# 基于 OpenStack 云平台的蛋白质折叠模拟计算方法

徐胜超[1],周继鹏[2],吕峻闽[1]
(1. 广州华商学院 人工智能学院,广东 广州 511300;
2. 暨南大学 信息科学技术学院,广东 广州 511300)

**摘 要**:在蛋白质科学研究领域,蛋白质折叠模拟计算的速度和准确性一直是研究的热点和难点。为了有效提升蛋白质折叠模拟计算的效率,该文提出了一种基于 OpenStack 云平台的蛋白质折叠模拟计算方法。该方法首先利用 HP 格点模型对蛋白质折叠过程进行简化,将复杂的氨基酸链抽象为黑球和白球的组合形式,从而降低了计算的复杂性。在计算过程中,采用了一种改进的 PERM 算法来计算每个球的权重,该算法能够更准确地评估折叠过程中的能量体系。此外,借助 OpenStack 云平台的强大计算能力,将氨基酸链的折叠计算任务分配到多个计算节点上并行处理,从而大大提高了计算速度。每个计算节点都会返回其临时结果,最终这些结果会被汇总并输出为最终的折叠结果。实验表明,该方法计算的能量与最低能量相符,计算时间明显优于 PERM 算法,其折叠结果与已有的最佳折叠情况一致,且计算时间更短,提高了蛋白质折叠模拟计算效率。

**关键词**:云计算;蛋白质折叠;模拟计算;格点模型;OpenStack 集群
**中图分类号**:TP393.4 **文献标识码**:A **文章编号**:1673-629X(2025)07-0041-07
doi:10.20165/j.cnki.ISSN1673-629X.2025.0040

# Protein Folding Simulation Computation Method Based on OpenStack Cloud Platform

XU Sheng-chao[1], ZHOU Ji-peng[2], LYU Jun-min[1]
(1. School of Artificial Intelligent, Guangzhou Huashang College, Guangzhou 511300, China;
2. School of Information Science and Technology, Jinan University, Guangzhou 511300, China)

**Abstract**: In the field of protein science, the speed and accuracy of protein folding simulation have always been the focus and difficulty. In order to effectively improve the efficiency of protein folding simulation calculation, we propose a protein folding simulation calculation method based on OpenStack cloud platform. The HP lattice model is used to simplify the protein folding process, and the complex amino acid chain is abstracted into a combination of black and white spheres, thus reducing the computational complexity. In the calculation process, an improved PERM algorithm is used to calculate the weight of each ball, which can evaluate the energy system in the folding process more accurately. In addition, with the powerful computing power of OpenStack cloud platform, the folding calculation task of amino acid chain is allocated to multiple computing nodes for parallel processing, which greatly improves the computing speed. Each compute node returns its temporary results, which are eventually summarized and output as the final fold result. Experiments show that the energy calculated by the proposed method is consistent with the minimum energy, and the calculation time is obviously better than that of the PERM algorithm. The folding result is consistent with the existing optimal folding situation, and the calculation time is shorter, which improves the simulation efficiency of protein folding.

**Key words**: cloud computing; protein folding; simulation computing; lattice model; OpenStack cluster

## 0 引 言

蛋白质折叠模拟计算是通过数学方法对蛋白质结构进行预测的一种方式[1-2],蛋白质的功能基本上是由蛋白质的折叠方式确定的,因此对蛋白质折叠方式

**收稿日期**:2024-11-21 **修回日期**:2025-03-25
**基金项目**:国家自然科学基金项目资助(61972444);广东省普通高校青年创新人才项目资助(2023KQNCX120);广州华商学院校级重点培育科研项目资助(2024HSZD01)
**作者简介**:徐胜超(1980-),男,通讯作者,硕士,副教授,研究方向为并行分布式处理软件。

进行模拟在生物学中具有重要意义[3]。目前采用本地计算机对蛋白质的折叠方式进行模拟计算,由于蛋白质的复杂性质通常计算会持续数十个小时乃至数天[4],如此长的计算时间会严重影响研究的进度。

研究人员开发了多种蛋白质折叠计算方法,如网格计算、集群计算等,但随着科技进步和数据增长,这些方法已无法满足需求。文献[5]提出基于氨基酸序列模拟折叠过程,通过计算能量函数评估折叠效果,并使用拐点驱动(Knee Point - Driven Evolutionary Algorithm,KNEE)算法筛选最佳结构。文献[6]探讨信使核糖核酸(Messenger RNA,mRNA)对折叠速率的影响,通过计算碱基对梯阶参量值分析折叠速率,但可能忽略了其他影响因素。文献[7]介绍利用力探针测量折叠能量格局,提供实验工具研究能量途径,但涉及复杂技术和条件限制。文献[8]描述通过小分子结合调节蛋白质折叠,改善折叠结果,但可能需要特定结合剂并存在潜在副作用。文献[9]提出的蛋白质折叠并行预测方法可提升计算效率、优化资源利用且可扩展,但算法复杂、有通信开销且依赖硬件。文献[10]提出的量子化学方法能微观精确描述、揭示本质机制、提供新视角,但计算资源需求大、模型简化难且理论理解门槛高。

OpenStack 云平台具有强大的计算能力、云服务集成能力、安全性以及社区支持。为此,该文提出一种基于 OpenStack 云平台的蛋白质折叠模拟计算方法改进型剪枝富集罗森布鲁斯方法(Modified Pruned - Enriched Rosenbluth Method, Mod - PERM)。利用 OpenStack 云平台能够很好地聚集大规模的分布式的计算资源,快速准确高效地完成蛋白质折叠模拟计算。

该方法的创新技术路线如下:

(1)通过计算机科学、生命科学等跨学科交叉,提出了基于 OpenStack 云平台的蛋白质折叠模拟计算大规模训练优化方法。

(2)使用疏水 - 亲水(Hydrophobic - Hydrophilic, HP)格点模型来简化蛋白质折叠过程,将氨基酸链视为黑球和白球的组合形式。利用折叠过程中的能量体系评价折叠结果,并选择最低能量的结果作为最佳结果。

(3)改进了精简富集罗森布鲁斯方法(Pruned - Enriched Rosenbluth Method,PERM),计算每个球的权重,以获得最佳的蛋白质折叠结果。

(4)通过将复杂的任务进行分解,利用 OpenStack 搭建分布式环境,将每个氨基酸链的计算任务分配给不同的计算节点,并运行改进 PERM 算法进行能量空间探索和模拟计算。最后,各节点计算结果汇总到主节点上,进行数据整合和结果分析,最终得出完整的蛋白质折叠计算结果。

## 1 蛋白质折叠模拟的数学模型选择

### 1.1 蛋白质折叠 HP 格点模型

在蛋白质折叠的模拟计算中,由于真实折叠过程极为复杂,研究人员采用了简化的 HP 格点模型[11-12]。该模型将蛋白质中的氨基酸分为两类:疏水性氨基酸(H)和亲水性氨基酸(P)。分别用黑球和白球表示。这样,一个氨基酸序列就转化为由黑球和白球组成的链条,其中每个球的间距固定为 1。在三维空间内,将 $n$ 个黑色或白色、半径为 $\frac{1}{2}$ 的球组成的一个链状结构便可以视为一个氨基酸序列。对于第 $i$($i = 1,2,\cdots,n$)个球,设($x_i,y_i,z_i$)为该球的球心坐标。因为球心坐标位于格点,所以球心坐标均为整数,将该氨基酸链上的球心坐标视为一个整体,该整体便是一个构形,即:($x_1,y_1,z_1,x_2,y_2,z_2,\cdots,x_n,y_n,z_n$)。对于蛋白质的折叠模拟计算,便是将构形中的球在格点空间中的位置进行调整,并且达到球心距离为 1 且能量函数最低。表达式为:

$$\min(E) = u_{ij}\sqrt{(x_i - x_{i+1})^2 + (y_i - y_{i+1})^2 + (z_i - z_{i+1})^2} \quad (1)$$

由于该问题为非线性问题,因此对于直接求解较为困难,该文通过引入新的能量对传统模型进行优化。由于疏水氨基酸单体依靠相互吸引力使氨基酸结构更加稳定,因此使用 $u_{ij}$ 描述黑球 $i$、$j$ 之间的引力。

$$u_{ij} = \begin{cases} -\frac{g_1}{r_{ij}^{21}}, r_{ij} \geqslant 1 \\ M, r_{ij} < 1 \end{cases} \quad (2)$$

式中,$r_{ij}$ 表示两球 $i$、$j$ 的球心距离,$M$ 表示正实数,$g_1$ 表示引力系数。

为保证这种情况不会发生,该文引入弹性力学,将氨基酸球体视为弹性实体,在受到挤压后拥有恢复原始形态的趋势,该趋势用公式表达为:

$$u'_{ij} = \begin{cases} 0, r_{ij} \geqslant 1 \\ 1, r_{ij} < 1 \end{cases} \quad (3)$$

在确定了引力和斥力后,最终的构形能量体系公式表达为:

$$E = u_{ij} + u'_{ij} = \sum_{i=1}^{n}\sum_{j>1}^{n} u_{ij} + \sum_{i=1}^{n}\sum_{j>1}^{n} u'_{ij} \quad (4)$$

### 1.2 用于 HP 格点模型的 PERM 算法

PERM 蛋白质扩展随机方法是一种增长型算法,用于以最小能量为原则完成 HP 格点模型求解。逐步放置黑球和白球在长度为 1 的链中,直至全部的球放完。在第 $n-1$ 的格局下,计算在第 $n$ 个球之前的格局 $n$ 权重预测值 $W_n^{\text{pred}}$ 以及对权重上下门限($W_n^{<}$,$W_n^{>}$)。当出现 $W_n^{\text{pred}} < W_n^{<}$ 情况时,保留格局 $n$ 的概率为 1/2,同时该格局的权重 $W_n$ 翻倍;如果出现 $W_n^{\text{pred}} \in [W_n^{<},$

$W_n^>$ ] 的情况,即全部保留该格局,并且权重不变;最后在 $W_n^{perd} > W_n^>$ 的情况下,在球 $n$ 的自由度 $K_{free}$ 个位置选择 $k = \min\{K_{free}[W_n^{pred}/W_n^>]\}$ 数量的分支,并且计算每个分支格局的权重,记为:( $W_n^{(1)}, W_n^{(2)}, \cdots, W_n^{(k)}$ )。

### 1.3　改进的 PERM 算法

PERM 算法在处理复杂优化问题时,可能因格局权重评判不够精确和上下门限策略设置不当而导致计算效率低下。如果格局权重的评判机制缺乏足够的适应性,可能会使算法在搜索过程中偏离最优解,增加不必要的计算开销。同时,上下门限策略如果控制不当,可能会导致算法过早收敛到局部最优解,或者在全局搜索上花费过多时间,这两种情况都会降低算法的整体效率。

为了解决这些问题,该文对 PERM 算法进行了改进,形成了 Mod-PERM 算法。改进后的算法通过引入初始变量 $C$ 和温度系数等参数,提高了算法的适应性和灵活性,从而显著提升了计算效率。初始变量 $C$ 优化了算法的初始搜索行为,帮助算法更快地定位到有希望的搜索区域。温度系数的调整则有助于平衡算法的探索和利用,使得算法在搜索初期能够进行更广泛的全局探索,而在后期则能够进行更精细的局部优化。这种策略的调整有效提高了算法的搜索效率和求解质量。在 Mod-PERM 算法中,需要预测在第 $n$ 个球放入前格局 $n-1$ 的权重 $W_n^{pred}$ ,以及计算上下门限,公式表达为:

$$W_n^{pred} = W_{n-1} K_{free} \tag{5}$$

$$W_n^> = C^> Z_n \tag{6}$$

$$W_n^< = C^< W_m^> \tag{7}$$

式中, $W_{n-1}$ 为格局 $n-1$ 权重值, $C^>$ 为初始基准值, $C^<$ 为初始变量, $Z_n$ 为 $W_n$ 期望值。$Z_n$ 为全部格局的权重的算数平均值。

比较 $W_n^{pred}$ 与上下门限( $W_n^<, W_n^>$ ),当 $W_n^{pred} < W_n^<$ 时,以 1/2 概率保留格局,以同样概率在 $K_{free}$ 个分支中同样选择分支 $i$ ,该分支格局权重为:

$$W_n = 2W_{n-1}\exp(-\beta\Delta E_i) \tag{8}$$

式中, $\beta$ 为温度系数, $\beta = \frac{1}{T}$ , $T$ 为温度, $\Delta E_i$ 为选择分支 $i$ 的能量变化。

$W_n^{perd} \in [W_n^<, W_n^>]$ 时,分支 $i$ 的格局权重为:

$$W_n = W_{n-1}\exp(-\beta\Delta E_i) \tag{9}$$

$W_n^{perd} > W_n^>$ 时, $i$ 对应的格局权重为:

$$W_n^{(i)} = \frac{W_{n-1}\exp(-\beta\Delta E_n^{(i)})K_{free}}{K} \tag{10}$$

式中, $\Delta E_n^{(i)}$ 为第 $n$ 个球放入分支 $i$ 后的能量变化。

分析上述可以发现在 $W^> = W^< = 0$ 情况下,根据式 10 即可计算出全部的分支权重, $W^> = +\infty$ 情况下,需要根据式 9 计算分支权重,分支格局的上下门限可通过式 6 以及式 7 计算,调整 $C^>$ 和 $C^<$ 控制搜索数的分支。当 $C^>$ 确定时,越小的 $C^<$ 会使 $C^<$ 搜索范围越大,从而会导致搜索速度变慢。

## 2　蛋白质折叠模拟计算的运行平台

### 2.1　OpenStack 云平台

OpenStack 目前由 Nova(计算)、Neutro(网络)、Horizon(控制面板)、Cinder(块存储)、Glance(镜像服务)、Keystone(认证服务)以及 Swift(对象存储)共七个部分组成。Mod-PERM 为保证利用 OpenStack 云平台顺利完成蛋白质折叠模拟计算,利用 HP 格点模型和改进的 PERM 算法完成蛋白质折叠模拟计算,并利用 OpenStack 平台完成蛋白质数据的存储管理、资源分配调度、子任务执行、主控程序处理,配置 Swift 对象以存储蛋白质数据,并配置 Nova 计算服务进行资源调度和子任务执行。接着,集成 Swift 和 Nova,确保它们协同工作。编写主控程序用于管理整个流程,包括提交作业和监控任务执行。最后测试和调试整个系统,确保其正常运行并能有效处理数据任务。对最终的蛋白质氨基酸折叠计算任务进行分配输出,提升蛋白质折叠模拟计算效率。

### 2.2　面向 OpenStack 云平台的蛋白质折叠任务划分

利用 OpenStack 的 Swift 进行蛋白质数据存储的具体步骤如下:

(1)创建容器。首先,在 OpenStack 的 Swift 中创建一个容器,用于存储蛋白质序列数据。容器可以通过命令行工具或 API 进行创建,并设置访问权限和元数据等信息[13-14]。

(2)上传数据。将蛋白质序列数据通过命令行工具或 API 上传至创建的容器中。可以将蛋白质数据分成多个部分进行上传,以提高并行处理的速度。

(3)数据管理和访问控制。通过 OpenStack 的 Swift 提供的功能,可以对蛋白质数据进行管理和控制访问权限。可以设置访问策略,确保只有授权用户能够访问和修改数据。

(4)数据备份和恢复。使用 OpenStack 的 Swift 提供的备份和恢复功能,可以定期备份蛋白质数据,以防止数据丢失或损坏,并在需要时进行恢复。

利用 OpenStack 的 Nova 进行资源调度与计算的具体步骤如下:

(1)资源评估。利用 Nova 提供的资源评估功能,可以根据已有的虚拟化环境信息和物理服务器资源来评估可用的资源容量和性能。

(2)资源请求与扩展。通过 Nova 的资源请求和扩展功能,可以根据蛋白质计算任务的需求,向

OpenStack 云平台请求合适的计算资源,包括虚拟机实例、中央处理器(Central Processing Unit, CPU)、内存等。

(3)资源调度和优化。利用 Nova 提供的资源调度和优化算法,根据指定的调度策略(如负载均衡、容错性等),将计算任务分配给最适合的物理服务器,并根据实际情况动态调整资源分配,以达到最佳的性能和效率。

(4)动态扩展和回收。根据蛋白质计算任务的实时需求情况,利用 Nova 的动态扩展和回收功能,自动添加或释放虚拟机实例和相应的资源,以满足计算任务的需求,同时实现资源的高效利用。

Mod-PERM 方法通过结合 OpenStack 的 Swift 和 Nova,可以实现对蛋白质数据的存储、管理和访问控制,以及对计算资源的分配与调度,从而提高蛋白质折叠计算任务的效率和可靠性[15-17]。

利用 Mod-PERM 算法进行蛋白质折叠模拟折叠计算时,对上下门限的计算需要设计 $C^>$ 和 $C^<$ 变量[18-19]。在 Mod-PERM 算法运行需要设定 $W^>$ 和 $W^<$ 的空间结构,得到 $C^>$ 和 $C^<$ 的初始值。该文利用 Python 并行编程的方式,将蛋白质折叠计算的最低能量寻优过程细分为小任务,并通过 OpenStack 云平台进行资源调度,将这些小任务分配到虚拟机上执行。利用 OpenStack API 管理云资源,并监控任务执行情况,最终将计算结果汇总处理[20-22],具体的原理如图 1 所示。

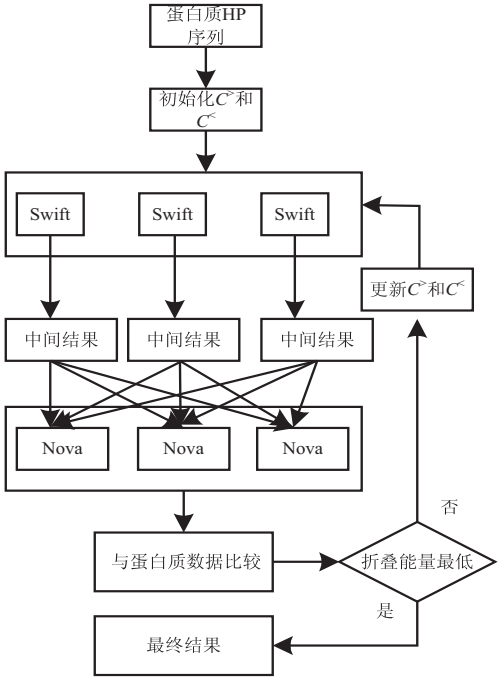



图 1 面向 OpenStack 的 Mod-PERM 任务划分与执行

## 3 实验与性能分析

### 3.1 实验数据的准备

由于 OpenStack 是一种开源应用,因此在实验中,选择物理主机自行搭建 OpenStack 云平台,搭建云平台的计算机硬件如表 1 所示,由四台物理服务器组成,安装 CentOS 操作系统。OpenStack 云平台的软件部署按照前面正文的步骤进行配置。

表 1 构建 OpenStack 云平台的计算机参数

| 节点类型 | CPU | 内存/GB | 硬盘 | 角色 | 网络 |
|---|---|---|---|---|---|
| 控制节点 | AMD 锐龙 5950 | 256 | 1 TB SSD+2 TB HDD | Master | 万兆网卡 |
| 计算节点 | AMD 锐龙 5950X | 128 | 1 TB SSD+8 TB HDD | Slave | 万兆网卡 |
| 存储节点 | AMD 锐龙 5950X | 128 | 1 TB SSD+10 TB HDD | Slave | 万兆网卡 |
| 网络节点 | AMD 锐龙 5950X | 32 | 512 GB SSD | Slave | 万兆网卡 |

其运行的主控程序部分代码如下:

```
import threading
import time
#模拟蛋白质折叠计算的大任务
protein_folding_task = "Complex Protein Folding Task"
#划分任务的函数
def divide_task(task):
#在此处对大任务进行划分,生成多个子任务
subtasks = [task[i:i+3] for i in range(0, len(task), 3)]
#以每 3 个字符为单位划分任务
return subtasks
#模拟 OpenStack 云平台资源调度和执行任务的函数
def execute_task(task):
  print(f"Executing task: {task}")
  time.sleep(2)  #模拟计算时间
  print(f"Task {task} completed.")
#主程序
if __name__ == "__main__":
  #划分大任务为小任务
  subtasks = divide_task(protein_folding_task)
  #使用多线程进行并行计算
  threads = []
  for subtask in subtasks:
    thread = threading.Thread(target=execute_task, args=(subtask,))
    threads.append(thread)
    thread.start()
  #等待所有子任务完成
  for thread in threads:
    thread.join()
```

```
print("All tasks completed. Results are ready for aggregation.")
```

用于蛋白质折叠模拟计算的蛋白质数据来源为 Protein Data Bank(PDB),网址为 Protein Data Bank(PDB):https://www.rcsb.org/。该数据库为世界最大的蛋白质结构数据库,该数据库内的数据情况如表 2 所示。

表 2　Protein Data Bank 数据库数据

| 数据类型 | 数据数量/个 |
|---|---|
| 蛋白质结构数据 | 160 000 |
| 蛋白质序列数据 | 10 000 000 |
| 蛋白质相互作用数据 | 20 000 |
| 生物大分子复合物数据 | 4 500 |

### 3.2　实验数据

在 PDB 数据库内随机挑选的 10 个氨基酸链进行蛋白质折叠模拟计算,蛋白质氨基酸链结构和目前发现的最低能量值 $E$ 结果如表 3 所示。

表 3　蛋白质数据的选择情况

| 序号 | $N$ | HP 链 | $E$ |
|---|---|---|---|
| 1 | 17 | HPPHPPHPHPPP | −5 |
| 2 | 20 | PHHHHHHPHHHHPHHPHHPP | −15 |
| 3 | 36 | $P_3H_2P_2H_2P_5H_7P_2H_2P_4H_2P_2HP_2$ | −15 |
| 4 | 48 | $P_2HP_2H_2P_2H_2P_5H_{10}P_6H_2P_2H_2P_2HP_2H_5$ | −23 |
| 5 | 50 | $H(HP)_4H_4PH(P_3H)_2P_4H(P_3H)_2PH_4(PH)_4H$ | −21 |
| 6 | 60 | $P_2H_3PH_8P_3H_{10}HP_3H_{12}P_4H_6PH(HP)_2$ | −35 |
| 7 | 64 | $H_{12}(PH)_2(P2H2)_2(P_2H)_2HP_2H(HP_2)_2(H_2P_2)_2(HP)_2H_{12}$ | −44 |
| 8 | 85 | $H_4P_4H_{12}P_6(H_{12}P_3)_3H(P_2H_2)_2P(PH)_2$ | −51 |
| 9 | 100 | $P_6HPH_2P_5H_3PH_5P_2P_2(P_2H_2)_2PH_5PH_{10}PH_2PH_7P_{11}H_7P_2HPH_3P_6HPH_2$ | −46 |
| 10 | 100 | $P_3H_2P_2H_4P_2H_3(PH_2)_3H_2P_8H_6P_2H_5P_9HPH_2PH_{11}P_2H_3PH_2PHP_2HPH_3P_6H_3$ | −49 |

### 3.3　效率测试

利用 PERM 算法与 Mod-PERM 算法对 10 个氨基酸链进行折叠模拟计算,两种算法计算出的最低能量 $E$ 以及时间 $T$ 如表 4 所示。

根据表 4 的数据分析,可以观察到几个重要的结果。首先,PERM 算法在处理较短氨基酸链的蛋白质折叠时遇到了计算难题,特别是在 5 号氨基酸链的情况下。这可能是因为 PERM 算法在折叠较短氨基酸链时,面临更多的可能构象和计算组合,导致计算复杂度增加。另外,在第 9 号和第 10 号氨基酸链的折叠计算中,观察到了最大的困难和耗时情况。这表明这些氨基酸链的结构和相互作用更加复杂,需要更多的计算资源和时间来找到最佳的折叠结果。

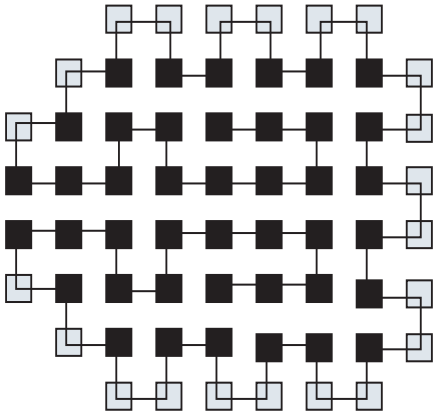
表 4　PERM 与 Mod-PERM 两种算法计算结果

| 序号 | $E_{\text{PERM}}$ | $T_{\text{PERM}}$/s | $E_{\text{Mod-PERM}}$ | $T_{\text{Mod-PERM}}$/s |
|---|---|---|---|---|
| 1 | −12 | 21 | −5 | 0.2 |
| 2 | −13 | 17 | −15 | 0.3 |
| 3 | −15 | 8.7 | −15 | 1.1 |
| 4 | −23 | 26 | −23 | 10 |
| 5 | −21 | 154 | −21 | 18 |
| 6 | −35 | 34 | −35 | 0.7 |
| 7 | −44 | 128 | −44 | 159 |
| 8 | −51 | 156 | −51 | 18 |
| 9 | −46 | 208 | −46 | 22 |
| 10 | −49 | 247 | −49 | 235 |

然而,当比较 PERM 算法与提出的 Mod-PERM 算法时,尽管两者得出的最低折叠能量相同,但在计算时间上,PERM 算法明显长于 Mod-PERM 算法。这意味着 Mod-PERM 算法在蛋白质折叠计算中具有更高的效率和优势,能够在更短的时间内给出相同质量的折叠结果。总的来说,基于对表 4 数据的分析可以得出结论:Mod-PERM 算法相对于 PERM 算法在处理较短氨基酸链和复杂结构的蛋白质折叠计算中具有更高的效率和准确性。这对于在蛋白质研究中提高计算速度和获得可靠的折叠结果具有重要意义。

### 3.4 功能测试

使用文中方法计算后的第 7 号氨基酸链折叠后形成的蛋白质结果如图 2 所示。通过图 2 的展示,可以明显看到链长为 64 的氨基酸链在经过折叠后的最低能量为-44,这与目前已知的最低折叠能量相匹配。这一结果有力地证明了文中方法能够高效地折叠蛋白质,并得到最佳的折叠结构进一步证实了该方法在蛋白质折叠模拟计算中的优越性和高效性。



*N*=64 *E*=-44

图 2 第 7 号蛋白质氨基酸链折叠结果

### 3.5 可靠性测试

为了进一步验证文中的可靠性,将计算准确率作为实验指标。分别采用文献[1]中的 Cloud-PERM 方法、文献[2]中的 Yarn PERM 方法、文献[6]方法、文献[7]方法及文中方法进行对比实验,结果如图 3 所示。

根据图 3 的分析可以得出结论:文中方法在蛋白质折叠计算中表现出更高的计算准确率,均超过 90%。这表明文中方法在预测蛋白质结构方面具有更优越的性能和准确性。与其他四种方法以及传统 PERM 方法相比,文中方法在蛋白质折叠计算任务中取得了更为显著的成绩。这种高准确率的表现可能归因于文中方法采用的创新技术或者更加有效的算法策略。此外,高达 90% 以上的计算准确率也意味着文中方法在实际应用中具有较高的可靠性和实用性,为蛋白质折叠领域的研究和应用提供了有力支持。综合以上分析,文中方法在蛋白质折叠计算方面的优势和潜力得到了充分展现,并具有广阔的应用前景。

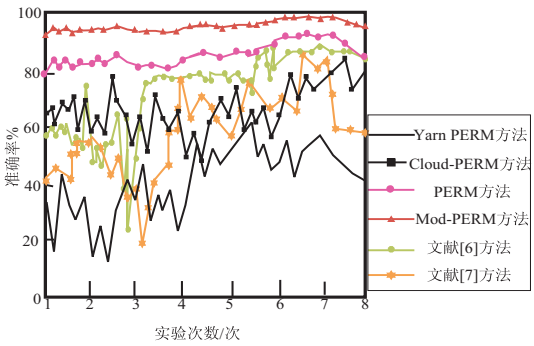



图 3 不同方法作用下的计算准确率对比结果

## 4 结束语

该文利用 OpenStack 云平台提出了一种基于 HP 格点模型和改进的 PERM 算法的蛋白质折叠模拟计算方法 Mod-PERM。与传统的 PERM 算法相比,改进后的 Mod-PERM 算法在计算时间方面具有显著优势。实验结果表明,Mod-PERM 算法给出的蛋白质折叠能量与最低能量一致,同时也与已知的最佳折叠情况相吻合。本研究的方法和结果对于加快蛋白质折叠模拟计算的速度和准确性具有重要意义,为蛋白质研究领域的进展提供了有力的支持。此外,基于 OpenStack 云平台的应用还展示了其在计算密集型任务中的潜力,为其他相似计算问题的解决方案提供了借鉴。尽管 OpenStack 云平台提供了强大的计算能力,但 Mod-PERM 算法在模拟大规模或高度复杂的蛋白质结构时,仍需要大量的计算资源,这可能对硬件和软件环境提出较高要求。未来的工作可以进一步优化算法和利用更多的云计算资源,以推动蛋白质结构预测和功能研究的发展。

**参考文献:**

[1] 徐胜超. 基于云计算的蛋白质折叠模拟计算[J]. 基因组学与应用生物学,2019,38(6):2551-2557.

[2] 宋 华,闫会峰. 面向云环境的蛋白质折叠模拟计算并行化算法[J]. 科学技术与工程,2018,18(5):258-263.

[3] YAN X,JIA Y,MAN H,et al. Tracking the driving forces for the unfolding and folding of kidney bean protein isolates: revealing mechanisms of dynamic changes in structure and function[J]. Food Chemistry,2023,402:134230.1-134230.11.

[4] ZHENG S,WEI Y,LIN Y,et al. Graphic contrastive learning analyses of discontinuous molecular dynamics simulations: study of protein folding upon adsorption[J]. Applied Physics Letters,2023,122(25):25370.1-25370.6

[5] 王雨林,张　彪,沈红斌. 基于多目标优化的蛋白质三维结构预测[J]. 江苏科技大学学报:自然科学版,2021,35(4):66-74.

[6] 程永霞,李瑞芳,赵瑞峰,等. 局部碱基对梯阶参量对蛋白质折叠速率的影响[J]. 内蒙古师范大学学报:自然科学汉文版,2022,51(2):134-141.

[7] PETROSYAN R,NARAYAN A,WOODSIDE M T. Single-molecule force spectroscopy of protein folding[J]. Journal of Molecular Biology,2021,433(20):167207. 1-167207. 19.

[8] CHITI F,KELLY J W. Small molecule protein binding to correct cellular folding or stabilize the native state against misfolding and aggregation[J]. Current Opinion in Structural Biology,2022,72(2):267-278.

[9] 李国辉,沈虎峻,张鼎林. 蛋白质折叠的并行预测方法:CN201410798385. 0[P]. 2024-09-25.

[10] 段莉莉. 量子化学方法对生物体系的研究以及蛋白质折叠[D]. 济南:山东师范大学,2010.

[11] DURHAM J,ZHANG J,HUMPHREYS I R,et al. Recent advances in predicting and modeling protein - protein interactions[J]. Trends in Biochemical Sciences,2023,48(6):527-538.

[12] DHANAPAL A,NITHYANANDAM P. An OpenStack based cloud testbed framework for evaluating HTTP flooding attacks[J]. WirelessNetworks,2021,27(8):5491-5501.

[13] 姚卫红,黄小远,方仁孝. 基于车联网应用的云平台任务调度算法[J]. 计算机仿真,2014,31(10):165-169.

[14] YUN L,XIAO Z,BINBIN L,et al. The research and analysis of efficiency of hardware usage base on HDFS[J]. Cluster Computing,2022,25(5):3719-3723.

[15] RAMDANE Y,BOUSSAID O,BOUKRA D,et al. Building a novel physical design of a distributed big data warehouse over a Hadoop cluster to enhance OLAP cube query performance[J]. Parallel Computing,2022,111(7):102918. 1-102918. 13.

[16] LEE T F,LIN K W,HSIEH Y,et al. Lightweightcloud computing-based RFID authentication protocols using PUF for e-healthcare systems[J]. IEEE Sensors Journal,2023,23(6):6338-6349.

[17] ZHANG S,LIAO J,WU S,et al. Atraceability public service cloud platform incorporating IDcode system and colorful QR code technology for important product[J]. Mathematical Problems in Engineering,2021,2021(26):5535535. 1-5535535. 15.

[18] DASARI C M,BHUKYA R. MapReduce paradigm:DNA sequence clustering based on repeats as features[J]. Expert Systems,2022,39(1):12827. 1-12827. 16.

[19] SARDAR T H,ANSARI Z. MapReduce-based fuzzy C-means algorithm for distributed document clustering[J]. Journal of The Institution of Engineers (India),Series B. Electrical eingineering, electronics and telecommunication engineering,computer engineering,2022,103(1):131-142.

[20] WANG J,LI W. Theconstruction of a digital resource library of english for higher education based on a cloud platform[J]. Scientific Programming,2021,2021(10):4591780. 1-4591780. 12.

[21] LIN D,QIAN Z,MASSIMO B,et al. Probing the protein folding energy landscape:dissociation of amyloidβ fibrils by laser-induced plasmonic heating[J]. ACS Nano,2023,17(10):9429-9441.

[22] CASIER R,DUHAMEL J. Synergeticeffects of alanine and glycine in blob-based methods for predicting protein folding times[J]. The Journal of Physical Chemistry B,2023,127(6):1325-1337.