



南開大學

Nankai University

计算机学院

算法导论实验报告

# 基于动态规划的加密货币市场策略优化

姓名：窦楷然

学号：2210652

专业：计算机科学与技术

2024 年 6 月 1 日

## 摘 要

本文提出了一种基于动态规划的算法，用于优化比特币的投资策略。通过分析比特币的历史价格数据，并结合每次交易的成本比例和最大交易次数等输入参数，算法计算出最优的买入和卖出时机，以实现最大收益。该算法还评估了投资策略的夏普比率和最大回撤，以帮助投资者进行风险管理和决策。实验结果表明，尽管策略在一定条件下可以实现高收益，但伴随着高风险，夏普比率为负且最大回撤较大。本文还讨论了算法的局限性，包括市场的非线性特征、风险管理指标的不足以及时效性问题。

由于空间和简洁性问题，文中没有附代码，所有源代码均上传到 github 上，链接为：

[https://github.com/Darkeyssss/NKU\\_Alg2024](https://github.com/Darkeyssss/NKU_Alg2024)，也可以[点击这里](#)。

**关键词：**高斯消去法，SIMD 优化，SSE 和 AVX 指令集，NEON 指令集，x86 和 ARM 平台

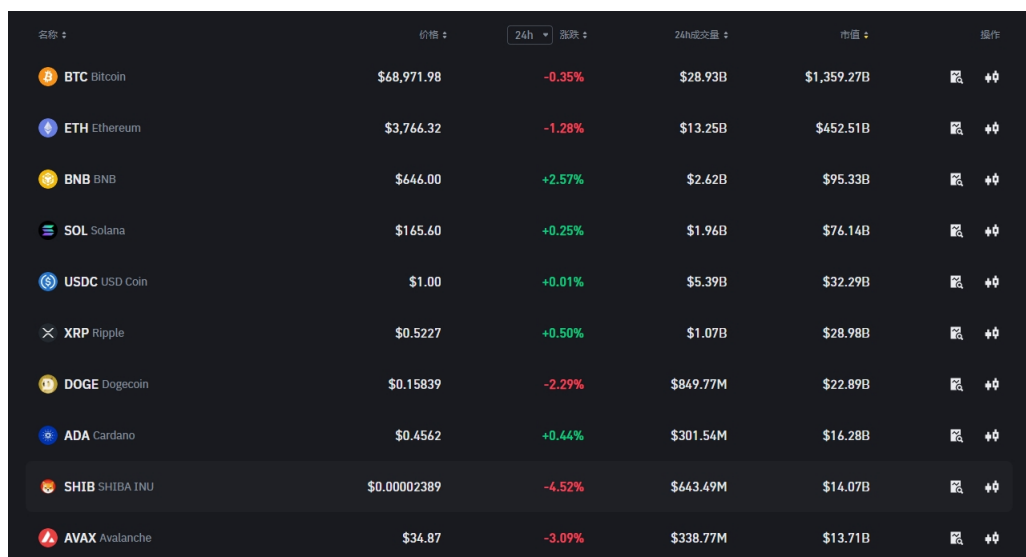
## 目 录

<b>1 算法背景</b>	<b>2</b>
<b>2 算法介绍</b>	<b>2</b>
2.1 问题描述	2
2.2 算法核心	3
2.3 算法设计	3
<b>3 测试数据来源</b>	<b>5</b>
<b>4 实验结果</b>	<b>6</b>
<b>5 算法局限性</b>	<b>6</b>

## 1 算法背景

随着区块链技术的迅速发展，加密货币市场迎来了爆炸性的增长，而其中比特币作为第一种加密货币，自 2009 年问世以来，已经成为这一市场的标志性资产，其价格在短短几年内经历了巨大的波动，从几美分的初始价值到数万美元的高峰，比特币吸引了全球各地投资者的广泛关注。然而，与传统金融市场相比，加密货币市场具有更高的波动性和风险，这种高波动性不仅带来了巨大的获利机会，同时也增加了投资的不确定性和潜在损失。因此，投资者在加密货币市场中不仅需要关注市场行情，更需要有效的投资策略和风险管理工具来实现收益的最大化和风险的最小化。

在此背景下，如何在高波动性和高风险的市场环境中，寻找最优的交易策略，成为投资者和研究人员共同关注的问题。在算法课上我学习过动态规划算法，而动态规划作为一种重要的优化方法，能够通过递归的方式，分解复杂问题并找到最优解，我在查阅资料后发现其在金融领域尤其是**投资策略优化**中有着广泛的应用前景。特别是对于比特币这样的高风险资产，传统的投资策略往往无法应对其剧烈的价格波动，动态规划方法则可以通过精细的计算，优化每一步的交易决策，从而在全局上实现收益的最大化。



名称	价格	24h 涨跌	24h成交量	市值	操作
BTC Bitcoin	\$68,971.98	-0.35%	\$28.93B	\$1,359.27B	👁️ ⬆️
ETH Ethereum	\$3,766.32	-1.28%	\$13.25B	\$452.51B	👁️ ⬆️
BNB BNB	\$646.00	+2.57%	\$2.62B	\$95.33B	👁️ ⬆️
SOL Solana	\$165.60	+0.25%	\$1.96B	\$76.14B	👁️ ⬆️
USDC USD Coin	\$1.00	+0.01%	\$5.39B	\$32.29B	👁️ ⬆️
XRP Ripple	\$0.5227	+0.50%	\$1.07B	\$28.98B	👁️ ⬆️
DOGE Dogecoin	\$0.15839	-2.29%	\$849.77M	\$22.89B	👁️ ⬆️
ADA Cardano	\$0.4562	+0.44%	\$301.54M	\$16.28B	👁️ ⬆️
SHIB SHIBA INU	\$0.0002389	-4.52%	\$643.49M	\$14.07B	👁️ ⬆️
AVAX Avalanche	\$34.87	-3.09%	\$338.77M	\$13.71B	👁️ ⬆️

图 1.1: 目前为止市值前 10 的加密货币

所以，本文提出了一种基于动态规划的方法，用于优化比特币的投资策略。[\[1\]](#) 通过输入每次交易的成本比例、最大交易次数和无风险利率，结合比特币的历史价格数据，算法能够计算出最大收益、夏普比率和最大回撤，并输出结果。通过这样的计算，投资者可以在制定投资决策时，有效地评估不同策略的风险和收益，从而在波动较大的加密货币市场中，找到适合自己的投资路径。我认为这不仅有助于提升投资收益，还能通过夏普比率和最大回撤的指标，帮助投资者进行风险评估和控制，从而实现更稳健的投资。

## 2 算法介绍

### 2.1 问题描述

该算法解决的问题是如何在考虑交易成本、最大交易次数和无风险利率的情况下，通过历史价格数据优化比特币的交易策略，以实现最大收益和风险控制，这些功能帮助投资者在波动较大的加密货

币市场中，更好地管理风险并优化投资回报，从而实现更稳健和高效的投资。

具体问题是：给定了一个我从网上下载的比特币交易数据，具体数据来源和形式我接下来会介绍，然后用户输入**每次交易的成本比例和最大交易次数**，然后算法可以求解出基于这段时间的比特币能够获得**的最大收益**，**买入和卖出日期索引以及夏普比率和最大回撤**，这些指标可以更好的帮助投资者进行投资。

**样例输入**

0.001 10

**样例输出**

```
最大收益: 37158.1  
买入日期索引: 15, 卖出日期索引: 359  
夏普比率: -0.880754  
最大回撤: 57.0214%  
  
C:\VSProjects\Project1\x64\Debug\Project1.exe (进程 60268)已退出, 代码为 0.  
按任意键关闭此窗口. . . |
```

## 2.2 算法核心

动态规划我在课上学过，是一种用于解决具有重叠子问题和最优子结构性问题的算法技术。它的**核心内容**是通过将问题分解为更小的子问题，并存储每个子问题的解，通过构建一个表格或数组来保存中间结果，避免了重复计算，从而提高了效率。算法的一般步骤包括定义状态和状态转移方程，初始化边界条件，以及通过填表法或递归来计算最终解。

在本算法中，我使用动态规划用于优化比特币的交易策略，具体体现在 `optimize_trading` 函数中。其思想是通过递归计算每个时间点在给定的最大交易次数和交易成本限制下的最优收益。首先，我定义一个二维动态规划表 `dp`，其中 `dp[i][t]` 表示在第  $i$  天进行最多  $t$  次交易的最大收益。然后，另一个表 `decision` 记录每个时间点的买卖决策。

为了计算每个时间点的最优收益，需要考虑两种情况：不进行交易，保持前一天的收益；或者进行一次交易，计算可能的最大收益。在迭代过程中，算法不断更新每个时间点的最优买卖决策，确保在每一步都能找到最大收益。具体来说，对于每一天，算法计算在当前价格下进行交易的潜在收益，并将其与前一天的收益进行比较，选择其中较大的值更新到 `dp` 表中。

通过这种方式，动态规划算法利用之前计算的子问题解，逐步构建出全局最优解，避免了重复计算，提高了计算效率。最终，通过回溯决策矩阵，算法能够确定最优的买入和卖出日期，并返回最大收益。这种方法有效地结合了交易成本和交易次数的限制，提供了一种在高波动性的比特币市场中优化投资策略的有效手段。

## 2.3 算法设计

### 动态规划部分

动态规划主要是通过状态转移方程实现递归。首先我初始化了两个二维数组 `dp` 和 `decision`，其中 `dp[i][t]` 表示在前  $i$  天内进行了  $t$  次交易的最大收益，`decision[i][t]` 用于记录是否在第  $i$  天进行交易决策。

然后使用状态转移方程递推，设 `dp[i][t]` 为在前  $i$  天内进行了  $t$  次交易的最大收益，`price[i]` 为第  $i$  天的价格，`cost` 为每次交易的成本，`min_price` 为在前  $i$  天内最小的有效价格，状态转移方程如下：

$$\text{min\_price} = \min(\text{min\_price}, \text{price}[i] - \text{dp}[i-1][t-1] - \text{cost}) \quad (1)$$

$$\text{potential\_profit} = \text{price}[i] - \text{min\_price} - \text{cost} \quad (2)$$

$$\text{dp}[i][t] = \max(\text{dp}[i-1][t], \text{potential\_profit}) \quad (3)$$

最终进行回溯决策，这样动态规划就结束了。下面是这个函数的伪代码。

---

**Algorithm 1** 优化交易策略的动态规划算法
 

---

**Input:** 价格序列  $\text{price}$ , 交易成本比例  $\text{cost}$ , 最大交易次数  $\text{max}$

**Output:** 最优交易的买入日期索引  $\text{buy\_day}$ , 卖出日期索引  $\text{sell\_day}$ , 最大收益  $\text{profit}$

```

1: 初始化  $n \leftarrow \text{price.size}()$ 
2: 创建二维数组  $\text{dp}[n][\text{max} + 1]$  并初始化为 0
3: 创建二维数组  $\text{decision}[n][\text{max} + 1]$  并初始化为 0
4: for  $t \leftarrow 1$  to  $\text{max}$  do
5:    $\text{min\_price} \leftarrow \text{price}[0]$ 
6:   for  $i \leftarrow 1$  to  $n - 1$  do
7:      $\text{min\_price} \leftarrow \min(\text{min\_price}, \text{price}[i] - \text{dp}[i-1][t-1] - \text{cost})$ 
8:      $\text{potential\_profit} \leftarrow \text{price}[i] - \text{min\_price} - \text{cost}$ 
9:     if  $\text{potential\_profit} > \text{dp}[i-1][t]$  then
10:       $\text{dp}[i][t] \leftarrow \text{potential\_profit}$ 
11:       $\text{decision}[i][t] \leftarrow 1$ 
12:     else
13:       $\text{dp}[i][t] \leftarrow \text{dp}[i-1][t]$ 
14:     end if
15:   end for
16: end for
17:  $\text{best\_profit} \leftarrow \text{dp}[n-1][\text{max}]$ 
18: 初始化  $\text{time}$  为一个空向量
19:  $t \leftarrow \text{max}$ 
20: for  $i \leftarrow n - 1$  downto 1 do
21:   if  $\text{decision}[i][t] == 1$  then
22:     将  $i$  加入  $\text{time}$ 
23:      $t \leftarrow t - 1$ 
24:   end if
25: end for
26: 反转  $\text{time}$ 
    return  $\text{time.front}(), \text{time.back}(), \text{best\_profit}$ 

```

---

### 夏普比率和最大回撤

夏普比率和最大回撤都是衡量投资组合风险调整后收益的重要指标，这一部分主要是为了评估投资的风险评估和对投资进行控制，属于经济学指标，我不是很理解，于是网上查询了其公式并编写在代码中。公式如下：

$$\text{Sharpe Ratio} = \frac{E[R_p] - R_f}{\sigma_p} \quad (4)$$

其中：

- $E[R_p]$  是投资组合的预期收益率。
- $R_f$  是无风险利率。
- $\sigma_p$  是投资组合收益率的标准差。

$$\text{Max Drawdown} = \max_{i \in [1, T]} \left( \frac{P_{\text{peak}} - P_i}{P_{\text{peak}}} \right) \quad (5)$$

其中：

- $P_{\text{peak}}$  是在时间段内达到的最高点的价格。
- $P_i$  是在时间段内某一天的价格。

### 3 测试数据来源

我的数据来源是 CoinMarketCap，这是一个专门提供加密货币市场资讯的网站，提供最新的加密货币价格、市值排名、交易量、市场趋势等相关数据。用户可以在 CoinMarketCap 上查询各种加密货币的实时价格走势，并了解项目背后的基本信息、团队信息以及市场情况。其下载的 csv 数据包括：timeOpen;timeClose;timeHigh;timeLow;name;open;high;low;close;volume;marketCap;timestamp

数据分别代表

timeOpen：交易开始的时间戳。

timeClose：交易结束的时间戳。

timeHigh：当天交易中的最高时间戳。

timeLow：当天交易中的最低时间戳。

name：比特币资产的名称。

open：开盘价。

high：最高价。

low：最低价。

close：收盘价。

volume：交易量。

marketCap：当天的市值。

timestamp：时间戳。

所有数据都是真实且最新的。

```
最大收益: 37158.1
买入日期索引: 15, 卖出日期索引: 359
夏普比率: -0.880754
最大回撤: 57.0214%

C:\VSProjects\Project1\x64\Debug\Project1.exe (进程 60268)已退出, 代码为 0。
按任意键关闭此窗口。 . . |
```

在导入数据的时候，因为这些数据下载下来都是 csv 的格式，而且都停留在第一行，所以我使用了 chatGPT 对数据进行了预处理，将其从 csv 格式变成 txt 格式。

预处理的代码我也放在 github 中了，但因为不是核心内容就不予以展示。

```
最大收益: 37158.1  
买入日期索引: 15, 卖出日期索引: 359  
夏普比率: -0.880754  
最大回撤: 57.0214%  
  
C:\VSPProjects\Project1\x64\Debug\Project1.exe (进程 60268)已退出, 代码为 0。  
按任意键关闭此窗口. . .
```

图 3.2: 部分数据

## 4 实验结果

对于之前的图??，可以看到最大收益是 37158.1，建议的买入日期索引是 15，卖出日期索引是 359，夏普比率是-0.880754，最大回撤是 57.0214%。

尽管策略在整个期间内实现了 37158.1 的收益，但夏普比率为负且最大回撤超过 57%，这表明该策略的风险非常高。所以高收益可能伴随着高波动性和高风险。而负的夏普比率和 57% 的最大回撤表明，在调整了风险后，策略的表现不如无风险利率，也就是说投资者在使用这个策略时，承担了额外的风险却没有获得相应的回报。得出结论这就是**高收入高风险的时候**。

紧接着我对于两个参数进行了调整，想看看不同策略的投资收益和风险如何。用控制变量法，我先调整了最大交易次数，为 100 次，大大增加，然后结果如下。

```
最大收益: 78149.3  
买入日期索引: 15, 卖出日期索引: 363  
夏普比率: -0.880754  
最大回撤: 57.0214%
```

虽然夏普比率为负且最大回撤依然很高，但是总收益变多了很多，这就说明只要交易次数增多总收益也会增多。

而每次交易的成本比例我也调整为 1，代表全都进行交易，结果如下：

```
最大收益: 37158.1  
买入日期索引: 15, 卖出日期索引: 359  
夏普比率: -0.990576  
最大回撤: 57.0214%
```

可以看到，总收益没有任何变化，但是夏普比率更低了，接近-1，所以可以得出这样的做法很激进，而且对于比特币交易而言得不偿失。

## 5 算法局限性

对于金融市场不甚了解的我只能总结出一个很粗糙的模型，因为在其中我假设市场是静态和线性的，忽略了市场中复杂的交互作用和非线性特征。我没有采用机器学习的非线性算法，而是使用了动态规划这样的算法，对于瞬息万变的市場考虑的不够周全，真正的市场受到政策、经济环境等多种因素的影响，而且历史规律可能不再适用于当前市场。

第二点，我认为我的**风险管理指标不足**。我这个现有的策略可能在面对极端市场条件时表现不佳，导致巨额亏损，比如，在市场崩盘或大幅波动时，策略可能无法及时调整，从而面临重大损失。而好巧

不巧加密货币市场的波动就是特别大，所以 dp 算法的止损机制和风险控制措施做的可能不够好。也许后期可以引入动态止损机制，根据市场波动性自动调整止损点，及时止损以避免重大损失，或使用仓位管理策略，控制每次交易的仓位大小，避免单笔交易风险过大。

第三点，也是最致命的一点，这个算法的时效性差。由于 CoinMarketCap 的 api 调用要收取相应费用，我选择了直接下载数据，然后再进行数据预处理和算法，这样只能对于已有的过去算法进行拟合和规划，无法做到机器学习这类算法的预测，而且以秒为单位的加密货币市场更是瞬息万变，我这种方法势必会很慢。

## 参考文献

- [1] Elliot L Amidon and Grath S Akin. Dynamic programming to determine optimum levels of growing stock. *Forest Science*, 14(3):287–291, 1968.