南开大学

Nankai University

南　　开　　大　　学

计　算　机　学　院

并行程序设计期末研究项目开题报告

Gröbner 基计算中的高斯消去

卢艺晗

年级：2022 级

专业：计算机科学与技术

指导教师：王刚

2024 年 4 月 6 日

目录

一、 高斯消元问题定义

(一) 普通高斯消元法

1. 简介

高斯消元法（也称高斯消去法）是求解线性方程组的一个基本方法, 它可以用于决定线性方程组的解、矩阵的秩, 以及可逆矩阵的逆。高斯消元法是通过矩阵的行变换达到消元的目的, 从而将方程组的系数矩阵转化为三角矩阵, 最后获得方程组的解, 其主要包括 3 个过程: 输入数据, 消元和回代求解。

将高斯消元具体化为五个步骤 [4] 如下:

（1）增广矩阵行初等行变换为行最简形;

（2）还原线性方程组;

（3）求解第一个变量;

（4）补充自由未知量;

（5）列表示方程组通解。

2. 算法伪代码

普通高斯消元法的算法思路可用伪代码表示如下:

Algorithm 1 普通高斯消元法的串行算法

```
1: n:=size(A)
2: //消去过程
3: for k := 1  to  n do
4:     for i := k+1  to  n do
5:         factor := A[i,k] / A[k,k]
6:         for j :=k+1 to n do
7:             A[i,j] :=A[i,j]-factor*A[k,j]
8:         end for
9:         b[i] :=b[i]-factor*b[k]
10:    end for
11: end for
12: //回代过程
13: x[n] :=b[n]/A[n,n]
14: for i :=n-1 to 1  do
15:    sum :=b[i]
16:    for j :=i+1 to n do
17:        sum:=sum-A[i,j]*x[j]
18:    end for
19:    x[i] :=sum/A[i,i]
20: end for
```

(二) Grobner 基计算中的高斯消去

1. 简介

Grobner 基的计算是密码学中的一个重要问题，它是从任意一个多项式理想的一组给定生成元，计算另一组性质良好的生成元，并称为该理想的 Gröbner 基。求 Grobner 基的过程中涉及到行减法的操作可以应用高斯消去法。这里的高斯消去法与普通高斯消去法相比，主要有以下三点区别：

（1）矩阵元素的值只能是 0 或 1；

（2）Grobner 基计算中的高斯消去只有异或运算：加法运算实际为异或运算（0+0=0, 0+1=1, 1+0=1, 1+1=0）；减法运算为加法的逆运算，亦为异或运算；乘法运算（0*0=0, 0*1=0, 1*0=0, 1*1=0）实际可消。

（3）矩阵行分为两类：“消元子”和“被消元行”，在输入时给定：在消去过程中，“消元子”充当减数，所有消元子首项位置均不同，但不涵盖所有对角线元素；“被消元子”充当被减数，若恰好包含消元子中缺失的对角线 1 元素，则升格为消元子。

2. 算法伪代码

Grobner 基计算中的高斯消去算法过程用伪代码描述大致如下：

Algorithm 2 Grobner 基计算中的高斯消去的串行算法

Input: 消元子和被消元行，分批（对于百万级大规模）
Output: 结果矩阵（每个批次均被处理完毕）

```
for each 批次 do
    for each 被消元行 do
        while 非空行且首项在当前批次内或首项在当前批次且无对应消元子 do
            检查首项，减去（异或）对应消元子
        end while
        if 该“被消元行”变为空行 then
            丢弃
        else if 首项被当前批次覆盖且无对应消元子 then
            升格为消元子
        else if 首项不被当前批次覆盖 then
            该行该批次计算完成
        end if
    end for
end forreturn 结果矩阵
```

二、 高斯消元问题的研究历史和现状

高斯消元问题是解线性方程组的重要方法之一，在进行大规模运算时相比其它的普通消元方法更有优势，具有很高的研究价值。但是传统的普通高斯消元算法仍存在着计算精度不足、计算复杂度较高等问题。根据调研，前人的研究工作大多围绕这两方面展开。

(一) 非并行方向

原始的普通高斯消元法在求解线性方程组过程中，系数相除所产生的舍入误差累积带入了未知量的直接求解式，导致了线性方程组解产生误差，使其计算精度不高。胡尧、罗文俊等人 [8] 将辗转相除法融入高斯消元过程中，避开除法运算产生的舍入误差，大大提高了计算精度。不过这一举措并未考虑到计算量的大小。文传军、许定亮等人 [4] 考虑到传统高斯消元步骤不清、变化较多，将其细分固化为 5 个基本步骤，提高算法的简洁实用性。但是高斯消元中仍存在公式使用不便、编程效率不高等问题，万新儒、刘单等人 [1] 提出了“四角规则”，无需计算公式即可完成消元，并针对应用较多的“按行消元、逐行规格化”将其与“逐行规格化、按列消元”对比，指出后者更为直观且计算效率更高。为了提高计算效率，陈恳、熊哲浩等人 [9] 在“四角规则”提出的基础上，将其应用到高斯-约当消元法中，构建特殊增广矩阵，结合反向消元设计求解方法，比普通高斯法、约当法相比，计算速度均提高约 60% 左右。

(二) 并行方向

通过并行设计提升高斯消元算法性能是前人工作的主要方向之一，涉及多种分配策略算法设计，结合多种并行编程平台。

20 世纪，并行编程平台尚未发展起来，关于高斯消去法的并行优化主要集中在算法设计方面。如游兆永、李磊等人 [6] 关于三角分解等问题，提出并行选主元高斯消去法的效率更优，提升 4 倍左右。

21 世纪初，全主元高斯消去法成为求解线性方程组较为流行的并行算法。全主元算法具有较高的精度和稳定性，但其串行算法时间复杂度较大。孙济州、孙敏等人 [3] 采用多进程与多线程混合的方式对其进行并行化，并对算法进行改进；此外，考虑到多为大规模矩阵运算，他们还采用 MPI 并行 I/O 技术提高读取文件的速度，降低对内存的需求；对于选主元后换行带来的通信开销，他们采用标注数组来避免；该方法的加速比随着方程组阶数的增大而增大，且具有较好的稳定性。熊健民、宋庭新等人 [7] 提出了全主元高斯消去法在有限元并行计算中的应用，用 Java 多线程和 MPI 技术实现了有限元的并行计算，以 0.75 到 0.8 的并行效率有效解决有限元中求解速度慢的问题。

在推广并行算法的同时，高斯消元串行算法本身的特性（各消去步计算量递减、逐行消去）导致并行化时处理负载不均，限制并行算法的性能提升。因此，高斯消元的并行分配策略也在研究中受到广泛关注。马丽，李红等人 [10] 提出了分块卷帘和首尾卷帘两种策略并进行对比分析。后者的优点是有效提高通信效率，使处理器负载更均衡，而前者的主要优点是增加任务粒度减少处理器间通信次数，但存在着不能显著改善处理器负载不均衡的缺点。刘琳、刘青昆等人 [2] 基于大规模线性方程组对内存容量的要求，针对对称方程组提出只计算上三角矩阵的并行方法，数据运算量大大降低。但是其中采用的“按行卷帘”策略并行计算粒度过小，存在着通信时间影响效率的缺点。毛飞、陈智骏等人 [5] 在 CUDA 新出现的浪潮下，提出和实现了全选主元高斯-约当消去法在 GPGPU 上的实现方法，相对 Intel 最新架构 CPU 的加速比超过了 6.5 倍。其优点是利用 GPGPU 架构特点，有效避免与 CPU 的不必要通信，大幅减少整体运行时间。

(三) 现状综述

高斯消元法经过前人研究，已经探索出 LU 分解、行/列/全主元等一系列优化方法，在并行领域，也从多线程、内存访问优化、并行分配策略等方面取得了一定的研究成果。随着并行编程平台和技术的兴起，高斯消元法的并行化研究始终持续，并结合不同平台技术，在内存访问优化、数据/任务分配等方向进行着更细致的研究。

三、 研究方案

(一) 期末研究目标规划

研究目标：在充分探索普通高斯消元法并行优化策略的基础上，从数据访问顺序和消元顺序方面探索 Grobner 基计算中的高斯消元法的并行优化策略，通过四次编程实现，综合分析 Grobner 基计算中的高斯消元在并行加速问题上可以采用的策略搭配方案。四次子问题设计思路如下:

(二) 四次子问题编程规划

1. SIMD 并行化

（1）任务安排：

在 ARM 平台上设计并编程实现普通高斯消元法的 Neon 算法，对比分析对齐与不对齐、对串行算法不同部分进行并行这些不同编程策略的差异，分别测试在不同问题规模下、不同编程策略下并行算法和串行算法，并进行对比和性能分析；

选取其他指令集（SSE、AVX）进行编程实现，对比不同指令集对于并行算法性能提升的影响；

对 Grobner 基计算中的高斯消去进行并行优化，对比不同编程策略下并行算法的性能提升；

对于以上的算法和指令集，使用 VTune 剖析程序性能，深入对比分析，得出子问题结论。

（2）拟实现思路：

采用 SIMD Intrinsics 函数对普通高斯消元、Grobner 基计算中的高斯消元分别进行向量化；分别对除法、消元、回代及三者结合的部分进行四路向量化并进行对比，在 AVX 指令集中采用 8 路向量化。

2. Pthread 和 OpenMP 多线程编程

（1）任务安排：

在 ARM 平台上，分别在 Pthread 和 OpenMP 上对普通高斯消去算法进行并行化。对不同问题规模、不同线程数下，实现不同算法策略，对比串行算法和并行算法，分析性能提升效果；并横向对比 Pthread 和 OpenMp 上的性能，分析原因；

将其与 SIMD 算法结合，对比串行算法与并行算法，探究是否会有更好的性能；

对 Grobner 基计算中的高斯消元分别在 Pthread 和 OpenMP 上进行并行化；对比分析性能差异；

（2）拟实现思路：

对除法部分，采用按列划分，对消去部分的两重循环，分别采用按行划分和按列划分两种算法策略，设计相应的同步机制。

3. MPI 编程

（1）任务安排：

在 ARM 平台上，对普通高斯消去法进行基础 MPI 并行化，在不同问题规模、不同线程数下，实现不同算法策略，对比串行算法和并行算法，分析性能提升效果；

将该算法与多线程、SIMD 结合，对比串行算法与并行算法，探究是否会有更好的性能；

对 Grobner 基计算中的高斯消元进行 MPI 并行化；

（2）拟实现思路：

分别采用块划分和循环划分两种策略，使用 VTune 深入剖析程序性能，并分析时间复杂度、加速比、通信开销等因素。

4. GPU 平台

（1）任务安排：

在 ARM 平台上通过 GPU 对普通高斯消元法进行并行加速。进行普通高斯消元法的串行算法和并行算法，以及并行算法与之前并行方法的性能对比，分析对性能提升的影响。

对 Grobner 基计算中的高斯消元在 GPU 上采取相似的策略进行并行加速，与串行算法对比分析。

将其与之前的并行思路整合，探索性能更优的模式。

（2）拟实现思路：

使用两个核函数分别对除法和消去两部分的循环进行展开，一个线程块固定一行的计算，块内的线程分别负责该行不同位置上元素的运算，最后进行块内同步。使用 VTune 深入剖析程序性能。

参考文献

[1] 万新儒, 刘单, 邵尉哲, and 陈恳. 高斯消元法计算技巧的研究及应用. **电力系统及其自动化学报**, 30:109–113, 2018.

[2] 刘琳, 刘青昆, and 宋小雨. 高斯消去的并行化研究. **计算机工程**, 37:40–42, 2011.

[3] 孙济洲, 樊莉亚, 孙敏, 于策, and 张绍敏. 改进的并行高斯全主元消去法. **天津大学学报**, (09):1115–1119, 2006.

[4] 文传军, 许定亮, and 华婷. 高斯消元五步骤法. **常州工学院学报**, 25:50–53, 2012.

[5] 毛飞, 陈智骏, 梁效斐, and 曹奇英. 使用 cuda 平台关于并行高斯-约当消去法的研究与比较. **计算机应用与软件**, 28:269–271, 2011.

[6] 胡洁游兆永, 李磊. 关于并行选主元高斯消去法. **数值计算与计算机应用**, (04):207–213, 1988.

[7] 石虎, 熊健民, and 宋庭新. 全主元高斯消去法在有限元并行计算中的应用. **湖北工业大学学报**, pages 67–69, 2008.

[8] 罗文俊胡尧. 改进 gauss 消去法求解线性方程组. **贵州大学学报 *(自然科学版)***, pages 127–131, 2004.

[9] 陈恳, 熊哲浩, 魏艺君, and 廖嘉文. 分段对称反向高斯-约当消元法及其应用. **计算机仿真**, 38:310–314+338, 2021.

[10] 马丽 and 李红. 高斯消去法并行任务分配策略对比. **吉林省教育学院学报**, 25(07):153–154, 2009.