一种求解机组组合问题的内点半定规划 GPU 并行算法

张宁宇,高 山,赵 欣
(东南大学 电气工程学院,江苏 南京 210096)

摘要:针对内点法求解机组组合问题的半定规划(SDP)模型时大规模线性方程组计算时间太长的问题,提出一种基于图形处理器(GPU)的 Krylov 子空间并行算法。该算法采用预条件处理的拟最小残差法(QMR 法),并以矩阵分块技术为基础,在 CSR 存储格式下使用 GPU 实现 Incomplete Cholesky 并行预处理矩阵的计算。通过对不同规模线性方程组的计算分析表明,与传统的 Cholesky 直接法相比,QMR 并行算法具有速度和存储优势,可获得良好的并行加速比。10~100 机 6 个系统的仿真结果也表明,该 SDP 并行内点法在减少计算时间的同时可求得近似最优解。

关键词:机组组合;半定规划;GPU;QMR;不完全 Cholesky 分解;并行算法;Krylov;线性规划
中图分类号:TM 76 **文献标识码**:A DOI:10.3969/j.issn.1006-6047.2013.07.022

0 引言

半定规划 SDP(SemiDefinite Programming)是线性规划的一种推广,它是在满足约束"对称矩阵的仿射组合半正定"的条件下使线性函数极大(极小)化的问题。目前为止,已有学者对 SDP 在电力系统优化调度中的应用作了初步研究,文献[1]使用 SDP 算法来求解中期水火电力系统调度问题,得到了较好的结果。文献[2-3]针对机组组合模型中存在 0/1 整数变量的特点,引入一种整数约束条件,同时将目标函数和各种约束条件转换成变量的二次形式,最后采用对偶变尺度算法对得到的 SDP 模型进行求解。SDP 算法与拉格朗日松弛法[4]和传统分支定界法[5]将机组给合(UC)模型分割成上下层循环迭代求解不同,它属于一种直接求解算法,尤其是在考虑网络安全约束情况下,可将所有约束化成相应的半定矩阵形式后使用内点法求解。

内点 SDP 存在以下 2 个问题:所需内存较大,由于每个约束条件均对应一个半定矩阵,随着机组组合模型变量和规模的增加,所需的内存空间急剧增加;内点法每次迭代中求解大规模、稀疏、半正定线性方程组需要消耗大量时间。针对第 1 个问题,可利用半定矩阵对称、稀疏的特点,采用稀疏存储技术减少所需内存。第 2 个问题归纳为解 $\boldsymbol{Ax}=\boldsymbol{b}$ 所示的大型稀疏线性方程组,解法可分为直接法和迭代法 2 类。直接法主要是通过对系数矩阵进行变换,如 Gauss 消元、Cholesky 分解、QR 分解等,将原方程组化为三角或三对角等容易求解的形式,然后通过回代或追赶等方法得到方程组的解。该方法一般能得到准确解,但由于固有的前推回代的特点,使得很难实现并行化,因而所需的内存与计算量均很大。迭代法可分为古典迭代法和 Krylov 迭代法两大类,其中,古典迭代法有 Jacobi 迭代法、Gauss-Seidel 迭代法、SOR 等。目前很少用古典迭代法直接求解大规模稀疏线性方程组,常结合 Krylov 迭代法来使用。Krylov 子空间方法的主要思想是为各迭代步递归构造残差向量,即第 n 步的残差向量 $\boldsymbol{r}_n$ 通过系数矩阵 $\boldsymbol{A}$ 的某个多项式与第 1 个残差向量 $\boldsymbol{r}_1$ 相乘得到。迭代多项式的选取应使所构造的残差向量在某种内积意义下相互正交,从而保证某种极小性,达到快速收敛的目的。由于迭代法的存储开销极大减少,同时每步迭代中只包括向量与矩阵的乘法和加法或者向量内积运算,便于并行加速,因此成为稀疏线性方程组并行算法研究的热点。

作为显卡上计算核心的图形处理器 GPU(Graphic Processing Unit),是一种用于密集数据计算的多核并行处理器,计算单元数量要远超过 CPU。已有部分学者对 GPU 在电力系统中的应用进行了研究,并取得了一定的成果。文献[6]把 GPU 应用到潮流计算中,取得了一定的加速比。文献[7-9]利用 GPU 实现了电力系统暂态仿真并行计算,相比传统的串行算法取得了良好的加速比。

为提高内点 SDP 算法中大型稀疏线性方程组的计算效率,本文提出了一种基于 GPU 的 Krylov 子空间并行算法。该并行算法针对系数矩阵稀疏、半正定的特点,采用预条件处理的拟最小残差法(QMR 法),并以矩阵分块技术为基础,在 CSR(Compressed

收稿日期:2012-07-10;**修回日期**:2013-06-18
基金项目:国家高技术研究发展计划(863 计划)资助项目(2011-AA05A105)
Project supported by the National High Technology Research and Development Program of China (863 Program) (2011-AA05A105)

Sparse Row)存储格式下由 GPU 实现了稀疏矩阵的 Incomplete Cholesky 预处理方法和所有迭代计算。实验分析表明,它是一种求解稀疏、对称、半正定线性方程组的有效方法。对 10~100 机 24 时段 6 个不同的算例进行仿真,结果表明:本文算法是一种十分有效的求解机组组合问题的算法,所达到的加速效果随着算例规模的增加而更加明显。

1　机组组合问题的 SDP 模型

1.1　SDP 原理

SDP 是线性规划的推广,是一种非光滑凸优化问题。将现代内点法应用于 SDP 后,可保证求解诸多凸优化问题时在多项式时间内收敛于最优解[10]。

SDP 模型的标准形式如下。

原问题:

$$\begin{aligned}&\min \quad \boldsymbol{C}\cdot\boldsymbol{X}\\&\text{s.t.} \quad \boldsymbol{A}_i\cdot\boldsymbol{X}=b_i \quad i=1,2,\cdots,m\\&\qquad \boldsymbol{X}\succcurlyeq 0\end{aligned}\tag{1}$$

对偶问题:

$$\begin{aligned}&\max \quad \boldsymbol{b}^{\mathrm{T}}\boldsymbol{y}\\&\text{s.t.} \quad \boldsymbol{A}^{\mathrm{T}}\boldsymbol{y}+\boldsymbol{Z}=\boldsymbol{C}\\&\qquad \boldsymbol{Z}\succcurlyeq 0,\boldsymbol{y}\in\mathbf{R}^m\end{aligned}$$

其中,$\boldsymbol{A}_i$、$\boldsymbol{C}$、$\boldsymbol{Z}$ 为 $n\times n$ 实对称矩阵,$\boldsymbol{A}=[\boldsymbol{A}_1\ \boldsymbol{A}_2\ \cdots\ \boldsymbol{A}_m]^{\mathrm{T}}$;$\boldsymbol{y}$ 为 m 维向量;$\boldsymbol{b}=[b_1\ b_2\ \cdots\ b_m]^{\mathrm{T}}$;"$\succcurlyeq$"表示左侧矩阵为半正定矩阵,即该矩阵的特征根均大于等于 0;"$\cdot$"为迹运算符,即 $\boldsymbol{C}\cdot\boldsymbol{X}=\sum_{i=1}^{n}\sum_{j=1}^{n}C_{ij}X_{ij}$。

比较成熟的 SDP 内点法有原始对偶内点法及对偶变尺度法等,本文所采用的不可行内点法属于前一类,可把一个不在可行域内的解作为初始解进行迭代运算,简化了初始化计算。

1.2　不可行内点法

求解式(1)所示的 SDP 原问题等价于求解其对偶问题的对数障碍问题,其一阶 KKT 最优条件为:

$$\begin{cases}\boldsymbol{A}\cdot\boldsymbol{X}=\boldsymbol{b}\\\boldsymbol{A}^{\mathrm{T}}\boldsymbol{y}+\boldsymbol{Z}=\boldsymbol{C}\\\boldsymbol{X}\boldsymbol{Z}=0\end{cases}\tag{2}$$

其中,$\boldsymbol{X}$、$\boldsymbol{Z}\succcurlyeq 0$,用 $\boldsymbol{X}\boldsymbol{Z}=\mu\boldsymbol{I}$ 代替式(2)中第 3 式(μ 为障碍因子,随着 $\mu\to 0$,$\boldsymbol{X}$、$\boldsymbol{Z}$ 的值为相应的最优值),并利用泰勒级数进行展开,得到以下线性系统:

$$\begin{cases}\boldsymbol{A}\cdot\Delta\boldsymbol{X}=\boldsymbol{b}-\boldsymbol{A}\cdot\boldsymbol{X}\\\boldsymbol{A}^{\mathrm{T}}\Delta\boldsymbol{y}+\Delta\boldsymbol{Z}=\boldsymbol{C}-\boldsymbol{A}^{\mathrm{T}}\boldsymbol{y}-\boldsymbol{Z}\\\Delta\boldsymbol{X}\boldsymbol{Z}+\boldsymbol{X}\Delta\boldsymbol{Z}=\mu\boldsymbol{I}-\boldsymbol{X}\boldsymbol{Z}\end{cases}\tag{3}$$

对式(3)进行求解便可得到 $\boldsymbol{X}$、$\boldsymbol{y}$ 和 $\boldsymbol{Z}$ 的搜索方向($\Delta\boldsymbol{X}$,$\Delta\boldsymbol{y}$,$\Delta\boldsymbol{Z}$)。但是直接求解得到的 $\Delta\boldsymbol{X}$ 不是对称矩阵,需引入对称化算子对式(3)中的第 3 式进行处理,算子如下[11]:

$$\boldsymbol{H}_{\mathrm{p}}(\boldsymbol{V})=[\boldsymbol{P}\boldsymbol{V}\boldsymbol{P}^{-1}+(\boldsymbol{P}\boldsymbol{V}\boldsymbol{P}^{-1})^{\mathrm{T}}]/2\tag{4}$$

其中,$\boldsymbol{V}$ 为需进行对称化处理的矩阵;$\boldsymbol{P}$ 表示不同的搜索方向,本文取 $\boldsymbol{P}=\boldsymbol{Z}^{-1/2}$。

得到:

$$\begin{cases}\boldsymbol{A}\Delta\boldsymbol{X}=\boldsymbol{r}_{\mathrm{p}}=\boldsymbol{b}-\boldsymbol{A}\boldsymbol{X}\\\boldsymbol{A}^{\mathrm{T}}\Delta\boldsymbol{y}+\Delta\boldsymbol{Z}=\boldsymbol{R}_{\mathrm{d}}=\boldsymbol{C}-\boldsymbol{A}^{\mathrm{T}}\boldsymbol{y}-\boldsymbol{Z}\\\boldsymbol{E}\Delta\boldsymbol{X}+\boldsymbol{F}\Delta\boldsymbol{Z}=\boldsymbol{R}_{\mathrm{c}}=\\\qquad \mathrm{svec}(\sigma\mu\boldsymbol{I}-\boldsymbol{H}_{\mathrm{p}}(\mathrm{smat}(\boldsymbol{X})\mathrm{smat}(\boldsymbol{Z})))\end{cases}\tag{5}$$

其中,$\boldsymbol{E}=\boldsymbol{P}^{-\mathrm{T}}\boldsymbol{Z}\otimes_{\mathrm{s}}\boldsymbol{P}$,$\boldsymbol{F}=\boldsymbol{P}\boldsymbol{X}\otimes_{\mathrm{s}}\boldsymbol{P}^{-\mathrm{T}}$,$\otimes_{\mathrm{s}}$ 为对称克劳内特积运算;svec 表示将对称矩阵转化为向量的运算,smat 为逆运算 svec 运算的逆运算,表示将向量转化为对称矩阵。消去式(5)中的 $\Delta\boldsymbol{X}$ 和 $\Delta\boldsymbol{Z}$ 可得:

$$\boldsymbol{G}\Delta\boldsymbol{y}=\boldsymbol{h}\tag{6}$$

其中,$\boldsymbol{G}=\boldsymbol{A}\boldsymbol{E}^{-1}\boldsymbol{F}\boldsymbol{A}^{\mathrm{T}}$,$\boldsymbol{h}=\boldsymbol{r}_{\mathrm{p}}-\boldsymbol{A}\boldsymbol{E}^{-1}(\boldsymbol{R}_{\mathrm{c}}-\boldsymbol{F}\boldsymbol{R}_{\mathrm{d}})$。求解线性方程组式(6)得到 $\Delta\boldsymbol{y}$ 后,便可代入式(5)求出 $\Delta\boldsymbol{X}$ 和 $\Delta\boldsymbol{Z}$,然后通过线性搜索得到迭代步长,当对偶间隙 $|\boldsymbol{C}\cdot\boldsymbol{X}-\boldsymbol{b}^{\mathrm{T}}\boldsymbol{y}|$ 满足指定精度时中止迭代。

1.3　机组组合问题的 SDP 模型

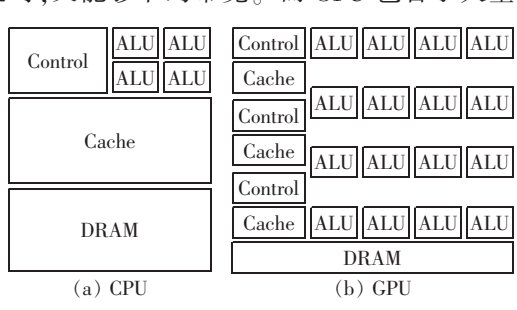
本文采用文献[6]所示的机组组合模型,机组的启动费用采用传统的冷热启动三段式模型,约束条件包括机组出力约束、最小启停时间约束、机组爬坡约束、功率平衡约束和旋转备用约束等。该模型本身并非凸优化问题,原因是机组启停变量为 0/1 整数变量,为此引入辅助变量 $Q^2=1$,同时将 0/1 整数变量约束用凸二次约束 $u_{i,t}^2-u_{i,t}Q=0$ 代替,$u_{i,t}$ 表示机组 i 在时段 t 的启停状态,这样 UC 问题就可描述为一个凸优化问题,进而用内点 SDP 法求解。关于机组组合 SDP 模型的具体介绍可见文献[6]。

2　GPU

2.1　GPU 结构

近年来,随着人们对计算机图像显示效果的要求越来越高,显卡上核心处理器 GPU 的计算量和吞吐量也越来越大。与 CPU 注重逻辑控制不同,GPU 主要负责大规模的密集型数据并行计算。

如图 1 所示,CPU 采用复杂的控制逻辑,用指令来控制单线程可执行程序的执行,还采用大型缓存,既可以减少访问复杂应用程序的指令和数据时产生的延时,又能够节约带宽。而 GPU 包含了大量的计



(a) CPU　　(b) GPU

图 1　CPU 和 GPU 的结构设计

Fig.1　Structure of CPU and GPU

算单元,通过多线程技术来提升运算速度与吞吐量。硬件充分利用因为等待访问内存而产生较长延时的大量线程,减少了控制逻辑中需要执行的线程,因此简化了逻辑控制和缓存单元。图中 Control 表示逻辑控制单元,ALU 表示运算单元,Cache 表示缓冲单元,DRAM 表示内存单元。

2.2 统一计算设备架构编程

统一计算设备架构 CUDA(Compute Unified Device Architecture)是 NVIDIA 在 2007 年推出的一种用于 GPU 的编程模型。该编程模型不需要借助于图形学应用程序编程接口(API),采用比较容易掌握的类 C 语言进行开发。

CUDA 模型中的线程采用线程网格(grid)和线程块(block)2 级结构。线程块网络和线程块中可以分别有多个线程块和线程(thread),且都有唯一的 ID 标志。线程是最基本的计算单元,可以分配到单个数据的计算中。以向量 $\boldsymbol{A}[n]$ 和 $\boldsymbol{B}[n]$ 相加为例,如果该程序在 CPU 上执行需要循环 n 次;当采用 CUDA 编程时,可以分配 n 个线程,单个线程根据自身 ID 来执行向量对应位置上元素的和运算,如 $A[i]+B[i]$,由此实现了并行化计算。

3 基于 GPU 的预条件处理 QMR 法

原-对偶内点法求解机组组合 SDP 模型时,求解式(6)所示的大型线性方程组为主要步骤,矩阵 $\boldsymbol{G}$ 的维数等于模型的约束条件数,例如 10 机 24 时段算例的约束条件数为 1 488,20 机算例为 2 928,100 机算例达到 14 448,且呈现高维数、稀疏、对称、半正定的特点,使用直接法(如 Cholesky 分解)求解需要大量的存储空间以及消耗很长的时间,而 Krylov 子空间法作为 20 世纪 90 年代才完整提出的一类迭代法,具有存储量少、计算量少且易于并行等优点,非常适合于并行求解大型稀疏线性方程组,且结合预条件处理技术可获得良好的收敛特性和较高的数值稳定性,目前已是求解大型稀疏线性方程组的最主要方法。关于 Krylov 子空间法的基本原理详见文献[12],这里不再赘述。

矩阵 $\boldsymbol{G}$ 具有部分特征根接近于 0 的特点,常用于求解对称矩阵的 CG(Conjugate Gradient)法可能会导致数值的不稳定性,因此,本文采用预条件处理 QMR 法来求解大型稀疏线性方程组。

3.1 预条件处理的 QMR 法

设有 n 阶线性方程组:

$$\boldsymbol{J}\boldsymbol{w}=\boldsymbol{k} \tag{7}$$

其中,$\boldsymbol{J}$ 为 $n\times n$ 维的实对称矩阵,$\boldsymbol{k}$ 为 n 维向量,$\boldsymbol{w}$ 为方程的解。

设 $\boldsymbol{w}_0$ 为该方程组迭代计算的初值,算法流程如下。

a. 初始化 $\boldsymbol{w}_0$,计算 $\boldsymbol{r}_0=\boldsymbol{k}-\boldsymbol{J}\boldsymbol{w}_0$,$v_0=0$,$\boldsymbol{q}_0=\boldsymbol{M}^{-1}\boldsymbol{r}_0$,$\rho_0=\boldsymbol{r}_0^{\mathrm{T}}\boldsymbol{q}_0$,$n=1$。

b. 计算 $\boldsymbol{t}=\boldsymbol{J}\boldsymbol{q}_{n-1}$,$\sigma_{n-1}=\boldsymbol{q}_{n-1}^{\mathrm{T}}\boldsymbol{t}$;如果 $\sigma_{n-1}<\varepsilon$,结束计算;否则,$\alpha_{n-1}=\rho_{n-1}/\sigma_{n-1}$,$\boldsymbol{r}_n=\boldsymbol{r}_{n-1}-\alpha_{n-1}\boldsymbol{t}$。

c. 计算 $\boldsymbol{u}_n=\boldsymbol{M}^{-1}\boldsymbol{r}_n$,$v_n=\|\boldsymbol{u}_n\|/\tau_{n-1}$,$c_n=1/\sqrt{1+v_n^2}$,$\tau_n=\tau_{n-1}v_nc_n$,$\boldsymbol{d}_n=c_n^2v_{n-1}^2\boldsymbol{d}_{n-1}+c_n^2\alpha_{n-1}\boldsymbol{q}_{n-1}$,$\boldsymbol{w}_n=\boldsymbol{w}_{n-1}+\boldsymbol{d}_n$。

d. 若 $\rho_{n-1}<\varepsilon$,结束计算;否则 $\rho_n=\boldsymbol{r}_n^{\mathrm{T}}\boldsymbol{u}_n$,$\beta_n=\rho_n/\rho_{n-1}$,$\boldsymbol{q}_n=\boldsymbol{u}_n+\beta_n\boldsymbol{q}_{n-1}$,$n=n+1$,返回步骤 **b**。

由此可知,除去步骤 **c** 中 $\boldsymbol{u}_n=\boldsymbol{M}^{-1}\boldsymbol{r}_n$ 的计算,n 次循环迭代的计算量主要包括 n 次矩阵向量乘法、$3n$ 次向量内积运算和 $3n$ 次向量加减法。上述运算可通过并行稀疏矩阵向量乘法和并行向量运算在 GPU 上实现并行运算,故并行算法实现的难点在于使用 GPU 计算预处理矩阵 $\boldsymbol{M}$ 和实现 $\boldsymbol{u}_n=\boldsymbol{M}^{-1}\boldsymbol{r}_n$ 运算。

3.2 基于矩阵分块的 Incomplete Cholesky 预处理矩阵计算

预处理技术通过改变系数矩阵的条件数来达到加快迭代法收敛速度的目的[12]。目前为止,尚没有一种适用于所有线性方程组的预处理技术,实际计算中应根据系数矩阵的特点来设计相应的预处理矩阵。文献[14-15]根据 Chebyshev 多项式求出矩阵 $\boldsymbol{A}$ 的近似逆矩阵作为 $\boldsymbol{M}^{-1}$,加快了收敛速度,但仅适用于强正则矩阵。文献[16]基于最小化 Frobenius 范数 $\|\boldsymbol{I}-\boldsymbol{A}\boldsymbol{M}\|$ 的方法来设计预处理矩阵,取得了一定的效果。文献[17]提出了一种对称超松弛(SSOR)法预处理技术,通过矩阵相乘和相加运算便可得到矩阵 $\boldsymbol{M}$。Incomplete Cholesky[18]对于正定型系数矩阵是一种有效的预处理方法,但在分解过程中前后行(列)元素之间是顺序进行的,不利于并行化计算。

基于矩阵分块技术,本文提出一种 Incomplete Cholesky 并行预处理方法,在 CSR 稀疏矩阵存储格式下使用 GPU 实现了预处理矩阵的并行计算,提高了计算效率。

在求解 UC 问题的半定规模过程中,矩阵 $\boldsymbol{G}$ 的结构如图 2 所示,考虑到对称性,只对下三角矩阵部分进行分析,阴影部分表示非 0 元素,对角线的元素可分为各个独立的 2×2 矩阵,如 $\boldsymbol{F}_i$、$\boldsymbol{B}_i$ 所示。可以看出矩阵 $\boldsymbol{F}_i$ 之间完全独立,可进行并行 Cholesky 分解;分解完成后,剩余矩阵部分 $\boldsymbol{B}$ 中非 0 元素利用 $\boldsymbol{F}_i$ 的分解结果进行 Incomplete Cholesky 分解,由于 $\boldsymbol{B}$ 中各行元素互相独立,因此可实现并行化;计算完成后,

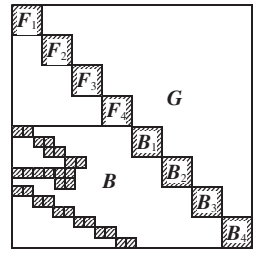


图 2 系数矩阵 $\boldsymbol{G}$

Fig.2 Coefficient matrix $\boldsymbol{G}$

可在矩阵 $\boldsymbol{B}$ 对角线上继续寻找互相独立的 2×2 矩阵进行并行 Cholesky 分解,如此循环直至得到 Incomplete Cholesky 预处理矩阵 $\boldsymbol{M}$。

矩阵 $\boldsymbol{G}$ 的分块流程具体如下。

a. 存放分块信息的向量 $\boldsymbol{P}_{os}$,flag = 1,P_{os}[flag] = 1,矩阵维数为 N,i 取 2,j 取 1。

b. $j=P_{os}$[flag]。

c. 如果 $G[i,j]$为非 0 元且 $i-j>2$,flag=flag+1,P_{os}[flag]=i,转步骤 **e**;否则转步骤 **d**。

d. 如果 $j<i-1$,$j=j+1$;否则,转步骤 **e**。

e. 如果 $i<N$,$i=i+1$,返回步骤 **b**;否则,结束计算。

实际计算中,矩阵 $\boldsymbol{G}$ 以 CSR 格式存储,分块计算中只需遍历非 0 元,有效减少了计算时间;此外,采用原–对偶内点法计算时,经过少数几次迭代后,矩阵 $\boldsymbol{G}$ 的非 0 元的位置便恒定下来,因此,矩阵分块信息不需每次都进行更新。以 10 机系统为例,Incomplete Cholesky 预处理矩阵计算的循环次数从分块前的 1 488 次降至分块后的 96 次,具体的计算消耗时间及加速比将在 4.2 中给出。

3.3　基于 GPU 的预处理 QMR 并行算法

CUDA 并行计算 kernel 以线程网格(grid)的形式组织,每个线程网格由若干个线程块(block)组成,而每个线程块又由若干个线程(thread)组成,实质上,kernel 是以 block 为单位执行的。

使用 GPU 实现基于矩阵分块的 Incomplete Cholesky 预处理矩阵计算如图 3 所示,给每个相互独立对角线上的 2×2 矩阵($\boldsymbol{F}_1$、$\boldsymbol{F}_2$、$\boldsymbol{F}_3$、$\boldsymbol{F}_4$)分配相应线程块(block0、block1、block2、block3)进行 Cholesky 分解;分解完成后,给矩阵 $\boldsymbol{B}$ 分配线程块(block4、block5、block6)对相应行的非 0 元素进行 Incomplete Cholesky 分解。其中,由于矩阵 $\boldsymbol{F}_1$、$\boldsymbol{F}_2$、$\boldsymbol{F}_3$、$\boldsymbol{F}_4$ 中计算量较小,线程块 block0、block1、block2、block3 中线程数取 32;block4、block5、block6 线程设计为二维形式,每一行线程对应于矩阵 $\boldsymbol{B}$ 中某一行的计算。

图 4 所示为 QMR 并行算法在 CPU 和 GPU 中的任务划分,将可以实现并行化的计算放在 GPU 中,而逻辑控制功能和简单的代数运算由 CPU 实现,既

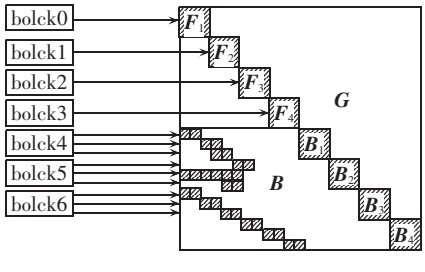


图 3 GPU 中的任务划分

Fig.3 Mission decomposition in GPU

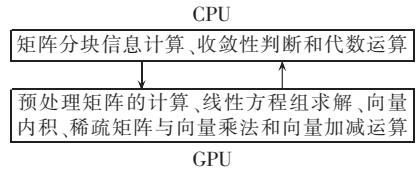


图 4 CPU 和 GPU 的主要计算内容

Fig.4 Main calculation contents of CPU and GPU

充分利用了 GPU 适合密集型并行运算的优点,又发挥了 CPU 具有逻辑控制功能的优势。值得注意的是,GPU 任务中线性方程组求解是指 QMR 算法中的 $\boldsymbol{u}_n=\boldsymbol{M}^{-1}\boldsymbol{r}_n$ 运算,求解时存在前推和回代 2 个过程,前推过程可以利用已得到的矩阵 $\boldsymbol{G}$ 的正向分块信息实现并行化,而回代过程需要对矩阵 $\boldsymbol{G}$ 进行一次反向的分块计算并根据得到的信息并行计算即可。

4　算例分析

4.1　开发环境及算例介绍

本文以 Microsoft Visual Studio 2008 和 CUDA 平台为开发环境,采用 C 语言编写了基于 GPU 的 Incomplete Cholesky 预处理 QMR 并行算法,并在同一平台上编写了基于 CPU 的 Incomplete Cholesky 串行分解和 Cholesky 直接法求解线性方程组的程序。硬件平台:CPU 为 Intel Core 2,主频为 2.4 GHz,内存为 10 GB;GPU 为 NVIDIA Tesla C2050,显存频率为 1 147 MHz。

4.2　Incomplete Cholesky 并行预处理矩阵的计算

衡量并行算法优劣的指标为所需存储量和加速比。对文中所述的 Incomplete Cholesky 预处理 QMR 而言,GPU 并行算法所需的存储量明显小于 CPU 串行算法,因此只对算法的加速比进行分析。表 1 列出了对不同规模的矩阵进行 Incomplete Cholesky 分解时,CPU 串行算法和 GPU 并行算法消耗的时间。可以看出,矩阵维数为 1 488 和 2 928 时,Incomplete Cholesky 分解的串行算法消耗的时间要小于并行算法,这是由于计算量不是太大且数据在 GPU 内存和 CPU 内存间的通信需要消耗一定的时间造成的,随着矩阵维数从 2 928 增加到 14 448,系数矩阵的分块信息没有发生改变,而每次循环计算时增加的计算量可以通过增加线程块的方式实现并行

表 1 不同规模矩阵的 Incomplete Cholesky 并行分解

Tab.1 Incomplete Cholesky parallel decomposition for matrixes of different sizes

编号	N	计算时间/s		加速比
		CPU 算法	GPU 算法	
1	1488	0.0116	0.0143	0.811
2	2928	0.1080	0.1140	0.950
3	5808	0.4930	0.4650	1.060
4	8688	1.2540	1.0720	1.170
5	11568	2.5970	1.9980	1.300
6	14448	4.2030	2.9600	1.420

化,因此 GPU 并行算法消耗的时间逐渐小于 CPU 算法,维数为 14448 时可取得 1.420 的并行加速比。

4.3 基于 GPU 的 QMR 预处理并行算法

QMR 算法中的参数 ε 设为 10^{-10},表 2 所示为 Cholesky 直接法和基于 GPU 的 QMR 预处理并行算法对不同规模线性方程组求解所需的时间。

表 2 不同规模的线性方程组求解

Tab.2 Calculation for linear equation sets of different sizes

编号	N	计算时间/s		迭代次数	加速比
		直接法	并行算法		
1	1488	0.0851	0.0969	18	0.88
2	2928	0.4960	0.3590	24	1.38
3	5808	2.5710	0.6180	7	4.16
4	8688	7.9390	1.3790	12	5.76
5	11568	19.7800	3.0520	35	6.49
6	14448	33.5700	4.5090	37	7.45

与 Incomplete Cholesky 并行算法类似,当维数为 1488 时,直接法的运行时间要小于并行算法,随着维数的增加,并行算法的速度优势逐渐体现出来,最终可取得 7.45 的并行加速比。在不同规模算例情况下,本文并行算法在经过几次迭代后可取得满足计算精度的结果,表明了算法的正确性。表 3 为 N=8688 时,QMR 并行算法运行时间的分布。表中解线性方程表示使用前推和回代求解方程组;向量运算包括向量间、向量与常数以及向量内积等计算;其他部分包括 CPU 和 GPU 之间的数据传递、CPU 的逻辑判断程序和 CPU 上代数计算等。虽然经过矩阵分块技术可以减少 Incomplete Cholesky 分解和使用前推回代法求解 $\boldsymbol{u}_n=\boldsymbol{M}^{-1}\boldsymbol{r}_n$ 的循环计算次数,然而其本身所具有的串行性质,使得其求解时间占总运行时间的绝大部分,而稀疏矩阵与向量乘法和向量相关运行运算在 GPU 可实现良好的并行化。

表 3 QMR 并行算法时间分布

Tab.3 Time consumption distribution of parallel QMR method

参数	Incomplete Cholesky	解线性方程	稀疏矩阵与向量乘法	向量运算	其他
时间/s	1.036	0.297	0.0098	0.0027	0.0335
比例/%	75.1	21.5	0.7	0.2	2.5

4.4 基于 GPU 的内点法求解机组组合问题的 SDP 模型

为了验证并行算法的有效性,采用 10、20、40、60、80 和 100 机 24 时段 6 个测试系统,20~100 机算例通过对 10 机算例的扩展得到,其发电机参数及各时段负荷数据见文献[19],所有机组均考虑爬坡约束,旋转备用取系统总负荷的 10%。

用 SDP 求解机组组合问题是一种直接求解法,不需要多重循环,也不同求解一系列子问题,可同时考虑所有的约束条件,由表 4 可见,通过 30 次左右

表 4 不同算例的内点法计算时间

Tab.4 Time consumption of interior point method for different cases

算例	计算时间/s		迭代次数	加速比
	直接法	并行算法		
10 机	10.13	10.61	27	0.95
20 机	39.02	35.19	28	1.11
40 机	151.26	94.65	29	1.60
60 机	403.74	206.22	30	1.96
80 机	863.11	361.26	30	2.39
100 机	1414.19	542.36	30	2.61

循环计算,便可得到解结果。当算例为 10 机系统时,GPU 并行内点算法的运行时间要略大于直接法,随着算例规模的增大,求解线性方程组式(6)的计算量逐渐增加,GPU 并行算法的并行效率得到了体现,加速比从 10 机系统的 0.95 变大到 100 机系统的 2.61。但是,由于 SDP 内点法除求解线性方程组外的其他计算也需要消耗一定的时间,因此取得的加速比要小于表 2 中的加速比。

建立机组组合模型时,机组启动费用采用时间的指数函数或者冷热启动三段式费用更加符合实际情况,然而却难以转换成 SDP 所要求的凸优化形式,因此,在建立 UC 问题的 SDP 模型时启动费用采用固定值方式,在通过内点法求解得到机组的最优启停状态并进行经济调度时,目标函数中再加入指数或者三段式的启动费用,这样可以保证计算结果的准确性。表 5 所示为采用本文算法对 6 个算例求解得到的运行成本。表 6 为不考虑爬坡约束时本文算法与其他算法计算结果的对比,表中,LR、GA、EP、LRGA、GAUC 分别指拉格朗日松弛法、遗传算法、进化算法、拉格朗日-遗传混合算法、基于分类的遗传算法。

可以看出,本文算法可以很好地处理爬坡约束,得到的运行费用略大于不考虑爬坡约束的情况。从表 6 可以看出,SDP 的并行内点法每次计算只需固

表 5 不同系统的运行成本

Tab.5 Operating cost of different systems

条件	运行成本/$					
	10 机	20 机	40 机	60 机	80 机	100 机
不考虑爬坡	565585	1127476	2250912	3382437	4520544	5621979
考虑爬坡	565751	1127808	2251575	3383405	4521824	5623623

表 6 各种算法计算结果的比较

Tab.6 Comparison of calculating results among different algorithms

算法	运行成本/$					
	10 机	20 机	40 机	60 机	80 机	100 机
LR	565825	1130660	2258503	3394066	4526022	5657277
GA	565825	1126243	2251911	3376625	4504933	5627437
EP	564551	1125494	2249093	3371611	4498479	5623885
LRGA	564800	1122622	2242178	3375065	4501844	5613127
GAUC	563977	1128098	2249790	3384293	4505614	5626514
本文算法	565585	1127476	2250912	3382437	4520544	5621979

定的迭代次数,随机性较小,可获得稳定的计算结果,得到的费用优于部分算法,并可得到较好的近似最优解。

通过上面的分析,可得到以下结论。

a. 基于矩阵分块技术的 Incomplete Cholesky 并行预处理矩阵计算,可获得一定的加速比。由于分解本身具有串行的性质,可取得的加速比有限,同时,在矩阵维数较小且计算量不大情况下,计算时间反而略大于串行算法。

b. 基于 GPU 的 QMR 预处理并行算法是一种有效的大型稀疏线性方程组求解方法,可通过少数几次迭代获得解结果。Incomplete Cholesky 预处理矩阵和前推回代的并行计算时间占解方程总时间的绝大部分,但该预处理法很好地处理系数矩阵半正定的特点,且在维数较大时可获得良好的加速比。

c. SDP 的并行内点法可提高算法的计算效率,但如 1.1 节所示,除式(6)所示的大型稀疏线性方程组求解外,包括其他大量的数据处理以及矩阵和向量间的基本运算,仍需要消耗一定的时间,故算法整体的加速比小于 QMR 预处理并行算法求解线性方程组。在建立机组组合的 SDP 模型时,文中所采用的启动费用的处理方式可以很好地解决启动费用难以化成凸规划形式的问题,提高了结果的准确性。

5　结论

SDP 作为一种很有前景的规划方法,存储量大和计算时间长一直是制约其更广泛应用的瓶颈,文中所述的 SDP 并行内点法可有效地减少程序的运行时间,并通过 10~100 机 6 个系统的仿真表明了算法的有效性,这为 SDP 在 UC 问题中的进一步应用提供了可能,也为求解其他组合优化问题带来了新的思路和方法。

参考文献:

[1] FUENTES-LOYOLA R,QUINTANA V H. Medium-term hydrothermal coordination by semidefinite programming[J]. IEEE Trans on Power Systems,2003,18(4):1515-1522.

[2] MADRIGAL M,QUINTANA V H. Semidefinite programming relaxations for {0,1} power dispatch problems[C]//IEEE Power Engineering Society Summer Meeting,1999. Edmonton,AL,Canada:[s.n.],1999:697-702.

[3] 韦化,吴阿琴,白晓清. 一种求解机组组合问题的内点半点规划方法[J]. 中国电机工程学报,2008,28(11):35-40.
WEI Hua,WU Aqin,BAI Xiaoqing. An interior point semidefinite programming for unit commitment problems[J]. Proceedings of the CSEE,2008,28(11):35-40.

[4] 张晓花,赵晋泉,陈星莺. 基于自适应系统优化算法的机组组合[J]. 电力自动化设备,2009,29(10):93-97.
ZHANG Xiaohua,ZHAO Jinquan,CHEN Xingying. Adaptive system optimization algorithm of power system unit commitment [J]. Electric Power Automation Equipment,2009,29(10):93-97.

[5] 谢国辉,张粒子,舒隽. 基于分层分枝定界算法的机组组合[J]. 电力自动化设备,2009,29(12):29-32.
XIE Guohui,ZHANG Lizi,SHU Jun. Unit commitment based on layered branch and bound algorithm[J]. Electric Power Automation Equipment,2009,29(12):29-32.

[6] 夏俊峰,杨帆,李静. 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. Implementation of parallel power flow calculation based on GPU[J]. Power System Protection and Control,2010,38(18):100-103.

[7] 张宁宇,高山,赵欣. 基于 GPU 的机电暂态仿真细粒度并行算法[J]. 电力系统自动化,2012,36(9):54-60.
ZHANG Ningyu,GAO Shan,ZHAO Xin. A fine granularity parallel algorithm for electromechanical transient stability simulation based on graphic processing unit[J]. Automation of Electric Power Systems,2012,36(9):54-60.

[8] JALILI-MARANDI V,DINAVAHI V. Large-scale transient stability simulation on graphics processing units[C]//IEEE Power & Energy Society General Meeting,2009. PES'09. Calgary,Canada:[s.n.],2009:1-6.

[9] JALILI-MARANDI V,DINAVAHI V. SIMD-based large-scale transient stability simulation on the graphics processing unit[J]. IEEE Trans on Power Systems,2010,25(3):1589-1599.

[10] TODD M J. Semidefinite optimization[J]. Actal Numerica,2001,10(3):515-560.

[11] TODD M J,TOJ K C. On the Nesterov-Todd direction in semidefinite programming[J]. SIAM Journal on Optimization,1998,8(3):769-796.

[12] YOUSEF S. Iterative methods for sparse linear systems[M]. [S.l.]:Society for Industrial and Applied Mathematics,2003:75-98.

[13] FREUND R W,NACHTIGAL N M. A new Krylov-subspace method for symmetric indefinite linear systems[C]//Proceedings of the 14th IMACS World Congress on Computational and Applied Mathematics. Atlanta,USA:[s.n.],1994:1-7.

[14] DAG H,SEMLYEN A. A new preconditioned conjugate gradient power flow[J]. IEEE Trans on Power Systems,2006,18(4):1248-1255.

[15] DAG H. An approximate inverse preconditioner and its implementation for conjugate gradient method[J]. Parallel Computing,2007,33(2):83-91.

[16] CHOW E,SAAD Y. Approximate inverse preconditioners via sparse-sparse iterations[J]. SIAM Journal on Scientific Computing,1998,19(3):995-1023.

[17] HELFENSTEIN R,KOKO J. Parallel preconditioned conjugate gradient algorithm on GPU[J]. Journal of Computational and Applied Mathematics,2011,236(15):3584-3590.

[18] 刘洋. 预条件处理 CG 法大规模电力系统潮流计算[J]. 中国电机工程学报,2006,26(7):89-94.
LIU Yang. The preconditioned CG method for large scale power flow solution[J]. Proceedings of the CSEE,2006,26(7):89-94.

[19] CHENG Chuanping,LIU Chihwen. Unit commitment by Lagrangian relaxation and genetic algorithm[J]. IEEE Trans on Power systems,2000,15(2):707-714.

[20] 张舒. GPU 高性能运算之 CUDA[M]. 北京:中国水利水电出版社,2009:50-75.

(下转第 138 页 continued on page 138)

QIAN Jun,LI Xinran,WANG Ling,et al. Load modeling oriented modeling of photovoltaic cell and fuel cell and its equivalent description[J]. Power System Technology,2011,35(4):135-142.

[24] 王铭,徐玉琴. 基于小波变换和BP网络的非故障相暂态电流故障定位[J]. 电力自动化设备,2006,26(4):25-27.
WANG Ming,XU Yuqin. Location fault using transient component of non-fault-phase based on wavelet analysis and BP neural network[J]. Electric Power Automation Equipment,2006,26(4):25-27.

[25] JONGEPIER A G,van der SLUIS L. Adaptive distance protection of double-circuit lines using artificial neural networks[J]. IEEE Transactions on Power Delivery,1997,12(1):97-105.

[26] 秦英林,田立军,常学飞. 基于小波变换能量分布和神经网络的电能质量扰动分类[J]. 电力自动化设备,2009,29(7):64-67.
QIN Yinglin,TIAN Lijun,CHANG Xuefei. Classification of power quality disturbance based on wavelet energy distribution and neural network[J]. Electric Power Automation Equipment,2009,29(7):64-67.

作者简介:

王 晶(1974-),女,河南邓县人,副教授,博士,主要研究方向为电力系统建模、电能质量仿真与识别、人工智能方法在电力系统中的应用、微网控制与优化方法(**E-mail**:kmhelen@zjut.edu.cn);

李瑞环(1990-),男,浙江富阳人,主要研究方向为直流微网的能量协调控制(**E-mail**:liruihuanzjhz@gmail.com);

束洪春(1961-),男,江苏丹阳人,教授,博士研究生导师,主要研究方向为电力系统继电保护和智能电网的稳定控制。

Energy coordination control based on intelligent MAS for DC microgrid

WANG Jing[1],LI Ruihuan[1],SHU Hongchun[2]

(1. Zhejiang University of Technology,Hangzhou 310014,China;
2. Kunming University of Science and Technology,Kunming 650051,China)

Abstract: An energy coordination control based on intelligent MAS(Multi Agent System) is proposed for the energy management and voltage control of DC microgrid. The framework of DC microgrid is designed and the mathematic models are established for PV,fuel cell and battery. A two-layer MAS is designed for the coordination control:the grid-connection and islanding of DC microgrid and the energy management of PV cells,fuel cells,loads and battery. 8 characteristic variables and 13 operations are extracted from various conditions of micro-sources and loads as,respectively,the input and output parameters of a neural network, which is then trained to obtain the decision maker of central Agent. The proposed strategy of energy coordination control is verified by simulation with MATLAB/Simulink for three operating conditions:variable light intensity,islanding after load increase and grid-connection after load increase.

Key words: multi agent systems; DC microgrid; neural networks; energy management; coordination control; voltage control

(上接第131页 continued from page 131)

作者简介:

张宁宇(1985-),男,山西忻州人,博士研究生,研究方向为并行计算、分布式发电运行与控制和负荷预测(**E-mail**:zhangny190@163.com);

高 山(1973-),男,山东济南人,副教授,研究方向为电力系统规划与优化运行、人工智能在电力系统应用、分布式发电运行与控制、负荷预测(**E-mail**:shangao@seu.edu.cn);

赵 欣(1983-),男,安徽池州人,博士研究生,研究方向为无功优化、电力系统次同步振荡、电力系统稳定分析、并行计算(**E-mail**:fzhjctl@163.com)。

GPU parallel algorithm of interior point SDP for unit commitment

ZHANG Ningyu,GAO Shan,ZHAO Xin

(School of Electrical Engineering,Southeast University,Nanjing 210096,China)

Abstract: Because the time consumption is too heavy when the interior point algorithm is used to solve the SDP(SemiDefinite Programming) model of unit commitment,the Krylov subspace parallel algorithm based on GPU(Graphic Processing Unit) is proposed,which,based on the matrix sectioning technology and in CSR storage format,applies the preconditioned QMR(Quasi-Minimal Residual) method to calculate the parallel Incomplete Cholesky preconditioning matrix by GPU. Analysis of the calculations for linear equation sets of different sizes shows that,with better parallel speedup ratio,QMR parallel algorithm is superior to the traditional Cholesky direct method in speed and memory. Simulative results of six systems of 10~100 units also show that,an approximate optimal solution can be obtained with less computing time by the SDP parallel interior point method.

Key words: unit commitment; semidefinite programming; GPU; QMR; Incomplete Cholesky decomposition; parallel algorithms; Krylov; linear programming