|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | | |  |
| **Elmer-8.4** | | |  |
| 应用**移植指南** | | |
| **文档版本** | **01** | |
| **发布日期** | **2022-07-30** | |
| 华为网格系统---方案4-032.png | | | | |
|  | 华为技术有限公司 | | 附件1-16K |  |

|  |
| --- |
| 版权所有 © 华为技术有限公司2022。 保留一切权利。  非经本公司书面许可，任何单位和个人不得擅自摘抄、复制本文档内容的部分或全部，并不得以任何形式传播。  商标声明  附件3-版权声明页图 和其他华为商标均为华为技术有限公司的商标。  本文档提及的其他所有商标或注册商标，由各自的所有人拥有。  注意  您购买的产品、服务或特性等应受华为公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，华为公司对本文档内容不做任何明示或默示的声明或保证。  由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。 |

|  |  |
| --- | --- |
| 华为技术有限公司 | |
| 地址： | 深圳市龙岗区坂田华为总部办公楼 邮编：518129 |
| 网址： | <https://www.huawei.com> |
| 客户服务邮箱： | [support@huawei.com](mailto:support@huawei.com) |
| 客户服务电话： | 4008302118 |

目 录

[目 录 ii](#_Toc107584868)

[1 介绍 1](#_Toc107584869)

[2 环境要求 2](#_Toc107584870)

[3 移植规划数据 4](#_Toc107584871)

[4 配置编译环境 5](#_Toc107584872)

[4.1 搭建鲲鹏基座软件环境 6](#_Toc107584873)

[4.2 安装Miniconda3 6](#_Toc107584874)

[4.3 安装OpenBLAS 7](#_Toc107584875)

[4.4 安装lapack 8](#_Toc107584876)

[5 获取源码 10](#_Toc107584877)

[6 编译和安装 11](#_Toc107584878)

[7 运行和验证 14](#_Toc107584879)

[8 修订记录 16](#_Toc107584880)

# 介绍

Elmer 是用于偏微分方程数值解的有限元软件。Elmer 能够处理任意数量的方程，因此非常适合模拟多物理问题。它包括结构力学、流体动力学、传热和电磁学等模型。用户还可以编写自己的方程式，与主程序动态链接。

Elmer 由几个部分组成。最重要的是有限元求解器 ElmerSolver、图形用户界面 ElmerGUI 和网格创建和操作工具 ElmerGrid。软件包中还包含一个可视化工具 ElmerPost，但已不再开发。

关于Elmer的更多信息请访问[Elmer官网](https://github.com/ElmerCSC/elmerfem)。

语言：C，C++，Fortran

一句话描述：Elmer 是用于偏微分方程数值解的有限元软件。

开源协议：GPL 2.0，LGPL 2.1

建议的版本

建议使用版本为“Elmer 8..4”。

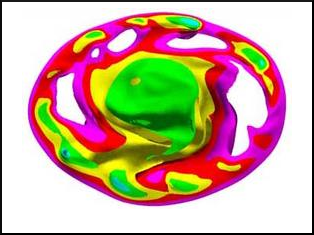


图1.Temperature distribution of melt flow in Czochralski growth of silicon

# 环境要求

硬件要求

硬件要求如表2-1所示。

硬件要求

| 项目 | 说明 |
| --- | --- |
| CPU | Kunpeng 920 |

软件要求

软件要求如表2-2所示。

注意

* 不同HPC应用的依赖软件不同，建议按照如下步骤判断其依赖软件：
* 1、查看其上游社区是否提供安装指导文档；
* 2、搜索网络上是否已经有社区提供相关安装文档或博客；
* 3、尝试安装该软件，根据报错情况，决定安装哪些依赖软件；
* 4、咨询华为工程师是否有相关经验。

软件要求

| 项目 | 版本 | 下载地址 |
| --- | --- | --- |
| Elmer | 8.4 | <https://codeload.github.com/ElmerCSC/elmerfem/tar.gz/scc20> |
| 毕昇编译器 | 2.1.0 | <https://www.hikunpeng.com/zh/developer/devkit/compiler/bisheng> |
| Hyper-MPI | 1.1.1 | <https://github.com/kunpengcompute/hmpi/archive/refs/tags/v1.1.1-huawei.tar.gz> |
| HUCX | 1.1.1 | <https://github.com/kunpengcompute/hucx/archive/refs/tags/v1.1.1-huawei.tar.gz> |
| XUCG | 1.1.1 | <https://github.com/kunpengcompute/xucg/archive/refs/tags/v1.1.1-huawei.tar.gz> |
| OpenBLAS | 0.3.6 | <https://github.com/xianyi/OpenBLAS/releases/download/v0.3.18/OpenBLAS-0.3.6.tar.gz> |
| lapack | 3.8.0 | <http://www.netlib.org/lapack/lapack-3.8.0.tar.gz> |
| 测试算例 | 官方算例包“ElmerTutorialsFiles\_nonGUI.tar.gz” | [https://sources.debian.org/data/non-free/e/elmer-doc/2014.02.06-1~bpo70%2B1/ElmerTutorialsFiles\_nonGUI.tar.gz](https://sources.debian.org/data/non-free/e/elmer-doc/2014.02.06-1~bpo70+1/ElmerTutorialsFiles_nonGUI.tar.gz) |

操作系统要求

操作系统要求如表2-3所示。

操作系统要求

| 项目 | 版本 | 下载地址 |
| --- | --- | --- |
| openEuler | openEuler 20.03 SP3 | <https://repo.openeuler.org/openEuler-20.03-LTS-SP3/> |
| Kernel | 4.19.90 | <https://gitee.com/openeuler/kernel> |

# 移植规划数据

本章节给出Elmer软件在移植过程中涉及到的相关软件安装规划路径的用途及详细说明。

移植规划数据

| 序号 | 软件安装规划路径 | 用途 | 说明 |
| --- | --- | --- | --- |
| 1 | /usr/local/elmer/bisheng | 毕昇编译器的安装规划路径。 | 这里的安装规划路径只是一个举例说明，建议部署在共享路径中。需要根据实际情况调整，**后续章节凡是遇到安装路径的命令，都以现网实际规划的安装路径为准进行替换，不再单独说明。** |
| 2 | /usr/local/elmer/hmpi | Hyper-MPI的安装规划路径。 |
| 3 | /usr/local/elmer/openblas | openblas的安装规划路径。 |
| 4 | /usr/local/elmer/elmer8.4 | Elmer的安装规划路径。 |

# 配置编译环境

前提条件

使用SFTP工具将各安装包上传至服务器对应目录下。

配置流程

配置流程

| 序号 | 配置项 | 说明 |
| --- | --- | --- |
| 1 | 搭建鲲鹏基座软件环境 | 参考4.1 搭建鲲鹏基座软件环境 |
| 2 | 安装miniconda3 | 参考4.2 安装miniconda3。 |
| 3 | 安装OpenBLAS | 参考4.3 安装OpenBLAS。 |
| 4 | 安装lapack | 参考4.4 安装lapack。 |

## 搭建鲲鹏基座软件环境

操作步骤

请参考《鲲鹏基座软件搭建指南.docx》

## 安装OpenBLAS

操作步骤

使用PuTTY工具，以root用户登录服务器。

执行以下命令解压OpenBLAS安装包。

**wget -t 20 -c -P /usr/local/ambertools https://github.com/xianyi/OpenBLAS/releases/download/v0.3.6/OpenBLAS-0.3.18.tar.gz**

注：*wget下载过于缓慢，推荐手动下载再上传到服务器上指定目录/usr/local/ambertools。*

**cd /usr/local/ambertools**

**tar -zxvf OpenBLAS-0.3.6.tar.gz**

执行以下命令进入解压后的目录。

**cd OpenBLAS-0.3.6**

编译OpenBLAS。

**CC=clang CXX=clang++ FC=flang make -j$(nproc)**

**make -j$(nproc)** **PREFIX=/usr/local/ambertools/openblas install**

设置modules文件。

**cat>"/usr/local/ambertools/openblas/OpenBLAS-0.3.6\_modulefiles"<<EOF**

**#%Module1.0**

**conflict OpenBLAS**

**variable modfile [file normalize [info script]]**

**proc getModulefileDir {} {**

**variable modfile**

**set modfile\_path [file dirname \$modfile]**

**return \$modfile\_path**

**}**

**set pwd [getModulefileDir]**

**set OpenBLAS \$pwd**

**setenv OpenBLAS \$OpenBLAS**

**prepend-path LD\_LIBRARY\_PATH \$OpenBLAS/lib**

**EOF**

加载环境变量

**module use /usr/local/ambertools/openblas**

**module load /usr/local/ambertools/openblas/OpenBLAS-0.3.6\_modulefiles**

若要避免每打开一个shell，导入一次变量。可写入到系统配置文件中。

**vi /etc/profile**

**新增如下内容：**

**module use /usr/local/ambertools/openblas**

**module load /usr/local/ambertools/openblas/OpenBLAS-0.3.6\_modulefiles**

----结束

## 安装lapack

操作步骤

使用PuTTY工具，以root用户登陆服务器。

执行以下命令解压 lapack 安装包。

**wget -t 20 -c -P /usr/local/ambertools http://www.netlib.org/lapack/lapack-3.8.0.tar.gz**

注：*wget下载过于缓慢，推荐手动下载再上传到服务器上指定目录/usr/local/ambertools。*

**cd /usr/local/ambertools**

**tar xf lapack-3.8.0.tar.gz**

执行以下命令进入解压后的目录。

**cd lapack-3.8.0**

执行以下命令进行修改文件。

**sed -i '1i\ REAL Function etime(time)' INSTALL/second\_INT\_ETIME.f**

**sed -i '2i\ REAL time(2)' INSTALL/second\_INT\_ETIME.f**

**sed -i '3i\ Call Cpu\_Time(etime)' INSTALL/second\_INT\_ETIME.f**

**sed -i '4i\ time(1) = etime' INSTALL/second\_INT\_ETIME.f**

**sed -i '5i\ time(2) = 0' INSTALL/second\_INT\_ETIME.f**

**sed -i '6i\ End Function' INSTALL/second\_INT\_ETIME.f**

**sed -i 's/REAL ETIME//g' INSTALL/second\_INT\_ETIME.f**

**sed -i 's/INTRINSIC ETIME//g' INSTALL/second\_INT\_ETIME.f**

**sed -i 's/T1 = ETIME( TARRAY )/T1 = etime( TARRAY )/g' INSTALL/second\_INT\_ETIME.f**

**sed -i '1i\ REAL Function etime(time)' INSTALL/dsecnd\_INT\_ETIME.f**

**sed -i '2i\ REAL time(2)' INSTALL/dsecnd\_INT\_ETIME.f**

**sed -i '3i\ Call Cpu\_Time(etime)' INSTALL/dsecnd\_INT\_ETIME.f**

**sed -i '4i\ time(1) = etime' INSTALL/dsecnd\_INT\_ETIME.f**

**sed -i '5i\ time(2) = 0' INSTALL/dsecnd\_INT\_ETIME.f**

**sed -i '6i\ End Function' INSTALL/dsecnd\_INT\_ETIME.f**

**sed -i 's/REAL ETIME//g' INSTALL/dsecnd\_INT\_ETIME.f**

**sed -i 's/INTRINSIC ETIME//g' INSTALL/dsecnd\_INT\_ETIME.f**

**sed -i 's/T1 = ETIME( TARRAY )/T1 = etime( TARRAY )/g' INSTALL/dsecnd\_INT\_ETIME.f**

**cp make.inc.example make.inc**

**sed -i 's/gcc/clang/g' make.inc**

**sed -i 's/gfortran/flang/g' make.inc**

**sed -i 's/librefblas.a/libblas.a/g' make.inc**

编译lapack

**make -j $(nproc)**

拷贝文件到指定目录

**cp \*.a /usr/local/ambertools/openblas**/**lib**

---结束

# 获取源码

操作步骤

下载Elmer源码包“elmerfem-scc20.tar.gz”。。

下载地址：[https://codeload.github.com/ElmerCSC/elmerfem/tar.gz/scc20](https://codeload.github.com/ElmerCSC/elmerfem/tar.gz/scc20" \t "_blank)。

----结束

# 编译和安装

操作步骤

使用PuTTY工具，以root用户登录服务器。

执行以下命令解压AmberTools21安装包。

**wget -t 20 -c "**[**https://codeload.github.com/ElmerCSC/elmerfem/tar.gz/scc20**](https://codeload.github.com/ElmerCSC/elmerfem/tar.gz/scc20)**" -O /usr/local/elmer /elmerfem-scc20.tar.gz**

注：*wget下载过于缓慢，推荐手动下载再上传到服务器上指定目录/usr/local/elmer。*

**cd /usr/local/elmer**

**tar -xvf elmerfem-scc20.tar.gz**

执行以下命令进入解压后的目录。

**cd elmerfem-scc20**

制作Patch补丁。

**vi BiSheng-elmer.patch**

写入以下内容并保存：

**diff -ur back/elmerfem-scc20/fem/src/modules/CoordinateTransform.F90 elmerfem-scc20/fem/src/modules/CoordinateTransform.F90**

**--- back/elmerfem-scc20/fem/src/modules/CoordinateTransform.F90   2020-04-02 21:35:08.000000000 +0800**

**+++ elmerfem-scc20/fem/src/modules/CoordinateTransform.F90  2022-08-17 10:24:34.087281000 +0800**

**@@ -280,6 +280,7 @@**

**LOGICAL :: UsePDecomp**

**REAL :: PDDetTol**

**INTEGER :: PDMaxIter**

**+    logical :: isnan1(3), isnan2(3), isnan3(3), isnan4(3)**

**CALL GetElementNodes(Nodes)**

**tmpvar => VariableGet( Mesh % Variables, 'alpha direction')**

**@@ -315,23 +316,27 @@**

**! surface**

**! -----------------------------------------------------**

**CoordSys(1,1:3) = normalized(MATMUL( alpha(1:nn), dBasisdx(1:nn,:)))**

**-      IF (ANY(ISNAN(CoordSys(1,:)))) THEN**

**+      isnan1 = ISNAN(CoordSys(1,:))**

**+      IF (ANY(isnan1)) THEN**

**print \*, "Element index = ", GetElementIndex(Element)**

**print \*, "Element aspect ratio = ", ElementAspectRatio(Model, Element)**

**CALL Warn('CoordinateTransform','Element coordinate system is NaN, this could be &**

**due to a poor mesh. Let us try to use the degenerate element normal as the local coordinate system alpha vector.')**

**CoordSys(1,1:3) = NormalOfDegenerateElement(Model, Element)**

**-        IF (ANY(ISNAN(CoordSys(1,:)))) CALL Fatal('CoordinateTransform','Degenerate element normal did not work...')**

**+        isnan2 = ISNAN(CoordSys(1,:))**

**+        IF (ANY(isnan2)) CALL Fatal('CoordinateTransform','Degenerate element normal did not work...')**

**END IF**

**CoordSys(2,1:3) = normalized(MATMUL( beta(1:nn), dBasisdx(1:nn,:)))**

**-      IF (ANY(ISNAN(CoordSys(2,:)))) THEN**

**+      isnan3 = ISNAN(CoordSys(2,:))**

**+      IF (ANY(isnan3)) THEN**

**print \*, "Element index = ", GetElementIndex(Element)**

**print \*, "Element aspect ratio = ", ElementAspectRatio(Model, Element)**

**CALL Warn('CoordinateTransform','Element coordinate system is NaN, this could be &**

**due to a poor mesh. Let us try to use the degenerate element normal as the local coordinate system beta vector.')**

**CoordSys(2,1:3) = NormalOfDegenerateElement(Model, Element)**

**-        IF (ANY(ISNAN(CoordSys(2,:)))) CALL Fatal('CoordinateTransform','Degenerate element normal did not work...')**

**+        isnan4 = ISNAN(CoordSys(2,:))**

**+        IF (ANY(isnan4)) CALL Fatal('CoordinateTransform','Degenerate element normal did not work...')**

**END IF**

**CoordSys(3,1:3) = normalized(crossproduct(CoordSys(1,1:3), CoordSys(2,1:3)))**

**@@ -414,8 +419,10 @@**

**REAL :: PDDetTol**

**INTEGER :: PDMaxIter**

**LOGICAL :: Converged**

**+    logical :: isnan5(3,3)**

**-    IF (ANY(ISNAN(RotMloc))) RETURN**

**+    isnan5 = ISNAN(RotMloc)**

**+    IF (ANY(isnan5)) RETURN**

**Converged=.FALSE.**

**DO i=1,PDMaxIter**

**diff -ur back/elmerfem-scc20/fem/src/ZirkaHysteresis.F90 elmerfem-scc20/fem/src/ZirkaHysteresis.F90**

**--- back/elmerfem-scc20/fem/src/ZirkaHysteresis.F90   2020-04-02 21:35:08.000000000 +0800**

**+++ elmerfem-scc20/fem/src/ZirkaHysteresis.F90  2022-08-17 10:27:30.875577000 +0800**

**@@ -1,7 +1,7 @@**

**MODULE zirka ! Pointwise zirka {{{**

**USE ISO\_C\_BINDING, ONLY: C\_INT, C\_LOC, C\_PTR, C\_F\_POINTER**

**USE GeneralUtils**

**-USE DefUtils**

**+!USE DefUtils**

**implicit none**

**private**

**@@ -520,7 +520,7 @@**

**rc\_p => rc**

**d = rc\_p % depth**

**rc\_p => RecurseDepth(rc\_p, B)**

**-  H = rc\_p % simple\_eval(B)**

**+  H = rc\_p % simple\_eval(rc\_p, B)**

**END FUNCTION ! }}}**

**@@ -644,8 +644,8 @@**

**x % dBrev = x%Bq - x%Bp**

**dBout = x % Bq - parent % parent % Bp**

**call master % ABCparams % GetABC(abs(dBout), abs(x % dBrev), x%a, x%b, x%c)**

**-  Hpp = x % parent % parent % simple\_eval(B)**

**-  Hp = x % parent % simple\_eval(B)**

**+  Hpp = x % parent % parent % simple\_eval(x % parent % parent, B)**

**+  Hp = x % parent % simple\_eval(x % parent, B)**

**x % dHrev = Hpp - Hp;**

**! parent => x**

**master % head  => x**

**@@ -723,15 +723,15 @@**

**if (present(rc0)) rc0\_p => rc0**

**k = rc\_p % depth**

**do while (.not. associated(rc\_p, rc\_p % parent % parent))**

**-    if(present(rc0)) X0 = rc0\_p % simple\_eval(B)**

**-    X = rc\_p % simple\_eval(B)**

**+    if(present(rc0)) X0 = rc0\_p % simple\_eval(rc0\_p, B)**

**+    X = rc\_p % simple\_eval(rc\_p, B)**

**if (present(rc0)) print \*, X, X0, X-X0**

**if (.not. present(rc0)) print \*, X, rc\_p % depth ! , c\_loc(rc\_p), c\_loc(rc\_p % parent)**

**rc\_p => rc\_p % parent**

**if(present(rc0)) rc0\_p => rc0\_p % parent**

**end do**

**-  X = rc\_p % simple\_eval(B)**

**-  if(present(rc0)) X0 = rc0\_p % simple\_eval(B)**

**+  X = rc\_p % simple\_eval(rc\_p, B)**

**+  if(present(rc0)) X0 = rc0\_p % simple\_eval(rc0\_p, B)**

**if (present(rc0)) print \*, X, X0, X-X0**

**if (.not. present(rc0)) print \*, X, rc\_p % depth ! , c\_loc(rc\_p), c\_loc(rc\_p % parent)**

**END SUBROUTINE rc\_printeval ! }}}**

**之后为其打上补丁：**

**patch -p0 < BiSheng-elmer.patch**

**之后修改/cmake/Modules/FindMKL.cmake第88行添加：**

**ELSEIF(${CMAKE\_Fortran\_COMPILER\_ID} MATCHES "Flang")**

**# Core libraries**

**SET(MKL\_BASENAME "flang")**

**SET(MKL\_THR\_BASENAME "flang")**

**SET(MKL\_Fortran\_FLAGS "" CACHE STRING "MKL Fortran flags")**

**之后修改fem/tests/CMakeLists.txt**

**将其中**

**FOREACH(D RANGE 1 ${depth})**

编译Elmer。

**mkdir build**

**cd build/**

**export CMAKE\_PREFIX\_PATH=/usr/local/elmer /openblas**

**FLAGS="-mcpu=tsv110 -I/usr/local/elmer/hmpi/include"**

执行cmake相关指令

**cmake -DCMAKE\_INSTALL\_PREFIX=/usr/local/elmer/elmer8.4-DCMAKE\_C\_FLAGS="-O3 -march=armv8.2-a -mtune=tsv110" -DCMAKE\_CXX\_FLAGS="-O3 -march=armv8.2-a -mtune=tsv110" -DCMAKE\_Fortran\_FLAGS="-O3 -march=armv8.2-a -mtune=tsv110" ../**

执行以下命令进行编译安装。

**make -j8**

**make install**

----结束

# 运行和验证

操作步骤

使用PuTTY工具，以root用户登录服务器。

执行以下命令创建hostfile文件并添加节点信息。

**echo**-e 'node1\nnode2\n…\nnodex' > */path/to/HOSTFILE*

下载官方算例包“ElmerTutorialsFiles\_nonGUI.tar.gz”。

链接地址为：<https://sources.debian.org/data/non-free/e/elmer-doc/2014.02.06-1~bpo70%2B1/ElmerTutorialsFiles_nonGUI.tar.gz>

执行以下命令解压官方算例包：

**tar -xvf ElmerTutorialsFiles\_nonGUI.tar.gz**

配置运行环境。

**source****/usr/local/elmer/elmer8.4/elmer.sh**

注：***/usr/local/elmer/elmer8.4/****为举例说明路径，请根据用例存放实际路径修改。*

执行以下命令进入测试目录。

**cd tutorials\_files/FlowResistance**

修改指定文件。

**vi bench\_elmer**

#!/bin/sh

sander=sander.MPI

cat<<EOF  > mdin

       benchmark a short md

&cntrl

       igb=1, cut=20.0, rgbmax=20.0,

       tempi=50.0, temp0=100.0, tautp=0.4, ntt=1,

       ntb=0, nstlim=1000, ntpr=10,

       ntx=1, irest=0,

       ntc=2, ntf=2, tol=0.0000001,

/

EOF

ElmerGrid 1 2 hole.grd -autoclean -metiskway 2

time mpirun --allow-run-as-root -mca btl ^openib -np 1 ElmerSolver

注：*-np 1*指定使用的CPU核数，请根据实际情况修改。

进行单节点测试

**./bench\_elmer**

若进行多节点测试

多节点测试以双节点为例，前提说明：

1、双节点算例运行在共享目录中，如第四章节配置编译中已配置，无需再配置PATH，LD\_LIBRARY\_PATH环境变量

2、相关参数

|  |  |
| --- | --- |
| **参数** | **说明** |
| **-np** | 指mpi运行的总进程数 |
| **-N** | 每个节点上运行的进程数 |
| **--hostfile** | 使用的节点名字 |

3、hostfile文件中放入自己指定的两个计算节点(如*Node1*和*Node2*,其中*Node1*和*Node2*均为主机名称，保证节点相互可以ssh切换成功)，如下所示:

Node1

Node2

进行步骤4修改指定文件：

**mpirun --allow-run-as-root --hostfile** hostfile **-x PATH=$PATH -x LD\_LIBRARY\_PATH=$LD\_LIBRARY\_PATH -np** *192* ***-N*** *96* **-mca btl ^openib sander.MPI ElmerSolver**

**./bench\_elmer**

注：多节点需要配置集群环境。

要验证elmer正确运行，请执行：

若终端输出中出现：

| ElmerSolver: \*\*\* Elmer Solver: ALL DONE \*\*\*

| -----------------------------------------------------

| ElmerSolver: The end

| SOLVER TOTAL TIME(CPU,REAL): 2.28 2.34

| ELMER SOLVER FINISHED AT: 2022/09/01 01:45:17

| real 0m2.366s

| user 0m2.287s

| sys 0m0.034s

则说明elmer测试成功完成。

注：此为单节点运行结果，s反应运行效率。

----结束

# 修订记录

| 发布日期 | 修订记录 |
| --- | --- |
| 2022-07-30 | 第一次正式发布。 |