基于GPU加速的大电网*N*-1故障扫描批量计算方法

张宸赓[1],许　寅[1],陈　颖[2],苏大威[3],李　一[3],刘思言[4]
(1. 北京交通大学 电气工程学院,北京 100044;2. 清华大学 电机工程与应用电子技术系,北京 100084;
3. 国网江苏省电力有限公司,江苏 南京 210024;4. 全球能源互联网研究院有限公司,北京 102209)

摘要:随着电网规模的不断扩大,从各种可能的设备开断情况中筛选出严重故障集成为*N*-1安全校验的重要耗时部分。为了加速大电网*N*-1安全校验的故障筛选,提出了一种基于中央处理器-图形处理器(CPU-GPU)异构计算框架的实时*N*-1故障扫描批量计算方法。考虑到不同工况下的计算存在粗粒度并行性,进一步挖掘计算中的细粒度并行性是提高计算效率的有效途径。提出了同时考虑断线故障和发电机开断故障这2种预想事故下的细粒度并行计算方法,并设计了关键计算步骤的核函数。增加考虑了网络拓扑中的移相器,使得计算精度更高,通过与IEEE标准算例和欧洲真实电网算例对比,验证了各工况下批量计算方法的正确性,并取得了显著的加速效果。

关键词:电力系统;静态安全分析;GPU;*N*-1故障扫描;CPU-GPU异构计算框架
中图分类号:TM 715　　**文献标志码**:A　　DOI:10.16081/j.epae.202007030

0　引言

随着电网规模的不断增大和结构的日益复杂,亟需建立更加可靠的电力运行监视和控制系统,保障电网的经济安全运行,同时也对电网实时准确的调度控制能力提出了更高的要求。*N*-1安全校验可用于检验电网中某一元件强迫退出运行后系统的运行状态,进而准确地指导调度采取措施。为避免发生元件过负荷或者负荷母线过电压所造成的设备损坏或大面积停电等严重事故,实时进行系统*N*-1安全校验尤为重要。但由于在线网络分析能力的不足,目前*N*-1潮流求解多采用离线方式,无法满足调度决策的在线需求。因此,提高在线*N*-1安全校验的计算能力对电力系统实时在线运行具有重要作用[1]。

N-1安全校验的实质是电力系统运行的稳态分析问题,即潮流问题[2]。在线或实时分析需要在短时间内完成计算。为加快大电网*N*-1安全校验的潮流计算速度,文献[3-4]采用补偿法,在发生断线故障的支路两端节点处引入补偿功率或补偿电流模拟支路开断,因而不必重新形成因子表。文献[5]提出了一种基于网络分块的新型迭代算法,通过降低网络维数加快计算速度。上述方法虽然取得了一定的加速效果,但是对于大电网而言,仍然无法在短时间内完成所有预想事故的分析,因而难以满足实时的全局分析要求。

为了进一步加快计算速度,常用的计算方法是采用多核中央处理器(CPU)并行计算技术,但是受内存带宽的制约,传统多核CPU并行计算效率不能随着核数增加而线性增加,解决内存带宽饱和问题需要部署具有独立物理内存的计算节点,这将大幅增加能耗和费用,因而不适用于大规模系统[6]。

与多核CPU相比,图形处理器(GPU)因其在浮点运算和内存带宽的优越性能而具有更为强大的并行计算能力,已经被成功运用到包括电力系统领域在内的众多科学计算领域[7]。针对不同的计算需求,相关学者提出了不同的GPU加速算法。针对潮流计算,不同的计算方法如高斯-赛德尔迭代法[8]、牛顿-拉夫逊法[9]以及快速分解法[10]等均在GPU内被加速实现。针对潮流计算中稀疏线性方程组的求解,文献[11-12]分别提出了基于GPU加速的求解稀疏线性方程组的不同方法,如LU分解、共轭梯度法。针对*N*-1安全校验,文献[13]实现了GPU上的批量LU分解并行求解器,提高批量潮流求解效率,但是关于*N*-1安全校验中批量场景生成等其他环节没有研究。文献[14]设计了批量求解稀疏线性方程组的QR分解算法和雅可比矩阵生成的计算方法,取得了不错的加速效果,但是由于受内存等方面的限制,可处理的故障规模不大。文献[15]设计了一种双层并行算法以进行大规模系统的*N*-1安全校验,在GPU中取得了近10倍的加速效果。文献[16]将*N*-1安全校验问题拼接成一个较大规模的潮流问题,并采用迭代计算法,虽然提高了整体计算效率但是迭代精度不高。上述基于交流潮流算法的*N*-1安全校验虽然取得了一定的加速效果,但是针对大规模系统,存在很严重的显存问题,无法进行全网的在线*N*-1安全校验。

实际上只有少数的紧急故障会危及系统安全,

收稿日期:2020-01-17;**修回日期**:2020-05-27
基金项目:国家电网公司总部科技项目(5455HJ180004)
Project supported by the Science and Technology Project of SGCC(5455HJ180004)

为了进一步提高计算速度，更为实用的方法是先采用直流潮流法快速进行N-1故障扫描，筛选出过负荷可能性较大的线路，形成紧急故障集从而缩小故障集数量，再针对故障集进行详细潮流分析[17]。因此，加速基于直流潮流算法的N-1故障扫描进而加速严重故障集筛选对于大电网的N-1安全校验具有重要意义。文献[18-19]针对直流潮流算法N-1故障扫描进行了GPU加速，但是上述方法还存在以下不足：未能充分挖掘断线故障和断发电机故障之间的细粒度并行特点；未考虑移相器，当网络拓扑中存在移相器时，计算结果将产生误差而不再适用；需要在CPU中完成求逆运算的准备工作，计算量大幅增加，同时，求逆过程将稀疏矩阵变为大规模实矩阵，增加了内存的占用，延长了通信的时间，当计算规模超出内存限制时，不具有良好的可拓展性。

综合以上问题，本文的主要工作内容如下：

(1)针对CPU和GPU组成的异构并行计算环境，设计了基于直流潮流算法的大型电网N-1校验故障扫描的计算架构，同时匹配该计算架构，设计了生成严格N-1故障扫描批量场景算例的核函数，提出了将所有断线故障和发电机开断故障这2种不同场景同时并行处理的细粒度并行算法，提高了计算效率；

(2)针对支路开断批量场景生成，本算法在拓扑中增设移相器，使计算结果精度更高；

(3)针对N-1安全校验不改变矩阵稀疏性的特点，采用批量LU分解求解大型稀疏线性方程组，从而避免求逆过程带来的计算量较大问题，并且，由于每个线性方程组之间相互独立，当系统规模继续增大时，更加容易进行数据切分，从而扩展到多GPU运行，与此同时，该算法可以根据变化后矩阵的奇异性筛选出断线故障造成的系统解列。

1 N-1故障扫描计算方法及框架

1.1 直流潮流模型

在N-1快速故障筛选中，相较于较高的计算精度，计算迅速、内存量少通常更为重要。并且筛选的重点集中于过负荷问题而非节点电压越界问题，即网络需要满足安全输送电力的要求。直流潮流模型是线性模型，可以快速分析断线故障或发电机开断故障时的线路过载情况，内存占用少，因此适用于筛选前的故障扫描[20]。

根据实际电网的特点，对直流潮流模型做出以下假设：①认为支路电抗远大于支路电阻，即忽略支路电阻；②认为支路两端节点电压的相角差很小；③忽略支路对地电纳；④各节点电压模值相等且均等于1 p.u.。

考虑移相器的直流潮流模型的功率平衡方程[21]为：

$$\boldsymbol{P}_{\mathrm{dc}}=\boldsymbol{B}_{\mathrm{dc}}\boldsymbol{\theta} \tag{1}$$

其中，$\boldsymbol{B}_{\mathrm{dc}}$为支路导纳矩阵；$\boldsymbol{\theta}$为电压相角列向量，且认为平衡节点相角为0；$\boldsymbol{P}_{\mathrm{dc}}$为等效功率注入列向量。上述变量中均不包含平衡节点所对应的变量。$\boldsymbol{P}_{\mathrm{dc}}$可以表示为：

$$\boldsymbol{P}_{\mathrm{dc}}=\boldsymbol{P}_{\mathrm{G}}-\boldsymbol{P}_{\mathrm{D}}-\boldsymbol{G}_{\mathrm{sh}}-\boldsymbol{P}_{\mathrm{shift}} \tag{2}$$

设i为任意节点，即：

$$\begin{cases}P^{i}_{\mathrm{dc}}=P^{i}_{\mathrm{G}}-P^{i}_{\mathrm{D}}-G^{i}_{\mathrm{sh}}-P^{i}_{\mathrm{shift}}\\P^{i}_{\mathrm{shift}}=-\theta_{\mathrm{shift}(ij)}/(k_{ij}x_{ij})\end{cases} \tag{3}$$

其中，$\boldsymbol{P}_{\mathrm{G}}$和$\boldsymbol{P}_{\mathrm{D}}$分别为发电机出力和负荷的列向量；$\boldsymbol{G}_{\mathrm{sh}}$为并联电导列向量；$\boldsymbol{P}_{\mathrm{shift}}$为移相器等效注入功率列向量；$P^{i}_{\mathrm{shift}}$为$\boldsymbol{P}_{\mathrm{shift}}$中节点$i$所对应的元素；$P^{i}_{\mathrm{dc}}$为$\boldsymbol{P}_{\mathrm{dc}}$中节点$i$所对应的元素；$P^{i}_{\mathrm{G}}$和$P^{i}_{\mathrm{D}}$分别为节点$i$的发电机出力和负荷；$G^{i}_{\mathrm{sh}}$为与节点$i$并联的电导；$\theta_{\mathrm{shift}(ij)}$和$x_{ij}$分别为支路$ij$的移相角和电抗；$k_{ij}$为支路$ij$的变压器变比，当支路为输电线路时$k_{ij}=1$。

根据求解的电压相角向量，求解支路的有功功率可得：

$$P_{ij}=\frac{\theta_i-\theta_j-\theta_{\mathrm{shift}(ij)}}{k_{ij}x_{ij}} \tag{4}$$

其中，P_{ij}为支路ij的有功功率；θ_i和θ_j分别为向量$\boldsymbol{\theta}$中节点i和节点j所对应的元素。

综上所述，本文采用的故障扫描步骤如下。

(1)断线故障：①根据支路开断信息修改矩阵$\boldsymbol{B}_{\mathrm{dc}}$的对应元素；②若开断支路包含移相器，则修改$\boldsymbol{P}_{\mathrm{shift}}$；③根据式(1)，求得电压相角向量$\boldsymbol{\theta}$；④根据式(4)，求得该工况下各线路的有功功率。

(2)发电机开断故障：①根据发电机开断信息修改向量$\boldsymbol{P}_{\mathrm{G}}$的对应元素；②根据式(1)，求得电压相角向量$\boldsymbol{\theta}$；③根据式(4)，求得该工况下各线路的有功功率。

1.2 算法体系架构

目前CUDA异构平台将GPU中流多处理器(SM)等硬件模块抽象为易于开发的编程模型，为CPU-GPU异构运算提供了良好条件，开发者可以轻松调用GPU资源进行并行计算。在该模型下，GPU程序被组织为计算核函数供CPU调用，核函数中包含大量线程，这些线程被组织为线程块，线程块又被组织为线程网络，可供随时查找利用[22]。

根据CPU-GPU的软硬件架构和编程模型，结合网络分析应用的特点，所提出的N-1故障扫描的批量计算框架如图1所示。考虑到支路开断和发电机开断这2种工况之间相互独立，存在粗粒度并行性，针对每种工况的计算步骤，可进一步挖掘其计算方法的并行性。因而，可根据数据之间的关系，采用细粒度并行求解，创建海量并发线程批量生成各工况下

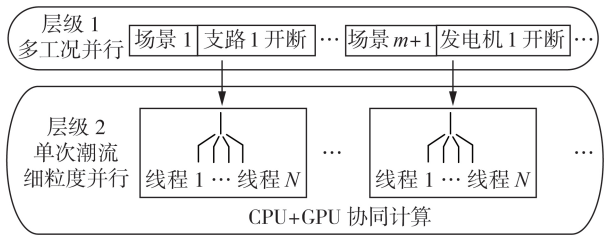

图 1　双层并行算法架构

Fig.1　Architecture of double-layer parallel algorithm

导纳矩阵、等效功率注入列向量、支路有功功率等。

2　N-1故障扫描并行求解方法

2.1　计算流程

本节提出了一种基于GPU加速的批量N-1故障扫描计算流程，如附录A中图A1所示。设计思路和细节如下：①CPU主要负责控制整个流程和如下2个方面具体的计算，一是准备基态数据和支路、发电机开断信息，二是对GPU的计算结果进行处理；②GPU负责批量生成$\boldsymbol{B}_{\mathrm{dc}}$、$\boldsymbol{P}_{\mathrm{dc}}$和线路功率等浮点运算，不论CPU还是GPU，由于求逆的计算量巨大，应尽量避免求逆运算，故采用批量LU分解和前代回代求解稀疏线性方程组。

2.2　基于数据集的细粒度计算方法

根据2.1节所提出的计算流程，进一步挖掘并行性，将断线故障和发电机开断故障的关键步骤相结合，设计细粒度并行优化计算方法，其细粒度并行优化算法见附录A中图A2。

设m为支路总数，s为发电机总数，n为节点总数（不包含平衡节点）。当发生发电机开断故障时，并不会改变导纳矩阵的元素，只需要进行一次LU分解，然后采用批量前代回代求解。但是为了使得算法之间具有更好的并行性，在发电机开断故障场景下也同样复制稀疏形式的导纳矩阵元素，无需创建线程对其进行修改，然后共同采用批量LU分解求解线性方程组。因此，本文将N-1故障扫描过程细分为5个层级，相同层级之间的计算无关联性，可以并行求解。第一层级为复制矩阵$\boldsymbol{B}_{\mathrm{dc}}$，创建$m$个线程批量修改前$m$个矩阵$\boldsymbol{B}_{\mathrm{dc}}$；第二层级为复制等效功率注入列向量$\boldsymbol{P}_{\mathrm{dc}}$并批量修改；第三层级采用批量LU分解求解稀疏线性方程组；第四层级为在所得解的平衡节点位置处加入0元素；第五层级为批量生成线路功率，将生成的各工况下线路潮流存储在一维数组中。后文将对各个层级的核函数做详细介绍。

2.3　批量生成导纳矩阵

由于$\boldsymbol{B}_{\mathrm{dc}}$为大型稀疏矩阵，本文采用压缩稀疏行（CSR）存储形式，$\boldsymbol{A}_{\mathrm{x}}$、$\boldsymbol{A}_{\mathrm{i}}$、$\boldsymbol{A}_{\mathrm{p}}$分别为数值、列号和行偏移的向量。断线故障只会影响矩阵$\boldsymbol{B}_{\mathrm{dc}}$的4个元素，不会影响矩阵的稀疏性，因此只需要批量生成$\boldsymbol{A}_{\mathrm{x}}$即可。用函数cudaMemcpy将$\boldsymbol{A}_{\mathrm{x}}$在显存中复制$m+s$份，创建$m$个线程并对前$m$个$\boldsymbol{A}_{\mathrm{x}}$进行修改，后$s$个$\boldsymbol{A}_{\mathrm{x}}$保持不变。为避免多次访问全局内存造成的延时，同时便于故障集过大时进行数据划分，不必将细粒度划分得过细，令1个线程处理1种线路开断，即1个线程负责修改导纳矩阵中的4个元素。

采用CUDA异构平台调用GPU的内核函数Kernel_1<N_{blocks}, N_{threads}>，其中第一个参数N_{blocks}为网格维度，表示启动线程块的数量；第二个参数N_{threads}为线程块维度，表示每个线程块中执行的线程数量。设$N_{\mathrm{threads}}=256$，由式（5）根据向上取整原则确定启动线程块的数量，文中所有核函数的线程配置方式均采用上述方式进行。

$$N_{\mathrm{blocks}}=\left\lfloor\frac{m+N_{\mathrm{threads}}-1}{N_{\mathrm{threads}}}\right\rfloor \tag{5}$$

以节点i对应的矩阵$\boldsymbol{B}_{\mathrm{dc}}$中主对角线元素的修改为例，首先利用每个线程编号索引支路编号，然后确定该支路的两端节点编号i和j。由于各节点已经按照“PQ节点–PV节点–平衡节点”的排列顺序在CPU中进行节点重排，当节点编号小于平衡节点编号时，需要根据节点编号、$\boldsymbol{A}_{\mathrm{i}}$和$\boldsymbol{A}_{\mathrm{p}}$索引需要修改的$\boldsymbol{A}_{\mathrm{x}}$的具体位置，再对其进行修改。该修改过程中的核函数伪代码见附录A中表A1。矩阵$\boldsymbol{B}_{\mathrm{dc}}$中其余3个元素的修改同理，不再赘述。

2.4　等效功率注入列向量并行计算

以下2种情形将导致等效功率注入列向量$\boldsymbol{P}_{\mathrm{dc}}$发生改变：①断线支路中含有移相器；②发电机开断故障。创建$m+s$个线程，前m个线程处理断线故障，后s个线程处理发电机开断故障。针对情形①，假设支路ij开断，根据上文分析$\boldsymbol{P}_{\mathrm{dc}}$中节点$i$对应元素和节点$j$对应元素将发生改变，节点$i$处修改变量表示为$\Delta P_{\mathrm{dc}}^{i}=-\theta_{\mathrm{shift}(ij)}/(k_{ij}x_{ij})$，节点$j$处修改变量与节点$i$处修改变量互为相反数，即$\Delta P_{\mathrm{dc}}^{j}=-\Delta P_{\mathrm{dc}}^{i}$，若支路中不含移相器，则$\theta_{\mathrm{shift}(ij)}=0$；针对情形②，假设节点$a$处的发电机开断，节点$a$处的注入功率将发生改变，进而改变$P_{\mathrm{dc}}^{a}$。图2为潮流方程右端项的并行计算过程。图中，$i(1)$和$j(1)$分别为支路1对应的流出和流入节点；$a(1)$为发电机1对应的节点编号；$(P_{\mathrm{dc}}^{i(1)})_1$为场景1中的向量$\boldsymbol{P}_{\mathrm{dc}}$中节点$i(1)$对应的元素，其余

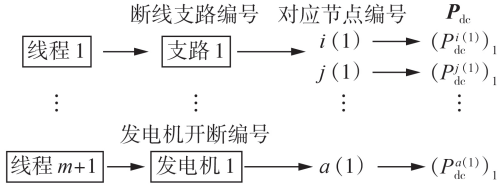

图 2　等效功率注入列向量计算核函数设计

Fig.2　Kernel function design for calculating equivalent power injection column vector

同理。

当线程编号索引小于m时，由线程编号索引到断线支路编号，根据对应的支路编号将等效功率注入列向量的第i个元素和第j个元素进行修改；当线程编号索引为m到$m+s$时，由发电机编号索引对应的节点编号，对节点注入功率进行修改。上述设计过程中不同的计算指令将分配到不同的线程束中，GPU并行处理此类运算，且并行度很高。

2.5 批量求解线性方程组

由于各场景下导纳矩阵$\boldsymbol{B}_{\mathrm{dc}}$的稀疏性相同，可以采用批量LU分解并行求解稀疏线性方程组。矩阵重排可以减少非零元素的注入，增加数值稳定性，且只需要进行1次，可以采用Colamd算法进行矩阵重排[23]。求解稀疏线性方程组有很多高性能计算库，如适用于CPU的NICSLU[24]和PARDISO[25]计算库，其分别采用Left-looking算法和超节点算法求解；除此以外还有针对GPU的cuSolver以及cuSparse函数库[22]，效率较高且接口友好。cuSolver函数库中的cuSolverRF函数库可在同一稀疏模式下给定新系数，同时快速重新分解，加速线性方程组的求解过程。本文求解时首先在CPU中进行一次矩阵重排和完整的LU分解（包括符号分解和数值分解），将其结果传输到GPU，再调用cuSolverRF函数库批量进行LU分解，快速并行求解线性方程组。后文将对比分析该算法与CPU的NICSLU计算库的算法性能。

2.6 修改电压相角向量

直流潮流模型平衡节点的相角为0，因此，对于2.5节中求得的电压相角向量，需要在平衡节点的位置加零元素，其核函数设计思路如图3所示。

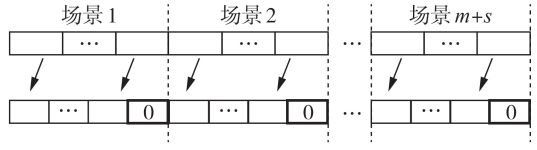

图3 批量修改电压相角向量核函数设计

Fig.3 Kernel function design for modifying voltage phase angle vector in batch

伪代码见附录A中表A2，令$\boldsymbol{x}$为2.5节求得的电压相角向量数组，新数组用$\boldsymbol{x}_1$表示。创建包含$(m+s)\times(n+1)$个元素的一维数组，采用cudaMemset函数将其全部置0；然后创建$(m+s)\times n$个线程，每个线程处理1个元素。CUDA异构平台采用单指令多线程架构管理和执行线程，每32个线程为1个线程束，线程束中所有线程同时执行相同的指令，由于线程并不能根据批量处理规模完美分配，此处对线程编号设置判断条件如下：编号小于$(m+s)\times n$的线程执行任务，多余的线程被过滤。此时，同1个线程束中依然执行相同的指令，不会导致线程束的分支，保持很高的并行性。同时上述算法中内存连续读取，保证同一线程束下内存的合并访问，使算法具有高效性。

2.7 批量求解支路功率

每一种工况下，都需要根据式(4)求解各支路的有功功率，且每条支路有功功率的计算相互独立。同样，不同工况之间无关联性，因此也可以并行求解。如图4所示，设计了并行求解各工况下支路有功功率的核函数。针对m条支路开断工况和s台发电机开断工况，同时创建$(m+s)\times m$个线程，前m个线程处理第1条支路开断的情形，线程1—m依次根据各支路信息索引支路两端的节点编号以及该支路的变压器变比、电抗、移相角等信息，再根据节点标号索引数组$\boldsymbol{x}_1$的对应元素，求得相应支路的有功功率。后面的m个线程重新遍历各条支路，依此类推。最终将支路有功功率计算结果按顺序存储于一维数组中，用P_i表示该数组的第i个元素。

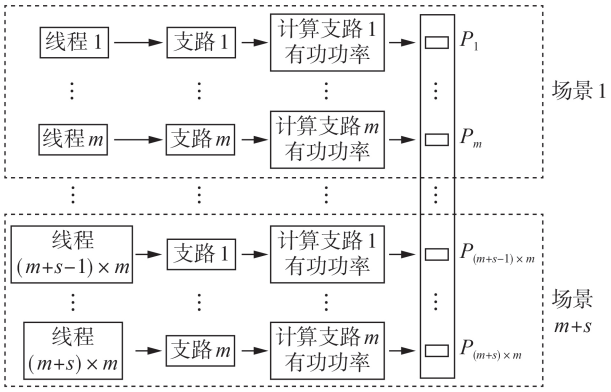

图4 求解支路有功功率计算核函数设计

Fig.4 Kernel function design for calculating branch active power

3 算例测试

3.1 测试平台

本文所进行的算例效率测试均在高性能计算服务器上完成，相应的软硬件平台见附录B中表B1。GPU采用NVIDIA公司出产的Tesla P100，该GPU具有3 584个CUDA核心，具有超强的浮点计算能力。CPU型号是Intel志强系列E3-1230 v5，运行主频为3.4 GHz。

3.2 测试算例

本文对比分析了5个不同规模的算例，Case 118、Case 300均为IEEE标准算例，Case 1 354、Case 2 869、Case 9 241则取自Matpower[21]，其中Case 9 241来自欧洲电网的真实算例。

3.3 正确性测试

随机选取1条支路开断和1台发电机开断，对各个测试算例进行支路潮流计算。将所得到的结果与Matpower中直流潮流计算结果进行对比，验证了本

文所提计算方法的正确性。其中，2种计算结果的最大误差如表1所示。

表1　正确性测试

Table 1　Correctness test

测试算例	最大潮流误差	
	支路开断	发电机开断
Case 118	$6.011\,2\times10^{-12}$	$5.099\,9\times10^{-12}$
Case 300	$1.220\,0\times10^{-11}$	$1.205\,1\times10^{-11}$
Case 1 354	$3.300\,0\times10^{-10}$	$3.250\,0\times10^{-10}$
Case 2 869	$3.211\,0\times10^{-9}$	$3.263\,0\times10^{-9}$
Case 9 241	$3.442\,0\times10^{-9}$	$3.550\,0\times10^{-9}$

3.4　效率测试

3.4.1　批量生成导纳矩阵计算部分

基于GPU和CPU批量生成导纳矩阵的计算时间和加速比变化趋势如图5所示。由图5可知，当算例规模较小时，采用CPU计算的效率更高，但随着规模的增大，故障集的增多，采用GPU计算相较于CPU可取得较大的加速效果，例如Case 9 241中加速比可达25。

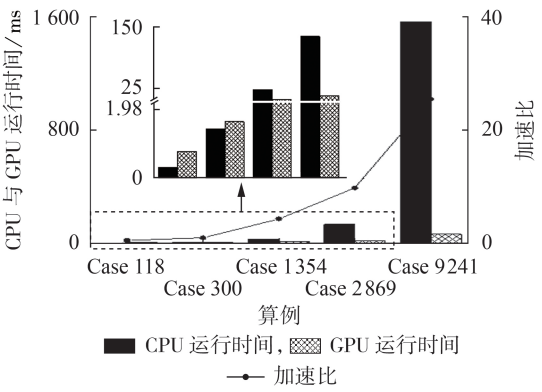

图5　批量生成导纳矩阵的计算效率及其放大图

Fig.5　Calculated results and enlarged drawing of admittance matrix generated in batch

3.4.2　等效功率注入列向量并行计算部分

批量生成节点等效功率注入列向量耗时在整体计算耗时中所占比重不大，与批量生成导纳矩阵类似，在规模较小的系统中CPU的计算效率更高，但随着节点数的增加，GPU逐渐展现出优越性。

图6为等效功率注入列向量并行计算效率测试结果。由图6中加速比变化趋势可得，针对Case 9 241，GPU并行算法相较于CPU算法可取得7倍的加速效果，从加速趋势来看，对于更大的系统，加速比还会继续增大。

3.4.3　批量求解线性方程组

批量求解稀疏线性方程组通常为潮流计算中最耗时的部分。如前所述，分别基于CPU的NICSLU计算库和GPU的cuSolver计算库完成批量线性方程组的求解，其计算效率如附录B中图B1所示。由图

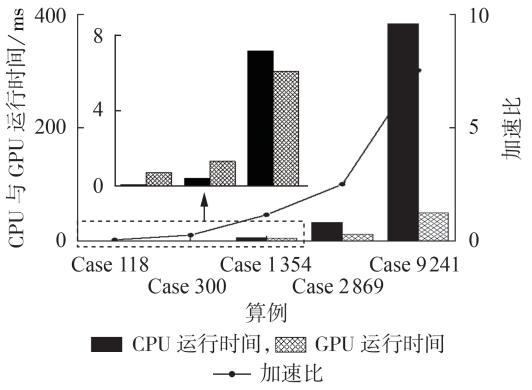

图6　等效功率注入列向量并行计算效率及其放大图

Fig.6　Parallel computation efficiency and enlarged drawing of equivalent power injection column vector

B1可知，批量求解线性方程组时，随着系统规模的增大，基于GPU的算法更加高效。

批量求解线性方程组的计算时间受很多因素的影响，如计算矩阵的维数、稀疏矩阵的非零元素、批量处理规模等，其详细测试结果见附录B中表B2。

3.4.4　修改电压相角向量的计算

修改电压相角向量的计算实际上为一对一的映射问题，非常适合在GPU中处理。不同的算例中基于CPU和GPU对电压相角向量进行批量修改的计算耗时对比和加速比变化趋势如附录B中图B2所示。结果表明，在各种规模下GPU算法都表现出更优越的性能。随着算例规模的增大，加速比逐渐趋于稳定。

利用Windows下的Nsight工具对GPU程序进行性能分析，Windows下的显卡型号为GeForce GT 710，每个流多处理器的占用率见附录B中图B3。1个流多处理器上实际运行的Warp数量占可运行线性束数量的百分比可达100%，充分发挥了GPU的特性。

3.4.5　批量求解支路功率

求解各工况下不同支路的有功功率存在双层的并行关系，各部分计算相互独立，不会造成线性束分支，显然更适合在GPU中并行计算，计算效率如附录B中图B4所示。结果显示，随着批量求解处理规模的增大，加速比曲线稳步上升，最终趋于平缓，加速效果明显。

3.4.6　整体计算效率

经过前文分析，基于GPU的$N-1$故障扫描在每一步的计算速度上都取得了很大的提升。将本文算法与CPU串行程序对比，测试结果如附录B中图B5所示。随着节点规模的增大和故障集的增多，加速比不断增加，针对Case 9 241，整体计算效率约为CPU算法计算效率的20倍，在1 s内可以完成所有支路开断和发电机开断场景下的潮流计算，单一故障扫描平均时间小于0.1 ms。

以Case 9 241为例，该系统包含16 049条支路和1 445台发电机，批量处理规模为17 494，GPU各部分计算时间占比见附录B中图B6。矩阵的重新LU分解和前代回代占据了大量的计算时间，经优化设计，该算法中每个核函数的时间占比都较小。

在Case 9 241中将本文算法与传统$N-1$方法如牛顿-拉夫逊法和快速分解法进行耗时对比，如图7所示。上述方法均在GPU中进行，逐渐增加批量处理的规模，测试各方法的耗时。测试结果表明，随着批量处理规模的增加，基于交流潮流的计算量越来越大，时间也成倍增加，当批量处理规模达到2 048时，牛顿-拉夫逊法出现显存问题，当处理规模达到3 072时，快速分解法出现显存问题，无法对全网进行$N-1$扫描，而本文算法不论从计算时间还是内存占用上都具有非常明显的优势，更加充分说明了采用基于直流潮流模型进行$N-1$故障筛选的必要性。

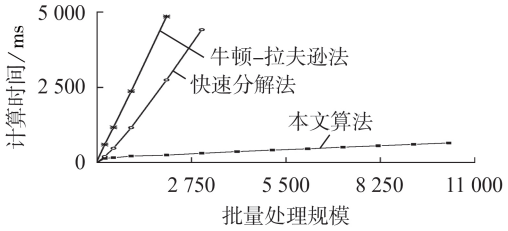

图7 本文算法与传统算法计算用时对比

Fig.7 Comparison of time consuming among proposed and traditional algorithms

4 结论与展望

为了提高大电网的$N-1$安全校验故障筛选效率，本文提出了适用于CPU-GPU协同异构的实时$N-1$故障扫描批量计算方法，设计了将断线故障和发电机开断故障并行处理的细粒度计算方法，在保证所提方法正确的同时，取得了与CPU串行程序相比更显著的加速效果。所提方法不仅满足实时$N-1$故障筛选应用的需求，还可以检查故障后网络的连通性（网络的连通性指断线故障引起网络解列所造成的方程组无解）。在计算速度要求很高且精度要求不高的场景下，该方法还可应用于$N-1$安全校验。在后续的研究工作中，需考虑在内存问题的基础上将数据进行划分，从而实现单CPU多GPU架构对更大系统规模的处理，进一步提升计算效率。

附录见本刊网络版（http://www.epae.cn）。

参考文献：

[1] 傅旭. 电力系统静态安全混合控制方法[J]. 电力自动化设备，2017，37(1)：124-130.
FU Xu. Hybrid control of power system static security[J]. Electric Power Automation Equipment，2017，37(1)：124-130.

[2] 王锡凡. 现代电力系统分析[M]. 北京：科学出版社，2003：90-104.

[3] 阳育德，陶琢，刘辉，等. 电力系统静态安全最优潮流并行计算方法[J]. 电力自动化设备，2019，39(1)：99-105.
YANG Yude，TAO Zhuo，LIU Hui，et al. Parallel computation methods for static security-constrained optimal power flow of power system[J]. Electric Power Automation Equipment，2019，39(1)：99-105.

[4] 阳育德，冯彦维，韦化. 基于补偿法的预防性静态安全控制[J]. 电力自动化设备，2015，35(12)：47-54.
YANG Yude，FENG Yanwei，WEI Hua. Preventive static security control based on compensation method[J]. Electric Power Automation Equipment，2015，35(12)：47-54.

[5] 王方雨，刘文颖，田浩，等. 基于网络分块算法的静态安全快速计算方法[J]. 电力自动化设备，2017，37(4)：203-209.
WANG Fangyu，LIU Wenying，TIAN Hao，et al. Fast static security calculation based on network partition algorithm[J]. Electric Power Automation Equipment，2017，37(4)：203-209.

[6] 陈国良. 并行计算：结构算法编程[M]. 北京：高等教育出版社，2003：13-17.

[7] GREEN R C，WANG L F，ALAM M. High performance computing for electric power systems：applications and trends[C]// 2011 IEEE Power and Energy Society General Meeting. San Diego，CA，USA：IEEE，2011：1-8.

[8] ROBERGE V，TARBOUCHI M，OKOU F. Parallel power flow on graphics processing units for concurrent evaluation of many networks[J]. IEEE Transactions on Smart Grid，2017，8(4)：1639-1648.

[9] JIANG H，CHEN D Y，LI Y L，et al. A fine-grained parallel power flow method for large scale grid based on lightweight GPU threads[C]//2016 IEEE 22nd International Conference on Parallel And Distributed Systems(ICPADS). Wuhan，China：IEEE，2016：785-790.

[10] LI X，LI F X，YUAN H Y，et al. GPU-based fast decoupled power flow with preconditioned iterative solver and inexact Newton method[C]//2017 IEEE Power & Energy Society General Meeting. Chicago，USA：IEEE，2017：2695-2703.

[11] JALILI-MARANDI V，ZHOU Z Y，DINAVAHI V. Large-scale transient stability simulation of electrical power systems on parallel GPUs[J]. IEEE Transactions on Parallel and Distributed Systems，2012，23(7)：1255-1266.

[12] 张宁宇，高山，赵欣. 一种求解机组组合问题的内点半定规划GPU并行算法[J]. 电力自动化设备，2013，33(7)：126-131，138.
ZHANG Ningyu，GAO Shan，ZHAO Xin. GPU parallel algorithm of interior point SDP for unit commitment[J]. Electric Power Automation Equipment，2013，33(7)：126-131，138.

[13] ZHOU G，BO R，CHIEN L，et al. GPU-based batch LU-factorization solver for concurrent analysis of massive power flows[J]. IEEE Transactions on Power Systems，2017，32(6)：4975-4977.

[14] ZHOU G，FENG Y J，BO R，et al. GPU-accelerated batch-ACPF solution for N-1 static security analysis[J]. IEEE Transactions on Smart Grid，2017，8(3)：1406-1416.

[15] CHEN D Y，JIANG H，LI Y L，et al. A two-layered parallel static security assessment for large-scale grids based on GPU[J]. IEEE Transactions on Smart Grid，2017，8(3)：1396-1405.

[16] 唐坤杰，董树锋，宋永华. 一种GPU-CPU异构运算框架加速的实时N-1交流潮流计算方法[J]. 中国电机工程学报，2018，38(15)：4329-4338，4633.
TANG Kunjie，DONG Shufeng，SONG Yonghua. A real-time N-1 AC power flow calculation method based on GPU-CPU

heterogeneous computing framework[J]. Proceedings of the CSEE, 2018, 38(15): 4329-4338, 4633.
[17] BALU N, BERTRAM T, BOSE A, et al. On-line power system security analysis[J]. Proceedings of the IEEE, 1992, 80(2): 262-282.
[18] 温柏坚，谢恩彦，刘明波. 匹配GPU编程架构的直流故障筛选算法设计[J]. 电力自动化设备，2017，37(5)：217-223.
WEN Bojian, XIE Enyan, LIU Mingbo. Design of DC contingency screening algorithm matching GPU programming architecture[J]. Electric Power Automation Equipment, 2017, 37(5): 217-223.
[19] ZHOU G, ZHANG X, LANG Y S, et al. A novel GPU-accelerated strategy for contingency screening of static security analysis[J]. International Journal of Electrical Power & Energy Systems, 2016, 83: 33-39.
[20] 李峰，李虎成，於益军，等. 基于并行计算和数据复用的快速静态安全校核技术[J]. 电力系统自动化，2013，37(14)：75-80.
LI Feng, LI Hucheng, YU Yijun, et al. Fast computing technologies for static security checking based on parallel computation and data reuse[J]. Automation of Electric Power Systems, 2013, 37(14): 75-80.
[21] ZIMMERMAN R D, MURILLO-SANCHEZ C E, THOMAS R J. MATPOWER: steady-state operations, planning, and analysis tools for power systems research and education[J]. IEEE Transactions on Power Systems, 2011, 26(1): 12-19.
[22] NVIDIA. CUDA toolkit documentation v10.0[EB/OL]. (2018-10-30)[2020-01-17]. http://docs.nvidia.com/cuda/archive/10.0.
[23] DAVIS T A, GILBERT J, LARIMORE S I, et al. A column approximate minimum degree ordering algorithm[J]. ACM Transactions on Mathematical Software, 2004, 30(3): 353-376.
[24] CHEN X M, WANG Y, YANG H Z. NICSLU: an adaptive sparse matrix solver for parallel circuit simulation[J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2013, 32(2): 261-274.
[25] SCHENK O, GARTNER K. Solving unsymmetric sparse systems of linear equations with PARDISO[J]. Future Generation Computer Systems, 2003, 20(3): 475-487.

作者简介：

张宸赓

张宸赓（1995—），男，山西临汾人，硕士研究生，主要研究方向为电力系统高性能计算（**E-mail**：chengengzhang@bjtu.edu.cn）；

许　寅（1986—），男，北京人，教授，博士研究生导师，主要研究方向为配电系统故障恢复与韧性、电力系统电磁暂态仿真与高性能计算（**E-mail**：xuyin@bjtu.edu.cn）；

陈　颖（1979—），男，北京人，副教授，博士，主要研究方向为电力系统动态仿真、并行和分布式计算（**E-mail**：chen_ying@tsinghua.edu.cn）。

（**编辑**　王欣竹）

Batch computing method for *N*-1 fault scanning of large power grid based on GPU acceleration

ZHANG Chengeng[1], XU Yin[1], CHEN Ying[2], SU Dawei[3], LI Yi[3], LIU Siyan[4]
(1. School of Electrical Engineering, Beijing Jiaotong University, Beijing 100044, China;
2. Department of Electrical Engineering, Tsinghua University, Beijing 100084, China;
3. State Grid Jiangsu Electric Power Co., Ltd., Nanjing 210024, China;
4. Global Energy Interconnection Research Institute, Beijing 102209, China)

Abstract: With the increasing scale of power grid, the selection of serious contingency from various possible equipment interruptions are becoming an important time-consuming part of *N*-1 safety verification. In order to accelerate the speed of fault selection of *N*-1 safety verification of large power grid, a real-time *N*-1 fault scanning batch calculation method based on CPU-GPU (Central Processing Unit-Graphics Processing Unit) heterogeneous computing framework is proposed. Considering that coarse-grained parallelism exists in computation under different working conditions, it is an effective way to improve computational efficiency to further exploit fine-grained parallelism in computation. A fine-grained parallel calculation method considering both branch break fault and generator break fault is proposed, and the kernel function of key calculation steps is designed. What's more, the phase shifter is considered in the network topology to improve the calculation accuracy. Compared with the IEEE standard cases and European actual grid cases, the correctness of the batch calculation method under various working conditions is verified, and a remarkable acceleration effect is achieved.

Key words: electric power systems; static security analysis; GPU; *N*-1 fault scanning; CPU-GPU heterogeneous computing framework

附录 A

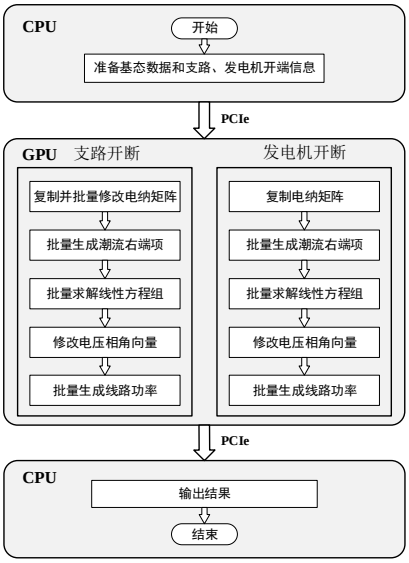

图 A1　GPU 加速算法流程图

Fig.A1　Flowchart of GPU acceleration algorithm

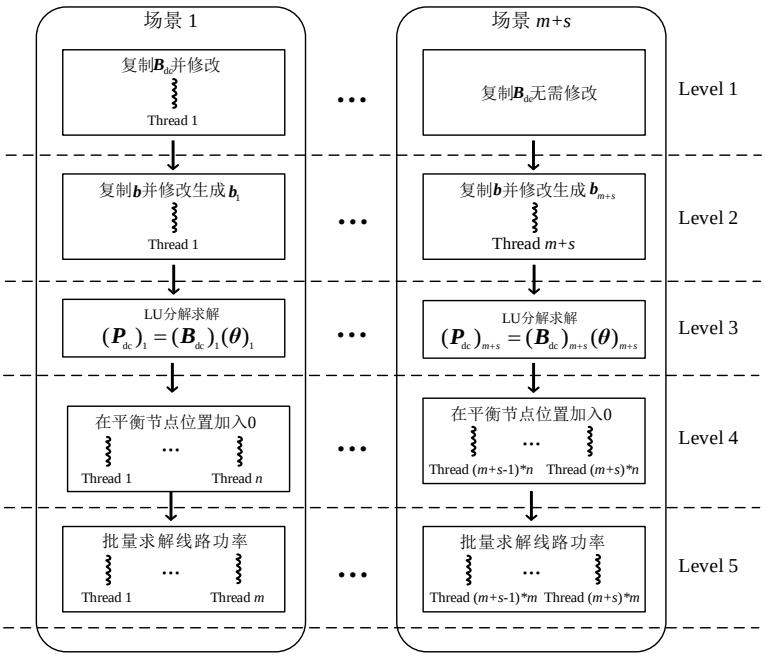

图 A2　细粒度并行优化算法

Fig.A2　Fine-gained optimization of parallel algorithm

表 A1 批量修改导纳矩阵核函数伪代码

Table A1 Kernel function pseudo-code for batch modification of admittance matric

Algorithm1 Kernel_1

→enter
tid←threadID
b←tid //索引到断线支路编号
if $b<m$
　i , j ←断线支路对应的节点编号
　if i <slack node ID
　index← A_p 、 A_i
　$(A_x^{index})_b=(A_x^{index})_b-1/(x_b k_b)$
　end if
end if
→exit

表 **A2** 批量修改电压相角向量核函数伪代码

Table A2 Kernel function pseudo-code for batch modification of voltage phase angle vector

Algorithm2 Kernel_3

→enter

tid←threadID

b←tid

index←b/n

offset←b%n

if $b<(m+s)*n$

$x_1[(n+1)*index+offset]=x[b]$

end if

→exit

附录 B

表 **B1** 软硬件平台

Table B1 Hardware and software platform

	CPU	GPU
硬件平台	E3-1230 v5 @3.40GHz	NVIDIA Tesla P100
软件平台	Visual Studio2017 X64	Visual Studio2017 Cuda 10.0

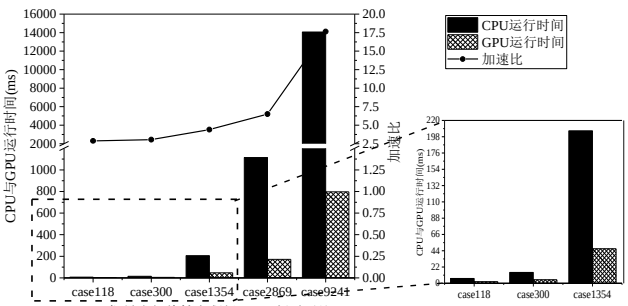

图 **B1** 批量求解线性方程组效率及其放大图

Fig.B1 Efficiency and enlarged drawing of batch solving linear equations

表 **B2** 批量线性方程组求解测试

Table B2 Calculation test of batch solving linear equations

节点数	非零元个数	批量处理规模	耗时（ms）	加速比
118	463	240	2.270	2.870
300	1115	480	4.823	3.049
1354	4763	2251	46.697	4.410
2869	10794	5092	171.426	6.501
9241	37644	17494	796.883	17.645

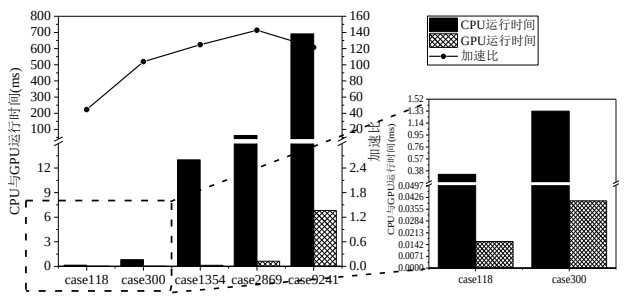

图 **B2** 修改电压相角向量计算效率及其放大图

Fig.B2 Calculation efficiency test of modifying voltage phase angle vector

Occupancy Per SM			
Active Blocks	8	16	
Active Warps	64	64	
Active Threads	2048	2048	
Occupancy	100.00%	100.00%	

图 **B3**　计算性能测试

Fig.B3　Computational performance test

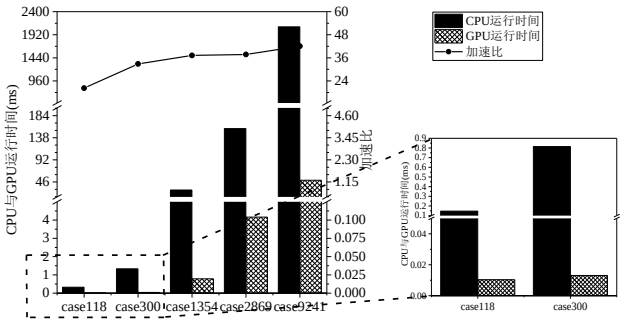

图 **B4**　批量求解支路功率效率及其放大图

Fig.B4　Efficiency test of batch solving branch active power

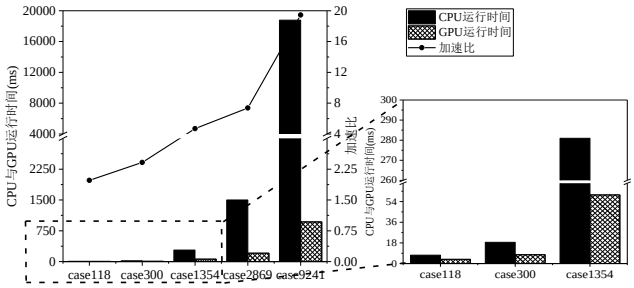

图 B5　整体计算效率及其放大图

Fig.B5　**Efficiency test of all calculations**

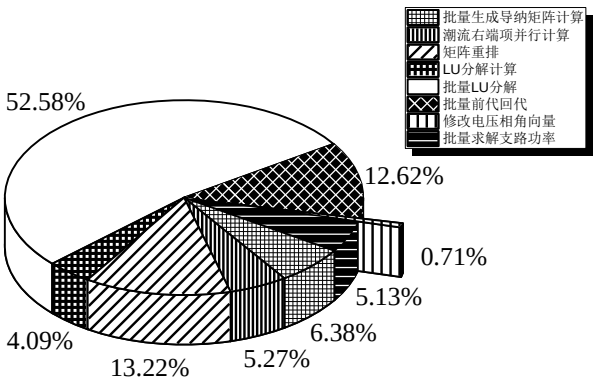

图 **B6**　各部分计算耗时比较

Fig.B6　Comparisons of calculation time of different parts