计算机学院

并行程序设计实验报告

期末研究项目开题报告

姓名：林逸典

学号：2213917

专业：计算机科学与技术

2024 年 4 月 6 日

目录

1 实验概要

我们计划开展一系列并行程序实验，以深入探索普通高斯消元法和状态压缩 Buchberger 高斯消元化算法在多种并行环境下的性能表现。实验将涵盖 SIMD、OpenMP、MPI、GPU 以及复合并行优化等多种并行计算技术，旨在全面评估这些算法在不同计算模式下的效率与可行性。本开题报告将详细阐述实验的背景、研究现状、研究目标以及具体的实验计划。

在前期研究过程中，我们已在算法导论相关实验中对状态压缩 Buchberger 高斯消元化算法进行了初步的探索，并将相关研究成果整理上传至GitHub平台。本次实验将基于这些已有成果，进一步拓展研究范围，结合多种并行计算技术，深入探究算法的并行化潜力及优化策略。

实验将分为五个阶段进行，每个阶段将针对一种特定的并行优化方法展开研究。我们将分别实现普通高斯消元法和状态压缩 Buchberger 高斯消元化算法的并行化版本，并通过严格的性能测试和对比分析，评估各并行化策略对算法性能的影响。此外，我们还将关注算法在不同计算规模下的表现，以全面评估其在实际应用中的效能。

实验结束后，我们将对所得数据进行详细的分析与总结，并撰写实验报告。报告将包括实验数据的整理与分析、算法性能的比较与评价、优化策略的讨论与改进以及实验结论与展望等内容。

2 问题描述

2.1 普通高斯消元法

2.1.1 算法描述

高斯消元法是一种求解线性方程组的经典算法，它通过对线性方程组的增广矩阵进行一系列初等行变换，首先将其前向消元转换为上三角矩阵，随后进行回代求解以得到行最简形式，从而实现方程组的求解。在本研究中，我们聚焦于线性方程组存在唯一解的情形，以简化程序正确性的验证。

$$C = [A \quad B] \rightarrow [E \quad B']$$

其中，C 是 $n \times (n+1)$ 的增广矩阵，A 是 $n \times n$ 的系数矩阵，B 是 $n \times 1$ 的列向量，E 是 $n \times n$ 的单位矩阵，B' 是线性方程组的 $n \times 1$ 的解向量。

高斯消元法主要包含以下两个核心步骤：

前向消元：对于每一列，选择一个绝对值最大的元素作为主元 (pivot)，通过行交换将其置于主对角线上。随后，通过行加减运算，使得当前列下方所有元素为零，从而将增广矩阵转换为上三角矩阵。

回代求解：从最后一行开始，利用上三角矩阵的结构特点，逐个求解未知数的值。通过依次回代，最终得到方程组的解，并将增广矩阵转换为行最简形式。

高斯消元法以其高效性和通用性在数值计算及科学研究中占据重要地位。在本研究中，我们将以高斯消元法为基础，进行相关的并行实验，以探索其在高性能计算领域的应用潜力。

2.1.2 算法实现

Algorithm 1 高斯消元法: 前向消元

for $I \leftarrow 1$ to n **do**
  选择第 I 列中绝对值最大的元素作为主元 *pivot*
  找到主元所在的行 *pivotcol*
  交换第 I 行和第 *pivotcol* 行

将第 I 行的主元元素变为 1（行内元素除以 $pivot$）
for $i \leftarrow I+1$ to n **do**
计算乘数 $k = C[i][I]$
for $j \leftarrow I$ to $n+1$ **do**
$C[i][j] \leftarrow C[i][j] - k \times C[I][j]$
end for
end for
end for

Algorithm 2 高斯消元法: 回带求解

for $I \leftarrow n$ to 1 step -1 **do**
for $i \leftarrow I-1$ to 1 step -1 **do**
$C[i][N+1] \leftarrow C[i][N+1] - C[I][N+1] \times C[i][I]$
$C[i][I] \leftarrow 0$
end for
end for

2.2 特殊高斯消元法——状态压缩 Buchberger 高斯消元化

2.2.1 背景描述

在计算机代数和符号计算领域，Grobner 基作为多项式环中的一组特殊生成元，确保了多项式除法运算结果的唯一性，对于求解多项式方程组、理想成员判断以及多项式约化等任务具有重要意义。作为计算 Grobner 基的核心算法，Buchberger 算法是欧几里得算法和线性系统中高斯消元法的泛化实现，在算法设计领域占据显著地位。

与常规的高斯消元法相比，Buchberger 算法展现出了其独特之处。该算法利用一个仅包含 0 和 1 元素的初始消元矩阵，并通过巧妙的减法运算（退化为异或运算）来执行消元过程。消元矩阵的稀疏性进一步提升了算法的效率。此外，算法通过引入消元子和消元行的概念，以及前向消元处理策略，逐步构建出一个类上三角消元子矩阵，从而完成 Grobner 基的计算。

为了进一步提升算法的性能，我们引入了状态压缩技术来优化 Buchberger 算法。由于消元矩阵的元素仅为 0 和 1，且随着运算的进行，元素分布逐渐趋向于等比例分布，我们采用 unsigned short 类型的变量来存储数据，并利用按位异或运算来实现消元操作。这种优化方法不仅有效减少了内存占用，还显著提高了运算速度，实现了算法在时间和空间上的双重优化。

	普通高斯消元法	状态压缩 Buchberger 算法
元素	实数	0 和 1
矩阵稀疏性	均可	稀疏
运算类型	常规运算	异或运算
回代求解	是	否
运算结果	解向量	类上三角消元子矩阵
主元处理	选择并交换	前向消元处理

表 1: 普通高斯消元法与 Buchberger 算法对比

表1对比了普通高斯消元法与状态压缩 Buchberger 算法在多个方面的差异，包括元素类型、矩阵稀疏性、运算类型、回代求解、运算结果以及主元处理策略等。通过对比，我们可以更清晰地理解状态压缩 Buchberger 算法的特点和优势。

在本研究中，我们将以状态压缩 Buchberger 算法为基础，开展一系列并行实验，旨在探索特殊高斯消元法在并行计算环境下的性能表现。通过深入分析算法的并行化潜力。

2.2.2 算法实现

Algorithm 3 状态压缩 Buchberger 高斯消元化: 前向消元

```
for I ← 0 to row − 1 do
    if A 的第 I 行被置空 then
        for i ← 0 to col2 − 1 do
            if B 的第 i 行未被删除且 B 的对应状态位为 1 then
                将 B 的第 i 行赋值给 A 的第 I 行
                从 B 中删除第 i 行
                break
            end if
        end for
    end if
    if A 的第 I 行仍为空 then
        continue
    end if
    for i ← 0 to col2 − 1 do
        if B 的第 i 行未被删除且 B 的对应状态位为 1 then
            for j ← I/16 to row/16 do
                B[i][j] ← B[i][j] ⊕ A[I][j]
            end for
        end if
    end for
end for
```

3 研究现状

普通高斯消元法的并行化研究已经取得了显著的进展，涉及 SIMD、OpenMP、MPI 和 GPU 等多种并行计算技术。这些研究不仅提高了算法的计算效率，而且为大规模数据处理和复杂计算问题提供了有效的解决方案 [6] [8] [3] [1] [5] [2] [4] [7]。

相比之下，对于 Buchberger 算法高斯消元化算法的并行化研究则相对较为匮乏。尽管 Buchberger 算法在多项式代数中具有重要的地位，但其并行化研究尚待深入。特别是对于状态压缩 Buchberger 高斯消元化算法，其并行化研究几乎处于空白状态。

因此，在本次实验中，我们将充分借鉴普通高斯消元法并行化的研究成果，尝试将其应用于状态压缩 Buchberger 高斯消元化算法的并行化过程中。我们期望通过这一研究，能够填补状态压缩 Buchberger 高斯消元化算法并行化研究的空白，并为该算法的进一步优化和应用提供有益的探索。

具体来说，我们将利用 SIMD 技术的并行处理能力，对状态压缩 Buchberger 高斯消元化算法中的关键计算步骤进行并行化改造。同时，我们将借助 OpenMP 框架实现多线程并行化，以充分利用多核处理器的计算能力。此外，我们还将探索 MPI 技术在分布式环境下的应用，以进一步扩展算法的计算规模。最后，我们将研究 GPU 并行化在状态压缩 Buchberger 高斯消元化算法中的实现方法，并利

用 GPU 的高性能计算能力提升算法的执行效率。

通过本次实验，我们期望能够深入理解状态压缩 Buchberger 高斯消元化算法在并行计算中的性能特点，并为其优化和应用提供有益的参考。

4 实验环境

4.1 实验平台

本实验采用华为鲲鹏 920 处理器作为核心计算平台，其基于 ARM 架构，CPU 主频高达 2.6GHz。该平台提供了稳定可靠的计算环境，为实验的顺利进行奠定了坚实基础。此外，实验平台配备了 191GB 的内存（RAM），确保了在处理大规模数据集时的高效性能。

处理器架构	ARM
CPU 型号	华为鲲鹏 920 处理器
CPU 主频	2.6GHz
内存（RAM）	191GB

表 2: 实验环境配置

4.2 实验数据集

为了全面评估普通高斯消元法与状态压缩 Buchberger 算法的性能，我们准备了两组不同的数据集。

普通高斯消元法的数据集是通过程序随机生成的，确保了数据的唯一性和解的存在性。这些数据集涵盖了不同规模的问题，以充分检验算法在不同情况下的表现。

状态压缩 Buchberger 算法所采用的数据集则来源于相关研究，并经过严格的格式规范化和数据清洗处理。数据集的准确性与可靠性得到了充分保障。该数据集包含多个精心设计的测试样例，每个样例具有不同的消元矩阵列数、消元子行数和消元行行数，旨在全面评估算法在不同规模和复杂度问题上的性能。

以下是状态压缩 Buchberger 算法实验数据集的详细统计信息：

测试编号	test1	test2	test3	test4	test5	test6	test7	test8	test9	test10	test11
消元矩阵列数	130	254	562	1011	2362	3799	8399	23045	37960	43577	85401
消元子行数	22	106	170	539	1226	2759	6375	18748	29304	39477	5724
消元行行数	8	53	53	263	453	1953	4345	14325	14921	54274	756

表 3: 状态压缩 Buchberger 算法实验数据集

通过这组精心构造的数据集，我们能够全面评估状态压缩 Buchberger 算法在不同问题规模和复杂度下的性能表现，为算法的进一步优化和应用提供有力支持。

5 研究计划

本次实验旨在全面评估和优化高斯消元法的性能，通过采用多种并行化技术，实现算法的高效执行。具体研究计划如下：

5.1 研究目标

本研究旨在通过 SIMD、OpenMP、MPI 和 GPU 等并行化技术，对高斯消元法进行并行化改造和优化，以提高算法的计算效率和性能。

5.2 研究内容与方法

1. **SIMD 并行化实验**

   - 利用 SIMD（单指令多数据）指令集对高斯消元法的核心计算部分进行并行化改造。

   - 设计合适的并行策略和数据布局，以充分利用 SIMD 指令的并行计算能力。

   - 通过性能测试和分析，评估 SIMD 并行化对算法性能的影响，并找出潜在的优化点。

2. **OpenMP 并行化实验**

   - 采用 OpenMP（开放式多处理）框架对高斯消元法进行多线程并行化改造。

   - 通过调整线程数量、任务划分等参数，分析多线程并行化对算法性能的影响。

   - 结合负载均衡和同步机制，优化多线程的执行效率，找出最佳线程配置。

3. **MPI 并行化实验**

   - 在分布式环境下利用 MPI（消息传递接口）技术对高斯消元法进行并行化改造。

   - 设计合适的并行策略和通信机制，以减小通信开销和提高并行效率。

   - 通过增加节点数量、调整数据分布等方式扩展计算规模，评估分布式并行化对算法性能的提升效果。

4. **GPU 并行化实验**

   - 利用 GPU（图形处理器）的高性能计算能力对高斯消元法进行 GPU 并行化改造。

   - 设计适合 GPU 计算的数据结构和算法流程，优化数据布局和访存模式。

   - 利用 CUDA 或 OpenCL 等编程框架实现 GPU 并行化代码，并评估 GPU 并行化对算法性能的提升程度。

5. **复合并行优化实验**

   - 结合 SIMD、OpenMP、MPI 和 GPU 技术，实现复合并行优化的高斯消元法。

   - 设计层次化的并行策略和任务划分方案，以充分利用不同计算资源的优势。

   - 通过对比分析不同并行策略的组合效果和性能差异，找出最优的并行化方案。

5.3 预期成果

提出一种基于多种并行化技术的高斯消元法优化方案，实现高效的高斯消元法并行化代码，并评估其性能优势。

6 总结

本报告详细探究了普通高斯消元法以及特殊的高斯消元法——状态压缩 Buchberger 高斯消元化，并围绕其性能优化展开了全面的研究。通过深入的算法分析和实验设计，我们对这两种算法的性能特点、优势与局限有了更为深刻的认识。

在实验设计方面，我们充分考虑了算法的应用场景和性能需求，精心构建了具备多种计算能力的实验平台，并收集了丰富多样的数据集。这些精心准备的实验条件为我们全面评估算法性能提供了坚实的基础。

在研究计划方面，我们明确了研究目标、内容和方法，并提出了预期的成果。我们计划进一步探索更多的优化策略和技术手段，以持续提高算法的性能和稳定性。同时，我们也将关注算法在实际应用中的表现，以便更好地满足实际需求。

总之，本报告通过深入研究和实验验证，对普通高斯消元法和状态压缩 Buchberger 高斯消元化进行了全面的性能评估和优化。我们成功设计了详细的研究计划，为后续的研究奠定了坚实的基础。

参考文献

[1] 丁强，谢红梅，何贵青. 基于 mpi 的并行分布式高斯消元算法设计和评估. 系统仿真学报.

[2] 刘成军. 基于消息传递接口的线性方程组并行计算研究——以改进的高斯消元法为例. 软件.

[3] 唐俭. Pc 机上并行计算线性方程组. 华东师范大学学报 *(自然科学版)*.

[4] 李晓梅，何新芳. 块对角主元法的并行计算. 国防科技大学学报.

[5] 田希山. 用部分选主元的高斯消去法并行求解线性方程组. 电脑知识与技术.

[6] 石虎，熊健民，宋庭新. 全主元高斯消去法在有限元并行计算中的应用. In 湖北省机械工程学会设计与传动学会、武汉机械设计与传动学会 *2008* 年学术年会, 中国, 10 2008. 中国会议.

[7] 谷国太，肖汉. 求解线性方程组的 gpu 并行算法. 河南水利与南水北调.

[8] 黄询. 桩筏基础有限元分析中并行计算程序开发. Master's thesis, 华东师范大学.