一种基于图形处理器加速的批量 LU 分解算法

李梦月[1]，王颖[1]，马刚[1]，周赣[2]
（1. 南京师范大学南瑞电气与自动化学院，江苏 南京 210023；
2. 东南大学电气工程学院，江苏 南京 210096）

摘　要：潮流计算是电力系统计算的基础，其核心是 LU 分解计算，因此电力系统潮流计算加速的关键在于 LU 分解加速。当前，基于中央处理器（CPU）的并行算法已经成熟，性能提升空间有限。图形处理器（GPU）作为协处理器，在科学计算方面具有强大的优越性，被广泛应用到电力系统潮流计算中。文中首先分析了 GPU 结构和并行运行架构，然后介绍了 LU 分解原理，并选择了合适的矩阵排序算法和稀疏矩阵存储模型，借助统一计算设备架构（CUDA）编程模型实现了基于 GPU 的单个 LU 分解和批量 LU 分解并行加速，最后在仿真设备上测试了 5 个不同的案例，对比分析其并行算法的加速效果。仿真测试结果表明，基于 GPU 的批量稀疏 LU 分解并行算法，平均可以获得 25～50 倍的加速效果。

关键词：图形处理器；潮流计算；批处理；LU 分解；并行算法
中图分类号：TM74　　**文献标志码**：A　　**文章编号**：2096-3203（2019）02-0057-07

0　引言

电能是涉及国计民生的基础性资源，随着电网的发展，其规模逐步扩大，结构更加复杂，电能安全和稳定性等问题面临严峻挑战。电力系统潮流计算是电力系统分析的基本运算，计算过程本质上可归结为对大规模稀疏矩阵的求解，研究如何提高潮流计算速度具有重要意义。潮流计算算法大致有牛顿-拉夫逊法、PQ 分解法和高斯-赛德尔迭代法，传统电力系统潮流计算以牛顿-拉夫逊法为主[1—3]。求解方法主要有直接法[4—6]和迭代法[7—8]两种。迭代法相较于直接法更适合并行计算，但是其不稳定的特征容易导致出现收敛速度慢甚至不收敛等问题，而且精度要求高时迭代次数明显增加。

文中求解的是规模小于一万阶的矩阵，可采用精确度较高的直接法。基于直接法时，降低其算法复杂度、提高计算效率和精度的关键在于 LU 分解算法的并行加速[9]。随着多核中央处理器（CPU）的迅速发展，基于多核并行加速的研究逐渐展开。文献[10]与文献[11]分别针对大规模潮流计算和三相不平衡配电网潮流计算的特征，以牛顿法为基础，提出了一种共享存储模式下对称多核并行计算方法。文献[12]提出了一种快速求解大规模安全约束最优潮流问题，利用多核计算机多线程编程实现算法并行加速。但是基于多核 CPU 并行潮流计算算法性能已经成熟，算法性能提升空间不大。

收稿日期：2018-11-07；修回日期：2018-12-11
基金项目：国家自然科学青年基金资助项目（51607093）

图形处理器（GPU）作为协处理器，有着 CPU 所不具备的多执行单元、大内存带宽和低成本等有利因素，性能大幅提升以及可编程性使其在电力行业的应用越来越广泛[13]。文献[14]提出了一种基于道路树分层的稀疏矩阵处理方法，并利用统一计算设备架构（CUDA）在 GPU 上并行加速。GPU 性质类似流处理器，适合一次进行大量相同的工作，对于不能高度并行化的工作，CPU 更具优势。算法若对 CPU 与 GPU 进行任务划分，采用异构系统实现 LU 分解加速，将具有很高的工程应用价值[15—17]。

针对潮流计算特性和 GPU 并行加速的特点，文中建立了一种基于 GPU 并行加速的批量 LU 分解算法，主要应用在潮流计算当中，考虑了矩阵预处理和批量 LU 分解加速等问题，最后采用 CPU+GPU 异构系统在仿真平台上测试了 5 个不同算例，对比分析并行算法的加速效果。

1　GPU 与并行计算技术

协处理器 GPU 的设计目标是在单位面积上实现极高的存储器带宽以及超强的运算能力，因此 GPU 含有大量的执行单元，运行大量相对简单的线程，这些线程类型高度统一且相互无依赖。若当前线程在等待数据，可先处理另一个已就绪的待处理线程，缓冲隐藏度较高。GPU 的性质类似于流处理器，比较有弹性，擅长逻辑控制、串行的运算，可以同时进行变化较多的工作。程序运算平台 CUDA 的程序线程结构如图 1 所示[18]。

CUDA 程序线程模型包含了两个并行逻辑层：

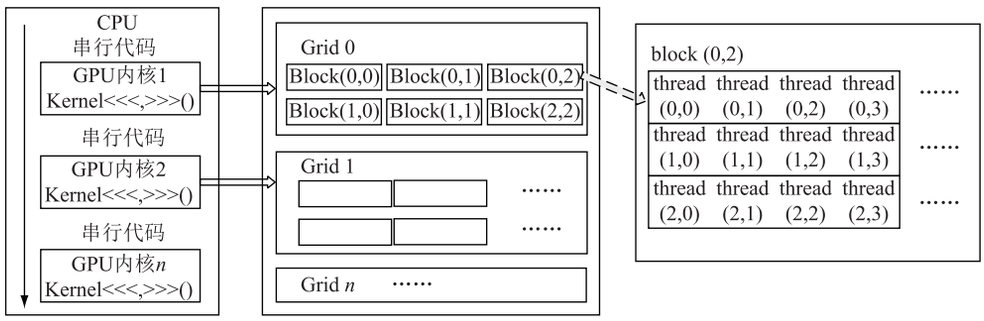

图 1　CUDA 程序线程结构

Fig.1　CUDA program thread structure

块(Block)层和线程(Thread)层。在执行时,Block 映射到流多处理器,Thread 映射到流处理器。

2　LU 分解数学模型

电力系统潮流计算本质上是形如 $Ax=b$ 的方程式中对矩阵 $\boldsymbol{A}$ 的求解,目前潮流计算方程的阶数通常都在几千阶,直接法求解的可行性较高。潮流计算核心的部分是 LU 分解,基于 GPU 加速的潮流算法主要就是在 LU 分解部分进行加速。LU 分解后,$\boldsymbol{A}x=b$ 可以用 $\boldsymbol{LU}x=b$ 表示,其中 $\boldsymbol{LY}=b$,$\boldsymbol{Y}=\boldsymbol{U}x$,则:

第 1 次填充:

$$U_{1j}=A_{1j}\quad j=1,2,\ldots,N \tag{1}$$

$$L_{i1}=A_{i1}/U_{11}\quad i=1,2,\ldots,N \tag{2}$$

第 r 次填充:

$$A_{rj}=\sum_{k=1}^{r-1}L_{rk}U_{kj}+U_{rj} \tag{3}$$

由上式可得:

$$U_{rj}=A_{rj}-\sum_{k=1}^{r-1}L_{rk}U_{kj} \tag{4}$$

$$L_{ir}=\frac{A_{rj}-\sum_{k=1}^{r-1}L_{ik}U_{Kr}}{U_{rr}}\quad i=r+1,r+2,\ldots,N \tag{5}$$

由式(4)可得,$\boldsymbol{U}$ 矩阵中第 r 行 j 列元素的值是 $\boldsymbol{A}$ 矩阵中 r 行 j 列元素减去两个向量的积。其中一个向量是 $\boldsymbol{U}$ 矩阵中第 j 列的 1 至 r 个元素组成的列向量,另一个是 $\boldsymbol{L}$ 矩阵中第 r 行的 1 至 r 个元素组成的行向量。同理可得 $\boldsymbol{L}$ 矩阵 i 行 r 列元素,如式(5)所示。

3　LU 分解并行算法的实现

根据 LU 分解的基本原理,结合 CPU 擅长逻辑运算、GPU 擅长高密度运算的特点,设计了适用于 GPU 并行计算平台的批量 LU 分解基本步骤,如图 2 所示。

GPU 具有共享内存的众核结构,且拥有更高的内存带宽,满足该算法对共享内存的要求,具有更高的可行性。考虑到 GPU 擅长的是逻辑简单并行化较高的任务,故而把逻辑复杂、且为单线程的任务放在 CPU 上执行[19]。

图 2　批量 LU 分解流程

Fig.2　Flow chart of batch-LU decomposition

3.1　预处理模块

基于 GPU 的单个 LU 分解加速性能毕竟有限,要想充分利用 GPU 的并行处理能力,还需要进一步地并行化分析,即批量 LU 分解。

数据在进行批处理之前首先要经过统一的数据预处理,有以下 3 个步骤:

(1) 利用按行压缩存储格式(CSR)对待处理的大型稀疏矩阵进行压缩处理;

(2) 通过近似最小度算法(AMD)进行矩阵排序,交换待处理矩阵 $\boldsymbol{A}$ 的行和列,以减少填入操作来降低运算量;

(3) 利用 Left-looking 算法对待处理矩阵 $\boldsymbol{A}$ 进行符号分解来形成消去树,通过确定矩阵 $\boldsymbol{A}$ 的稀疏结构来确认 LU 分解过程中非零元素填充位置。

3.1.1　稀疏矩阵的存储

电力网络的特点决定了潮流计算过程中会出现大量稀疏矩阵,待求解矩阵含有大量零元,容易

造成存储空间的浪费[20]。专门针对大型稀疏矩阵的存储方法有很多,常见的存储格式有 CSR 格式(见图 3)和坐标表示存储格式(COO)。

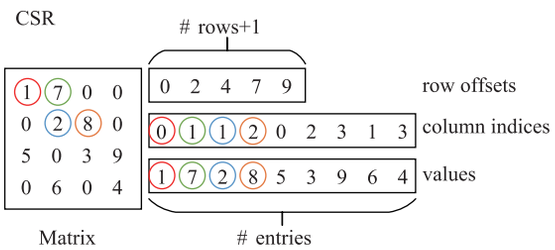

图 3 CSR 存储格式

Fig.3 CSR storage format

COO 格式最简单,但需要记录的信息较多,每个三元组自己可以定位,因此空间不是最优。CSR 由数值(values)、列号(column indices)和行偏移量(row offsets)来表示,是整体编码的方式而不是三元组。其中,"行偏移量"是指矩阵中非零元素的起始偏移位置。相对于 COO 格式,CSR 格式原理简单、压缩效果好,是一种很有效的存储方案,因此文中采用 CSR 格式。

3.1.2 矩阵排序算法

在数据预处理模块中,将待处理矩阵进行重排序,可以减少填入操作来降低运算量。比较常见的算法有 AMD 算法、反向 Cuthill-McKee 排序算法(RCM)以及列近似最小度排序算法(COLAMD)。根据各种算法的特性,文中选择的是基于外部程度近似的 AMD 算法,算法速度比其他最小度算法快,而且通常会产生更好的排序,与其他排序算法相比最具竞争力。

AMD 算法通过减少带状系统(矩阵)的带宽,在预处理阶段为大型并行计算减少内存占用。算法处理的矩阵必须是对称矩阵,首先要将矩阵 $\boldsymbol{A}$ 对称化,先求出 $\boldsymbol{A}+\boldsymbol{A}^{\mathrm{T}}$。经 AMD 排序算法处理后,可得矩阵 $\boldsymbol{A}$ 的对称最小度排列向量 $\boldsymbol{P}$,可极大降低 LU 分解过程中对对称矩阵 $\boldsymbol{P}(\boldsymbol{A}+\boldsymbol{A}^{\mathrm{T}})\boldsymbol{P}^{\mathrm{T}}$ 注入非零元的操作。

3.1.3 Left-looking 算法

根据矩阵元素的分布形式及其不同的乘法法则,可以将基于 GPU 加速的 LU 分解划成多种形式,常见的有 Left-looking 算法和 Right-looking 算法。Left-looking 算法和 Right-looking 算法的不同之处在于:Right-looking 算法元素计算过程中不存在依赖关系;Left-looking 算法每次迭代计算,部分计算数据来自当前列左边部分元素,迭代计算过程中各列相互影响,这种依赖关系可由消去树分层表示。

Left-looking 算法的程序和运算示意如图 4 所示,矩阵的绿色网格区域元素为待消去运算元素,灰色区域元素为待归一运算元素,只有灰色区域(当前列)归一运算完成之后绿色区域才能进行消去运算。

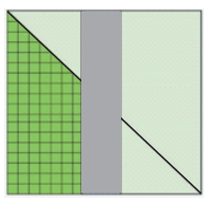

图 4 Left-looking 算法

Fig.4 Left-looking algorithm

Left-looking 算法较强的数据依赖性使其并行化有些困难,并且数据间的高依赖性使其只能高效运行在共享内存的计算设备上,这种依赖特性一方面限制了不同层次间的数据处理无法同时进行,另一方面也说明了同一层次的列之间没有依赖关系。因此,同一层次内可以实现并行处理运算,尤其针对大规模电力网络,由于支路总数远远大于层次数,分层更能够发挥并行运算的优势,提高了计算速度。

3.2 批量 LU 分解

基于 GPU 并行加速的 LU 分解模块非常重要,但是基于 GPU 加速的单个 LU 分解并行度毕竟有限,GPU 大量的处理单元得不到有效利用,而批处理 LU 分解具有良好的加速效果,可以获得更高的并行度。

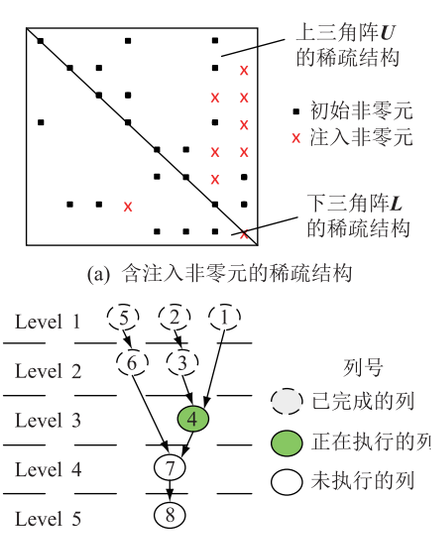

图 5 8 阶矩阵的并行性分析

Fig.5 Parallelism analysis of an 8-order matrix case

以图 5 所示的简单 8 阶稀疏矩阵为例来研究 LU 分解的并行化方法。由上文可知,稀疏 Left-looking LU 分解中各列的更新操作的依赖性是由上

三角矩阵 $\boldsymbol{U}$ 的稀疏结构决定的，及矩阵符号分解后的稀疏结构。

图 5(a)中已经标明了矩阵的原始非零元以及注入元的位置，可用消去树来描述矩阵 $\boldsymbol{U}$ 所含的依赖关系，通过消去树指引消去过程。

图 5(b)中，消去树中 Level 1 中的列 1 指向 Level 3 中的列 4，这表明，列 4 的数据更新依赖于列 1，则称节点 4 是节点 1 的“父节点”，且是唯一的一个“父节点”，对应节点 1 是节点 4 的“子节点”，但不是唯一的一个，因为节点 4 的子节点还有 Level 2 中的节点 3。由此可以得出节点 i 的父节点 j 应满足如下范围：

$$j=\min\{k=\mid u_{ik}\neq 0,1\leqslant i\leqslant k\leqslant n\} \quad (6)$$

式中：u_{ik} 为上三角矩阵 $\boldsymbol{U}$ 中的元素；n 为矩阵 $\boldsymbol{U}$ 的阶数。

无子节点的节点放在消去树的 Level 1 中，对于有子节点的节点所在层的层号则满足：

$$d_j=\max\{d_i\mid i\in M\}+1 \quad (7)$$

式中：d_i 为 i 节点的层号；M 为节点 j 对应的所有子节点构成的集合。

消去树分层的目的在于找出互相没有依赖关系的节点(同一层上节点无依赖关系)，矩阵在进行消去操作时节点不互相影响，故这些节点的消去操作可以在 GPU 上并行处理，增加并行度。

批处理 LU 分解方法来求解一组相似的稀疏线性方程组 $\boldsymbol{A}_i x_i=b_i(i=1,2,\cdots)$，并行化的前提条件是每个稀疏矩阵 $\boldsymbol{A}_i$ 具有相同或者相近的稀疏结构且数值上的差异不大。

文中在设计批处理 LU 分解时遵循了以下几个原则：

(1) 统一预处理格式。由于在 LU 分解过程中，所有待处理的稀疏矩阵 A_i 使用的都是相同的消去树，因此都要使用统一的稀疏格式以获得统一的非零元结构。出于同样的考虑，预处理模块也是如此，必须使用统一模块。

(2) 批量处理。如图 5 所示，Level 1 有 3 个节点可以并行处理，而 Level 3、Level 4、Level 5 仅有一个节点，并行度很低，基于 GPU 加速的单个 LU 分解在这阶段类似完全串行，而在批量处理结构中，有多个 LU 分解子任务中同时进行，大大增加了 Level 3—Level 5 节点更新操作的并行度。

(3) 合并访问。GPU 有着多层存储空间，每次进行数据更新操作需要访问多个存储空间。批处理数量较少时，GPU 数据处理速度极快而获取数据缓慢，这大大降低了执行效率；随着批处理数量增加，采用合并访问思想，可大幅提升访问效率，GPU 并行处理效率得以有效提升。

基于 GPU 的批处理操作可以并行处理多个 LU 子任务，以上文 8 阶稀疏矩阵 LU 分解算法为例，其算法设计如图 6 所示。在同一层内的列 1、列 2 和列 5 之间没有依赖关系，因此可以实现并行运算。

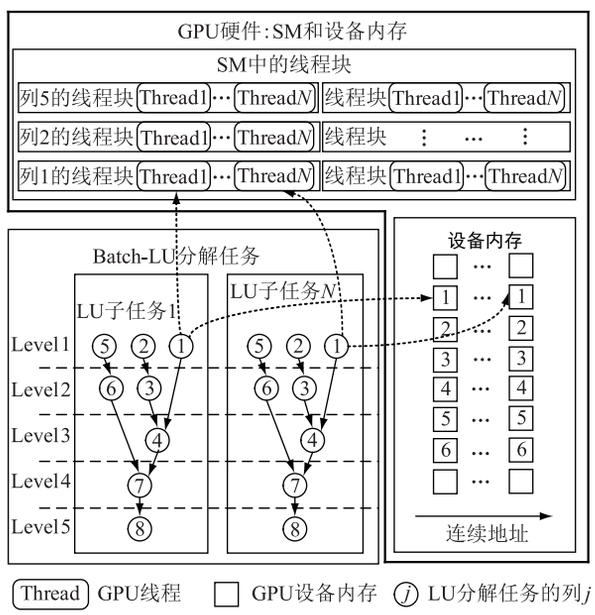

图 6 稀疏批处理 LU 算法设计

Fig.6 Scheme design of sparse batch-LU solving algorithm

4 算例分析

文中使用的仿真平台配置的 CPU 型号为 Intel E7200，GPU 为 GeForce 9500 GT，利用 Visual Studio 2015 + CUDA 9.1，实现了基于 CPU 的单个 LU 分解和基于 GPU 的单个及批量 LU 分解。测试算例基本信息如表 1 所示，批处理 LU 分解加速性能测试结果如表 2 所示。

表 1 算例基本参数信息

Table 1 Basic parameter information of the examples

算例	名称	节点数	支路数	发电机数
1	IEEE-300	300	411	69
2	江苏电网	1 079	1 830	153
3	Case2383wp	2 383	2 896	327
4	华北电网	4 273	6 266	961
5	Case9241pegase	9 241	16 049	1 445

其中，S_b 表示批处理数量，$S_b=1$ 表示单个 LU 分解，表内数据表示不同算例批量处理 S_b 个任务的总耗时。为了方便对比同规模算例的加速效果，用耗时比上批处理数量，即平均耗时来比较，如图 7 所示，虽然总耗时随批处理数量的增大而增大，但平

表 2 批处理 LU 分解耗时

Table 2 Calculation time of batch-LU solver ms

S_b	分解耗时				
	算例Ⅰ	算例Ⅱ	算例Ⅲ	算例Ⅳ	算例Ⅴ
1	0.31	0.74	1.64	1.25	5.46
32	0.33	0.78	1.71	1.30	5.90
64	0.33	0.82	1.84	1.45	6.58
128	0.35	0.89	2.07	1.74	8.30
256	0.37	1.04	2.63	2.24	12.54
512	0.45	1.37	4.57	3.63	22.50
1 024	0.65	1.99	8.70	6.50	43.00
2 048	1.14	3.40	16.70	12.40	83.90

均耗时呈下降趋势,表明了并行算法的优越性。

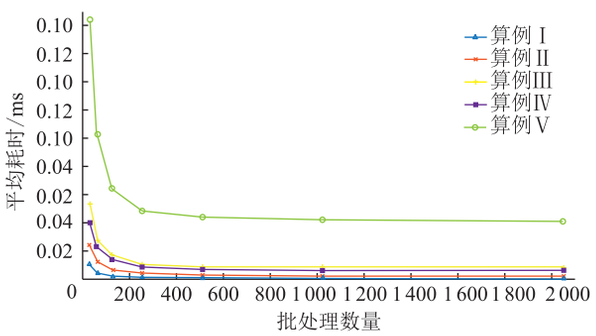

图 7 批处理不同规模 LU 分解平均耗时

Fig.7 Average compulation time of batch-LU Solving for different size

图中 y 轴表示的是 GPU 处理批量 LU 分解时单个任务耗时,x 轴表示的是批处理数量。可以看出:当批处理数量很小时,批处理平均耗时较长;随着批处理数量的增加,平均耗时直线下降,优越性逐渐显现;当批处理数量进一步增多到一定程度,批处理平均耗时趋于平稳,并行处理进入饱和阶段。

为了进一步证明 GPU 的并行加速效果,根据表 2 绘制对比图,如图 8 所示。图 8(a)为 CPU 串行 LU 分解耗时与 GPU 批处理量为 64 的平均耗时对比图,可以看出批处理 LU 分解 GPU 加速性能的绝对优势;图 8(b)为批处理数量为 64 和 1 024 时平均耗时对比图,可以看出批处理规模越大,GPU 并行加速效果越明显。

为了进一步分析算法的并行加速效果,方便 CPU 串行处理速度和 GPU 批量处理平均速度的对比,定义加速比:

$$S = t_1/t_2 \tag{8}$$

式中:t_1为在 GPU 批处理并行算法下运算一个任务的平均耗时;t_2为基于 CPU 的算法下完成同一个任务的耗时[21]。不同算例 GPU 并行加速效果用加速比如表 3 所示。

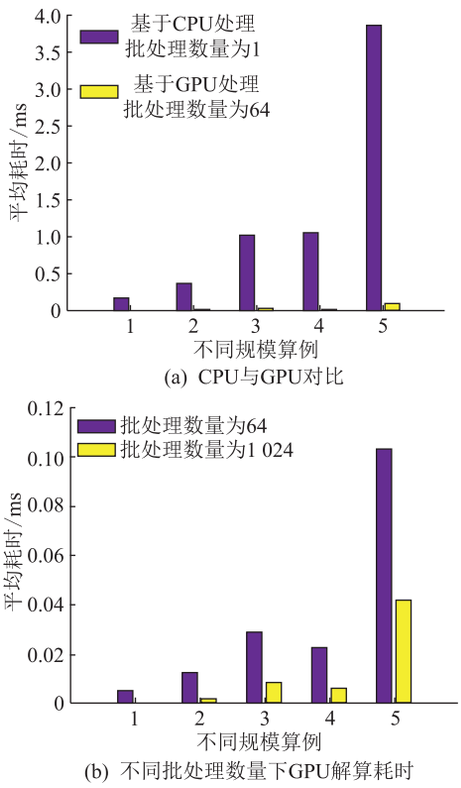

图 8 LU 分解耗时对比图

Fig.8 Time-consuming comparison chart of LU decomposition

表 3 GPU 并行加速效果

Table 3 GPU parallel acceleration effect

算例	平均耗时 (S_b=64)/ms	耗时 CPU /ms	S_1	平均耗时 (S_b=1 024)/ms	S_2
1	0.005 156	0.170 9	33.15	0.000 635	8.12
2	0.012 813	0.363 4	28.36	0.001 943	6.60
3	0.028 750	1.019 8	35.47	0.008 496	3.38
4	0.022 656	1.056 7	46.64	0.006 348	3.57
5	0.102 813	3.864 7	37.59	0.041 992	2.45

由表 3 可明显看出,在不同规模的算例下,GPU 批处理 LU 分解并行加速效果十分明显,在批处理数量为 64 时,相对于基于 CPU 分块 LU 分解算法求得的结果,平均有 25~50 的加速效果,加速效果最高可达 46.64 倍。仿真验证结果表明 GPU 并行加速的可行性,尤其是在批处理方面加速效果明显,在电力系统灵敏度分析等领域有着广阔的前景。

5 结论

针对电网的稀疏特性和 GPU 的架构特点,文中提出实现一种基于 GPU 并行加速的 LU 分解算法,主要应用在潮流计算当中。除了基于消去树的 Left-looking 直接求解算法,充分利用了 GPU 的大量执行单元求解稀疏线性方程组。为了进一步减小

算法复杂度，文中选择了算法速度更快的 AMD 排序算法，并用 CSR 格式存储稀疏矩阵。在 LU 分解过程中，把逻辑复杂且为单线程的任务放在 CPU 上执行，所有逻辑简单并行化较高任务放在 GPU 上，故而减少了数据在 CPU 和 GPU 之间的传输时间，提高了算法的运行速度。仿真验证结果表明，基于 GPU 的批量稀疏 LU 分解并行算法，相对于基于 CPU 分块 LU 分解算法，平均可以取得 25～50 倍的加速效果，在电力系统潮流并行计算方面具有一定的实用价值。

参考文献：

[1] 夏沛，汪芳宗. 大规模电力系统快速潮流计算方法研究[J]. 电力系统保护与控制，2012，40(9)：38-42.
XIA Pei，WANG Fangzong. Study on fast power flow calculation methods of large-scale power systems[J]. Power Systems Protection and Control，2012，40(9)：38-42.

[2] 涂志军，曹晔，李伟，等. 一种新型高斯赛德尔算法在电力系统潮流计算中的应用研究[J]. 江西科技，2010，28(6)：807-813.
TU Zhijun，CAO Ye，LI Wei，et al.Application research of a new gaussian seidel algorithm in power system power flow calculation [J]. Jiangxi Science，2010，28(6)：807-813.

[3] 李俊，肖宏，唐荣平，等. 一种潮流计算的 PQ 改进算法研究[J]. 大众科技，2014，16(10)：103-105.
LI Jun，XIAO Hong，TANG Rongping，et al. Research of power flow calculation based on an improved PQ algorithm [J]. Popular Science & Technology，2014，15(4)：103-105.

[4] 刘乐. 基于牛顿拉夫逊法的含分布式电源配电网潮流计算[D]. 石家庄：河北科技大学，2018.
LIU Le. Power flow calculation with distributed generation distribution network based on newton raphson method[J]. Shi jiazhuang：Hebei University of Science and Technology，2018.

[5] 陈珩. 电力系统稳态分析[M]. 北京：中国电力出版社，2016：97-107.
CHEN Heng. Power system steady state analysis[m]. Beijing：China Electric Power Press，2016：97-107.

[6] 徐晓飞，曹祥玉，姚旭，等. 一种基于 Doolittle LU 分解的线性方程组并行求解方法[J]. 电子与信息学报，2010，32(8)：2019-2022.
XU Xiaofei，CAO Yuxiang，YAO Xu，et al. Parallel solving method of linear equations based on doolittle LU decomposition [J]. Journal of Electronics & Information Technology，2010，32(8)：2019-2022.

[7] VAN A R A M. A general purpose version of the fast decoupled load flow[J]. IEEE Tans PWRS，2012，4(2)：760-770.

[8] 邓兴升，孙虹虹. 自适应谱修正 LU 分解法解算高病态法方程[J]. 大地测量与地球动力学，2014，34(6)：135-139.
DENG Xingsheng，SUN Honghong. Self-adaptive spectrum correction LU decomposition algorithm for solving a normal equation with severely ill-conditioned matrix[J]. Journal of Geodesy and Geodynamics，2014，34(6)：135-139.

[9] 张国芳，罗雅迪，李静，等. 大电网潮流修正方程并行求解实现方法[J]. 电力系统保护与控制，2017，45(19)：117-122.
ZHANG Guofang，LUO Yadi，LI Ting，et al. Parallel solution method of power flow correction equation for large-scale power grid[J]. Power System Protection and Control，2017，45(19)：117-122.

[10] 马洲俊，樊飞龙，王勇，等. 基于多源异构数据的配电网故障信息挖掘与诊断[J]. 供用电，2018，35(8)：31-39.
MA Zhoujun，FAN Feilong，WANG Yong，et al. Distribution network fault information mining and diagnosis based on multi-source heterogeneous data [J]. Distribution & Utilization，2018，35(8)：31-39.

[11] 孟晓丽，唐巍，刘永梅，等. 大规模复杂配电网三相不平衡潮流并行计算方法[J]. 电力系统保护与控制，2015，43(13)：45-51.
MENG Xiaoli，TANG Wei，LIU Yongmei，et al. Parallel computing of three-phase unbalanced power flow in large-scale complex distribution network[J]. Power System Protection and Control，2015，43(13)：45-51.

[12] 傅志生，白晓清，李佩杰，等. 一种快速求解大规模安全约束最优潮流的多核并行方法[J]. 电力系统保护与控制，2015，43(3)：29-37.
FU Zhisheng，BAI Xiaoqing，LI Peijie，et al. A high-speed multi-core parallel method for solving large-scale security constrained OPF[J]. Power System Protection and Control，2015，43(3)：29-37.

[13] 夏俊峰，杨帆，李静，等. 基于 GPU 的电力系统并行潮流计算的实现[J]. 电力系统保护与控制，2010，38(18)：100-103.
XIA Junfeng，YANG Fan，LI Jing，et al. Implementation of parallel power flow calculation based on GPU[J]. Power System Protection and Control，2010，38(18)：100-110.

[14] 陈德扬，李亚楼，江涵，等. 基于道路树分层的大电网潮流并行算法及其 GPU 优化实现[J]. 电力系统自动化，2014，38(22)：63-69.
CHEN Deyang，LI Yalou，JIANG Han，et al. A parallel power flow algorithm for large-scale grid based on stratified path trees and its implementation on GPU[J]. Automation of Electric Power Systems，2014，38(22)：63-69.

[15] 陈颖，林锦贤，吕暾. LU 分解和 Laplace 算法在 GPU 上实现[J]. 计算机应用，2011，31(3)：851-855.
CHEN Ying，LIN Xianjin，LYU Yun，et al. Implementation of LU decomposition and Laplace algorithms on GPU [J]. Journal of Computer Applications，2011，31(3)：851-855.

[16] 唐坤杰，董树锋，宋永华. 基于不完全 LU 分解预处理迭代法的电力系统潮流算法[J]. 中国电机工程学报，2017，37(S1)：55-62.
TANG Kunjie，DONG Shufeng，SONG Yonghua. Power flow algorithm based on iterative method with incomplete LU decomposition preconditioning [J]. Proceedings of the CSEE，2017，37(S1)：55-62.

[17] LI X, LI F. GPU-based power flow analysis with Chebyshev Preconditioner and conjugate gradient method[J]. Electric Power Systems Research, 2014, 116(11): 87-93.

[18] FANBER R. CUDA application design and development[M]. Waltham: Elsevier, 2011: 133-145.

[19] 卢凤顺,宋君强,银福康,等. CPU/GPU 协同并行计算研究综述计算机科学[J]. 计算机科学,2011,38(3):5-9.
LU Fengshun, SONG Junqiang, YIN Fukang, et al. Survery of CPU/GPU synergetic parallel computing[J]. Computer Science, 2011, 38(3): 5-9.

[20] 鲍威,凌卫家,张静,等. 含 VSC-MTDC 的交直流混合电网潮流计算模型及稀疏性处理技术[J]. 电力自动化设备,2016,36(5):43-48.
BAO Wei, LING Weijia, ZHANG Jing, et al. Power flow calculation model and parse matrix disposal techniques for AC/DC hybrid power system with VSC-MTDC[J]. Electric Power Automation Equipment, 2016, 36(5): 43-48.

[21] 秦成明. 基于多核 CPU 加速的电力系统并行潮流算法研究[D]. 南京:东南大学,2017.
QIN Chengming. Parallel power flow algorithm for power system based on mutli-core CPU acceleration[J]. Nanjing: Southeast University, 2017.

作者简介:


李梦月

李梦月(1996),女,硕士在读,研究方向为电力系统分析、新能源发电及入网技术(E-mail:15850633368@163.com);

王颖(1995),女,硕士在读,研究方向为新能源发电及入网技术;

马刚(1984),男,博士,副教授,研究方向为新能源发电及入网技术、电力系统分析及故障诊断相关技术。

A GPU-accelerated algorithm of batch-LU decomposition

LI Mengyue [1], WANG Ying[1], MA Gang[1], ZHOU Gan[2]

(1. Nari School of Electrical Engineering and Automation, Nanjing Normal University, Nanjing 210023, China;
2. School of Electrical Engineering, Southeast University, Nanjing 210096, China)

Abstract: Power flow calculation is the basis of power system calculation, and its core is LU decomposition calculation. Therefore, the key to power system power flow calculation acceleration is LU decomposition acceleration. Currently, parallel algorithms based on central processing units (CPU) have matured and limited space for performance improvement. As a coprocessor, the graphics processor (GPU) has powerful advantages in scientific computing and is widely used in power system power flow calculation. This paper first analyzes the GPU structure and parallel operation architecture, then introduces the LU decomposition principle, and selects the appropriate matrix sorting algorithm and sparse matrix storage model. The GPU-based single LU decomposition is realized by the unified computing device architecture (CUDA) programming model. Parallel acceleration with batch LU decomposition. Finally, five different cases were tested on the simulation device, and the acceleration effect of the parallel algorithm was compared and analyzed. The simulation test results show that the GPU-based batch sparse LU decomposition parallel algorithm can obtain an acceleration effect of 25~50 times on average.

Keywords: power flow calculation; batch-processing; LU decomposition; parallel algorithm

(编辑 钱悦)