作品申报书

大规模非对称不定带宽线性代数方程组求解算法大赛-基于GPU加速的高性能ILU(0)- PBICGSTAB方法求解大规模线性代数方程组

edge-knife

# 一、基本信息

1. 作品标题： 基于GPU加速的高性能ILU(0)-PBICGSTAB方法求解大规模线性代数方程组
2. 摘要： 本项目是一种基于预处理稳定双共轭梯度法的不完全LU分解求解L-U矩阵的算法系统。该系统利用基于依赖分析的不完全LU分解作为预条件，结合C/CPP实现和C-CUDA编程，实现了高效的矩阵求解。通过自研实现关键算法瓶颈点，并利用底层语言进行性能优化，该系统在三角矩阵求解、LU分解等关键算法上的计算性能显著优于业界普遍水平，提升了10-20倍。尤其在处理L型三角稀疏矩阵时，该系统的性能超过了NVIDIA官方CUDA-TOOLKIT- cusparse方法50倍左右。这一成果为科学计算、数值分析和工程领域中的大规模矩阵问题提供了高效解决方案。
3. 作品软件分类: 开源软件/工业软件
4. 应用领域：计算流体力学/数值模拟
5. 开放源码许可证类型（可多个）：AGPL-3.0

# 背景及应用领域

船舶流体动力学是研究船舶在流体中运动规律的科学。随着科技的发展，船舶流体动力学在船舶设计、航行安全、能源节约等方面发挥着越来越重要的作用。而计算流体动力学（CFD）作为船舶流体动力学的重要分支，通过数值模拟的方法对流体流动进行模拟和分析，具有高效、经济、灵活等优点。因此，开发适用于船舶流体的CFD软件对于推动船舶工业的发展具有重要意义。

本项目旨在实现适用于船舶流体CFD软件的底层迭代算法，以提高CFD软件的计算精度和效率。本项目的应用领域非常广泛，包括但不限于以下几个方面：

船舶设计：通过CFD软件模拟船舶在各种工况下的流体动力学性能，优化船舶设计，提高航行效率；航行安全：通过CFD软件模拟和分析船舶在复杂环境下的流体动力行为，为航行安全提供保障。；能源节约：通过CFD软件模拟和分析船舶的流体动力性能，优化船舶的能源配置和利用，降低能耗和排放。；海洋工程：通过CFD软件模拟和分析海洋环境下的流体动力学问题，为海洋工程设计和运行提供技术支持。

# 国内外现状及分析对比

国内在船舶流体CFD软件领域的研究和应用起步较晚，但近年来随着国家对船舶工业的重视和投入的增加，国内在该领域的研究和应用水平得到了快速提升。国内的一些高校、科研机构和企业已经开始自主研发船舶流体CFD软件，并取得了一定的成果。同时，国内的一些软件公司也积极跟进，推出了一些商业化的船舶流体CFD软件，但与国际先进水平相比仍有一定的差距。

国外在船舶流体CFD软件领域的研究和应用起步较早，一些国际知名的流体动力学软件开发商如ANSYS、FLUENT等已经推出了商业化的船舶流体CFD软件，并在船舶工业界得到了广泛应用。这些软件在算法设计、计算精度、计算效率等方面都达到了较高的水平，成为国内外船舶流体CFD软件市场上的主导产品。

与国外相比，国内在船舶流体CFD软件领域的研究和应用水平还有一定的差距，主要表现在以下几个方面：

算法设计：国内在算法设计方面相对较为薄弱，而国外的知名CFD软件开发商通常拥有自己的算法库和工具箱，能够根据不同的流体动力学问题快速设计出高效的求解算法。

计算精度：国内在计算精度方面与国外相比还有一定差距，需要进一步提高计算精度以更好地满足实际应用需求。

计算效率：国内在计算效率方面与国外相比也有一定差距，需要进一步优化算法和软件架构以提高计算效率。

软件开发和商业化：国内在软件开发和商业化方面与国外相比还有很大的提升空间，需要进一步加强软件工程化和商业化方面的能力，提高软件的稳定性和易用性。

# 特点和设计思路

本系统基于预处理稳定双共轭梯度法，利用不完全LU分解作为预条件，结合C/CPP实现和C-CUDA编程，实现了高效的矩阵求解。其特点包括：

高效性：通过预处理技术和不完全LU分解，显著提高了计算性能，相较于业界普遍水平提升了10-20倍。

适用性：尤其适用于处理各类型稀疏矩阵求解

稳定性：采用稳定双共轭梯度法，保证了求解过程的稳定性和准确性。

易用性：提供C/CPP接口和C-CUDA编程支持，方便用户进行二次开发和集成。

本系统的设计思路主要包括以下几点：

基于预处理技术：通过不完全LU分解作为预条件，降低求解难度，提高计算效率。

优化算法瓶颈：自研实现关键算法瓶颈点，针对性地进行性能优化。

利用底层语言优化：利用C/CPP和C-CUDA编程，充分发挥底层语言的性能优势，发挥CPU GPU的计算能力，提升计算速度。

# 数据结构及功能描述

数据结构描述：

本项目涉及的数据结构主要是基于CSR格式存储的稀疏矩阵，以及使用一维数组表示的方程右端项向量和解向量。CSR（Compressed Sparse Row）是一种常用的稀疏矩阵存储格式。在CSR结构中，稀疏矩阵被压缩成三个数组：(row\_ind, col\_ind, values)。

row\_ind：存储每行非零元素的索引，不包含行尾的nnz值。

col\_ind：存储非零元素对应的列索引。

values：存储非零元素的值。

这种结构的特点是：只存储非零元素的位置和值，大大节省了存储空间。

功能描述：

本系统包含三个主要模块：

1. 文件输入输出模块：

* 功能：负责读取和写入CSR格式的稀疏矩阵数据。
* 流程：

1. 读取文件中的row\_ind、col\_ind和values。
2. 将这三个数组转换为CSR格式的稀疏矩阵。
3. 将处理后的矩阵数据传递给其他模块。

2.系数矩阵预处理模块：

* 功能：对系数矩阵进行必要的预处理操作，如排序、压缩、LU分解等。
* 流程：

1. 根据row\_ind和col\_ind对矩阵进行排序，确保矩阵的非零元素按照行优先的顺序排列。
2. 对稀疏矩阵进行压缩，去除重复和非必要的元素。
3. 对稀疏矩阵进行不完全LU分解，以加速后续方程求解速度。

3.方程组求解模块：

* 功能：基于预处理的系数矩阵，使用特定的迭代算法（如预处理稳定双共轭梯度法）进行方程组的求解。
* 流程：

1. 初始化迭代算法所需的参数。
2. 执行迭代计算，直到满足收敛条件或达到最大迭代次数。
3. 返回方程的解。

# 六、体系结构和关键技术点

作品体系结构描述：

本作品主要由三个模块组成：文件输入输出模块、系数矩阵预处理模块和方程组求解模块。各模块之间通过数据流进行交互，形成完整的系统流程。

文件输入输出模块：职责：负责读取和写入外部文件，特别是CSR格式的稀疏矩阵数据。与其他模块的关系：此模块将读取的稀疏矩阵数据传递给系数矩阵预处理模块，并将预处理后的数据传递给方程组求解模块。同时，它还负责将求解模块的结果输出到文件中。

系数矩阵预处理模块：职责：对从文件输入输出模块接收到的系数矩阵进行必要的预处理，如排序、压缩、LU分解等。与其他模块的关系：从文件输入输出模块接收原始数据，经过预处理后，将数据传递给方程组求解模块。

方程组求解模块：职责：基于预处理的系数矩阵，使用特定的迭代算法进行方程组的求解。与其他模块的关系：从系数矩阵预处理模块接收预处理后的数据，进行方程组的求解，并将结果返回给文件输入输出模块进行输出。

关键技术点：

预处理技术：预处理技术是提高迭代算法效率的关键。通过对系数矩阵进行不完全LU分解，我们可以加速迭代算法的计算过程，提高整体效率。

迭代算法：本作品采用了稳定双共轭梯度法作为迭代算法。这种方法在求解大型稀疏线性方程组时具有较高的计算效率和稳定性。

并行计算：利用现代计算机的多核处理器，和GPU的众核处理能力，我们将整个计算过程设计为并行化，从而提高整体计算速度。尤其是在矩阵乘法和迭代计算过程中，充分利用了并行化技术。

优化技术：针对关键算法瓶颈，我们进行了深度优化，包括但不限于算法实现、内存管理、线程调度等方面的优化，从而确保了系统的高效运行。

# 七、功能模块设计

* 文件输入输出模块

实现方式：使用C++的文件流操作，如ifstream和ofstream，来读取和写入文件。对于CSR格式的稀疏矩阵，我们特别定义了数据结构来存储row\_ind、col\_ind和values。

实现流程：

1. 打开指定的输入文件，读取row\_ind、col\_ind和values。
2. 将读取的数据转换为内部使用的数据结构。
3. 将数据传递给下一个模块。
4. 计算完成后，打开指定的输出文件，将结果写入。

* 系数矩阵预处理模块

实现方式：对接收到的稀疏矩阵数据进行不完全LU分解

实现流程：

1. 接收从文件输入输出模块传递过来的数据。
2. 对系数矩阵各非0元素进行LU分解依赖元素分析
3. 基于依赖元素分析，完成LU分解预处理
4. 将预处理后的数据传递给方程组求解模块。

* 方程组求解模块

实现方式：使用稳定双共轭梯度法作为迭代算法。对预处理后的数据进行迭代计算，直到满足收敛条件或达到最大迭代次数。

实现流程：

1. 接收预处理后的数据。
2. 初始化迭代算法所需的参数，如初始解、迭代次数等。
3. 执行迭代计算，直到满足收敛条件或达到最大迭代次数。
4. 将结果返回给文件输入输出模块进行输出。

# 八、源代码、部署及维护说明

源代码位于https://atomgit.com/ax-b/000006-edgeknife.git

部署及维护说明详见代码仓库

# 测试报告与用户手册

代码运行操作: ./megasolve [矩阵行数] [非零元数] [矩阵文件] [右端项文件] [初始值文件] [输出结果文件]

本项目以上面的运行方式，分别测试了两个案例：：静水状态下KVLCC2叠模（超大型油轮）湍流绕流计算以及斜航状态下SUBOFF水下潜艇湍流绕流计算。

KVLCC2算例向量维度为351864（网格数），矩阵非零元数为2426070；SUBOFF算例向量维度为1277000（网格数），矩阵非零元数为8946506。

设置精度目标误差为1e-6，相应求解时间如下，均能成功收敛

|  |  |  |  |
| --- | --- | --- | --- |
| 算例 | 数据特征 | 精度 | 耗时 |
| KVLCC2-u分量 | 向量维度 351864，非零元 242670 | 1e-6 | 41ms |
| KVLCC2-v分量 | 1e-6 | 40ms |
| KVLCC2-w分量 | 1e-6 | 41ms |
| SUBOFF-u分量 | 向量维度 1277000，非零元 8946506 | 1e-6 | 92ms |
| SUBOFF-v分量 | 1e-6 | 91ms |
| SUBOFF-w分量 | 1e-6 | 93ms |

# 十、总结

本次项目主要聚焦于开发一个基于预处理稳定双共轭梯度法的稀疏矩阵求解系统。通过三个模块的协同工作，实现了从文件输入/输出、系数矩阵预处理到方程组求解的全流程操作。

项目成果：

并行计算：通过并行计算技术，充分利用了现代计算机的多核处理器，大大提高了整体计算速度。

稳定性：通过稳定双共轭梯度法，确保了方程组求解的稳定性和准确性，避免了不收敛或发散的问题。

模块化设计：项目采用了模块化的设计理念，使得每个模块独立工作，提高了系统的可维护性和扩展性。

综上所述，本项目在多个方面取得了显著的成果，为科学计算和工程领域中的大规模矩阵问题提供了高效的解决方案。同时，我们也从中学到了宝贵的经验教训，为未来的项目开发奠定了基础。