5 结论

# 6 致谢

# 7 参考文献

# 附录A 部分核心代码