swOpenFoam 使用说明

Version 0.5

目录

一 概述

目前，神威国产平台和商用 x86 平台上分别部署了 OpenFOAM-2.1.1、OpenFOAM-3.0.0、OpenFOAM-v1706 版和 OpenFOAM-v1806 版。据测试，OpenFOAM-3.0.0 及以上版本中，OpenFOAM-3.0.0 具与其他版本相当的单核性能，但具有最好的 MPI 扩展性（图 1）。对于使用高于 3.0.0 版本 OpenFOAM 的用户，在功能不受到较大限制的情况下，建议转向 3.0.0 版本以获得更好的性能。

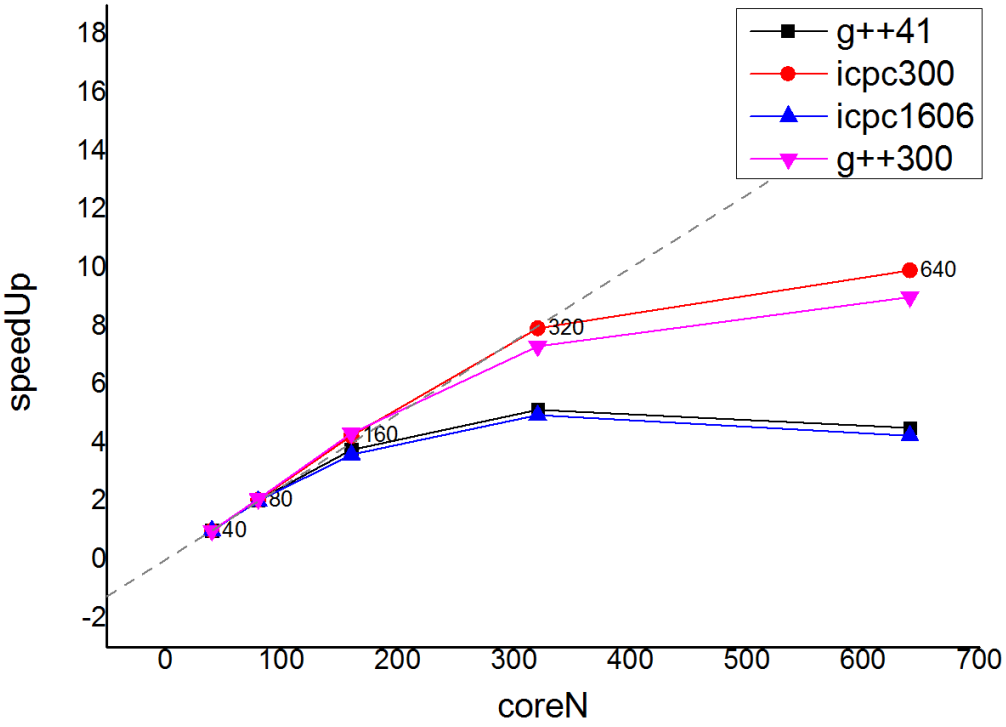

图 1 322 万非结构网格 pisoFoam 求解扩展性对比，图例代表所使用的编译器和 OpenFOAM 版本

最近我们还提供了基于 OpenFOAM-3.0.0 开发的神威众核加速版——swOpenFOAM。经测试，典型算例中整体相对主核加速最高 4.14 倍（国产神威平台每个进程运行于一个核组，一个核组包括 64 众核和 1 个主核），在神威平台利用 swOpenFOAM 进行 CFD 仿真计算将大大加速计算过程。目前已经发布的最新版本为 swOpenFOAM-v0.1 测试版。

在太湖之光上运行 OpenFOAM 算例之前，可以利用一键宏配置相关编译和运行环境，针对不同平台、不同版本的 OpenFOAM，配置命令见表 1：

表 1

平台	国产平台	x86 平台
OpenFOAM-3.0.0	of300sw	of300x86
OpenFOAM-2.1.1	of211sw	of211x86
OpenFOAM-v1706		ofv1706x86
OpenFOAM-v1806		ofv1806x86
swOpenFOAM-v0.1	ofsw	

运行配置命令之后使用 which 命令可以找到相关可执行程序。切换到算例所在目录后，请使用 bsub 提交任务至相应队列进行测试和计算。

本说明的第二章节介绍了直接使用 swOpenFOAM 的相关方法，直接使用 OpenFOAM 的用户可以只阅读该章节，建议同时参考视频培训资料；第三章节主要介绍了 OpenFOAM 用户扩展程序在太湖之光上的编译方法。

二 使用方法

1. OpenFOAM 在太湖之光上的一般使用方法

OpenFOAM 的 tutorials 中自带的 Allrun 脚本在集群环境下不能够使用，建议手动提交或者编写脚本进行批量任务。国产平台 OpenFOAM 使用示例详见图 2：

```
[swsjtu@psn004 motorBike]$ of211sw
[swsjtu@psn004 motorBike]$ which simpleFoam
/home/export/online1/SHARE/OpenFOAM_SW/OpenFOAM-2.1.1/platforms/linux64swgccDPOpt/bin/simpleFoam
[swsjtu@psn004 motorBike]$ ls
0.org  Allclean  Allrun  constant  system
[swsjtu@psn004 motorBike]$ cp -r 0.org/ 0
[swsjtu@psn004 motorBike]$ bsub -I -b -q q_sw_share -share_size 7000 -host_stack 1024 -n 1 blockMesh
Job <6525822> has been submitted to queue <q_sw_share>
some node is sleeping, waiting for dispatch ...
dispatching ...
/*---------------------------------------------------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.1.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
Build  : 2.1.1-baacd438ab95
Exec   : /home/export/online1/SHARE/OpenFOAM_SW/OpenFOAM-2.1.1/platforms/linux64swgccDPOpt/bin/blockMesh
Date   : Mar 16 2017
Time   : 09:04:14
Host   : "vn002901"
PID    : 1212
^^^^^^^^^, tmp is (nil)
Case   : /home/export/online1/systest/swsjtu/OpenFOAM-2.1.1/tutorials/incompressible/simpleFoam/motorBike
nProcs : 1
```

图 2 国产平台 OpenFOAM 使用示例

其中，-share_size 规定了每个进程任务能使用的最大内存，通常取 7000M 左右；-host_stack 是主核开辟的栈空间，默认为 8M，一般而言≥128M，对 OpenFOAM 程序则建议取为 1024M。

```
[swsjtu@psn004 motorBike]$ of300x86
[swsjtu@psn004 motorBike]$ which simpleFoam
/home/export/online1/SHARE/OpenFOAM_x86/OpenFOAM-3.0.0/platforms/linux64IccDPInt64Opt/bin/simpleFoam
[swsjtu@psn004 motorBike]$ blockMesh
/*---------------------------------------------------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  3.0.0                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
Build  : 3.0.0-6abec57f5449
Exec   : blockMesh
Date   : Mar 16 2017
Time   : 09:37:20
Host   : "psn004"
PID    : 30206
Case   : /home/export/online1/systest/swsjtu/OpenFOAM-2.1.1/tutorials/incompressible/simpleFoam/motorBike
nProcs : 1
sigFpe : Enabling floating point exception trapping (FOAM_SIGFPE).
fileModificationChecking : Monitoring run-time modified files using timeStampMaster
allowSystemOperations : Allowing user-supplied system call operations

// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //
Create time
```

图 3 x86 平台 OpenFOAM 在登录节点的运行（不推荐）

如图 3 所示，x86 版本的 OpenFOAM 可以在登录节点直接运行。但是，本地运行一般仅限于单进程，小算例测试，或检查编译是否正常等简单情况。一般情况下请使用 bsub 命令提交到 x86 队列，例如“bsub –I –q q_x86_share –n 4 simpleFoam -parallel”，否则会造成登陆节点负载过大，影响其它用户。

2. swOpenFOAM 的使用方法

基于 OpenFOAM-3.0.0 版本，我们进行了加速和优化，并于 2018 年 12 月 20 日发布了 swOpenFOAM-v0.1 测试版。

类似地，首先需要配置相关编译和运行环境，swOpenFOAM 的配置命令为 ofsw（见图 4）。

```
[guhf@psn002 ~]$ ofsw
[guhf@psn002 ~]$ which simpleFoam
/home/export/online1/SHARE/OpenFOAM_SW/OpenFOAM-SW/OpenFOAM-3.0.0/platforms/linux64s
wgccDPInt32Opt/bin/simpleFoam
[guhf@psn002 ~]$
```

图 4 swOpenFOAM 的配置

而后才可以使用 bsub 命令将计算任务提交至神威平台，典型提交命令为：

```
bsub -I -q q_sw_share -n 8 -share_size 7000 -host_stack 1024 -b -cgsp 64 -o log.sw icoFoam -parallel
```

相对一般主核版本，此处的命令多了若干选项。其中，-b 表示从核函数栈变量放在从核局部存储上，在使用 swOpenFOAM 时必须使用该选项来获得加速性能；-cgsp 则指定每个核组内需要的从核个数，该参数固定为 64。

swOpenFOAM 默认使用加速版的插值计算和一些加速版向量操作。额外的加速接口需要通过选项控制，可在离散格式参数设置文件 fvSchemes 和线性求解器参数设置文件 fvSolution 中进行设置。目前可用的加速接口有 fvSchemes 中 Guass 梯度算子对应的从核加速版本（以下简称加速版）为 swGauss 。替换方式如图 5：

```
21 }
22
23 gradSchemes
24 {
25     default         swGauss linear;
26     grad(p)         swGauss linear;
27     //default         Gauss linear;
28     //grad(p)         Gauss linear;
29 }
30
31 divSchemes
32 {
```

图 5 加速接口在 fvSchemes 的配置

fvSolution 中多重网格矩阵求解器 GAMG 的加速版为 swGAMG，常用多重网格光顺器 GaussSeidel 被升级后的加速版 swCheby 所替代；双共轭梯度线性求解器 PBiCG 的升级版为加速版 swPBiCGStab，配合该求解器可以搭配加速版的预条件子 swDiagonal。注意：swGAMG 常用于对称的压力泊松方程求解，而 swPBiCG 一般用于非对称方程求解。替换方法如图 6 所示：

```
solvers
{
    p
    {
        //solver          GAMG;
        solver        swGAMG;
        tolerance       1e-6;
        relTol          0.01;
        //smoother        GaussSeidel;
        smoother      swCheby;
        nPreSweeps      0;
        cacheAgglomeration true;
        nCellsInCoarsestLevel 50;
        agglomerator    faceAreaPair;
        mergeLevels     1;
    }

    U
    {
        //solver          PBiCG;
        solver          swPBiCGStab;
        //preconditioner  diagonal;
        preconditioner  swDiagonal;
        tolerance       1e-05;
        relTol          0;
    }
}
```

图 6 加速接口在 fvSolution 的配置

线性求解器目前提供最常用方法，但实际问题中，控制方程、离散方式和网格各不相同，不排除出现不收敛的可能。

3. 可视化及并行后处理

在计算比较复杂的算例时可以借助开源软件 ParaView 实现可视化，从而辅助计算设置。首先通过 paraView 命令进行环境配置，而后使用 paraview 命令直接打开 ParaView 的图形界面（注，此处需要 X 窗口程序，推荐使用 MobaXterm）。如图 7 所示，打开 OpenFOAM 网格的一般步骤为：

1. 在 case 文件夹下创建任意名称且后缀为.foam 的空文件（一般设置为与算例同名）
2. 通过 paraview 的文件浏览窗口打开.foam 文件
3. 选择算例类型为 Reconstructed Case（默认初始网格没有被并行分解且网格量较小）

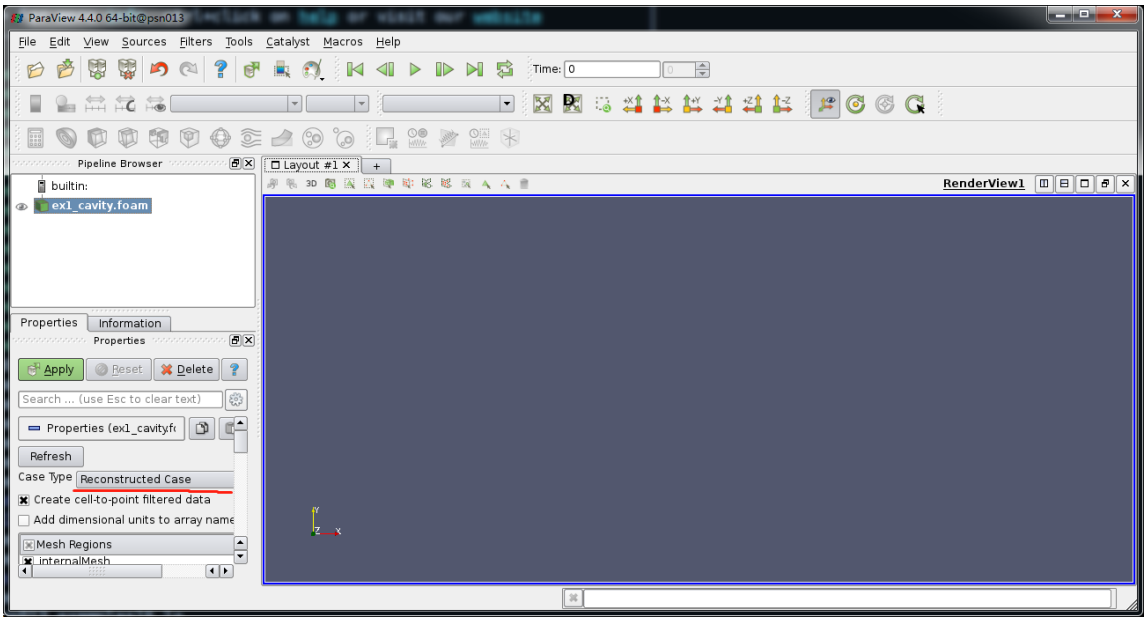

图 7 使用 ParaView 打开 OpenFOAM 网格

另外，使用 ParaView 还可以进行并行后处理，大大提高后处理效率。首先，使用 pvServer 配置 ParaView Server 的环境变量。

然后使用“startPvserver q_x86_expr 8 11121”命令提交任务来启动 pvServer。其中，“startPvServer”是预先写好的启动脚本；“q_x86_expr”是队列名称；“8”是使用的进程数，原则上应小于等于计算使用的进程数；“11121”是端口号，取值范围为 1~49152。为了避免同一登录节点多个用户提交 pvserver 导致端口冲突，端口号尽量选择特殊的值。

任务提交完成之后，日志中 Connection URL 行会反馈两个重要的信息，当前 server 的主机号（Host）和当前端口号（Port），详见图 8（此处 Host：cn060953，Port：11121）：

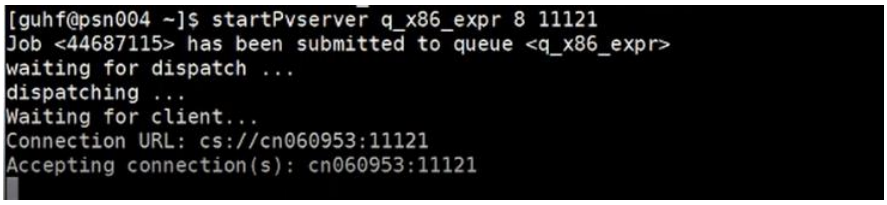

图 8 使用 ParaView 打开 OpenFOAM 网格

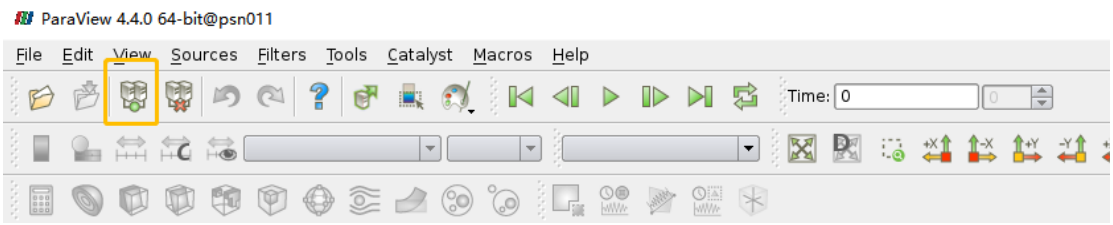

图 9 使用 ParaView 连接 pvServer

进一步打开 ParaView 中的 Connect 按钮（图 9），并在弹窗中点击 Add Server 按钮，修改其中的 Host 为 URL 中提示的主机号，Port 为端口号，最后点击 Connect 按钮（图 10）。在终端窗口刷新的日志中出现“Client connected.”的提示之后，再次使用 ParaView 打开算例，并在 Case Type 中选择“Decomposed Case”来完成 ParaView 的并行后处理（图 11）。

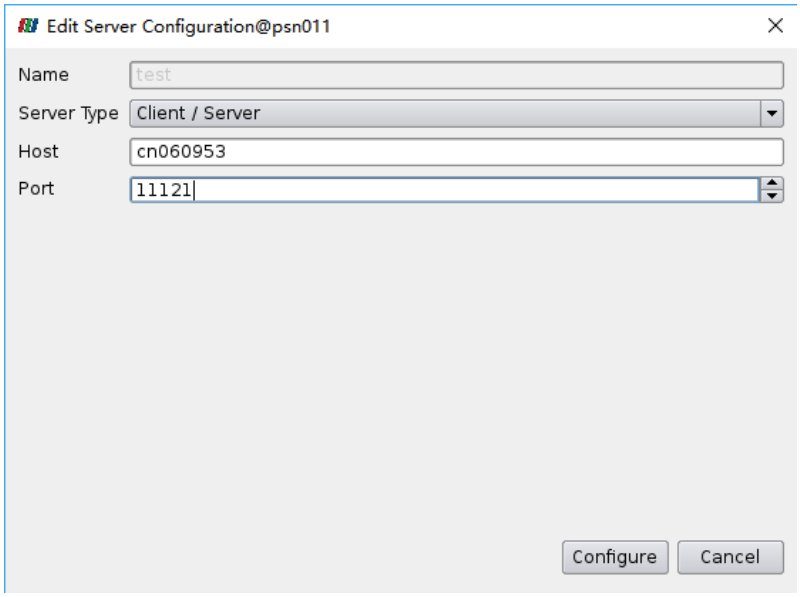

图 10 pvServer 的配置

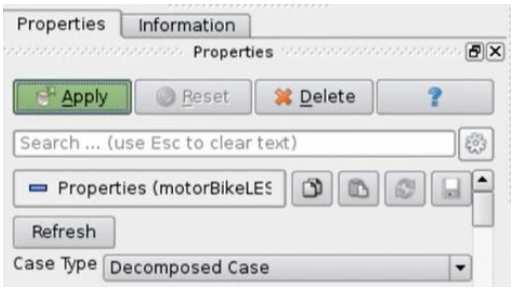

图 11 Case Type 中选择“Decomposed Case”

4. 几点注意

a. 国产神威平台上的 OpenFOAM 不支持 code stream 模式，也不支持 functionObject，因此建议后处理步骤在求解结束后再单独进行。

b. 由于 OpenFOAM 较为庞大，包含的可执行程序和库文件众多，我们并未一一测试。如果发现某个国产神威平台上的前后处理程序不能正常运行，可以切换商用 x86 版本进行后处理。如果 solver 发生错误，请先检查算例

文件是否正确配置，然后与我们支持人员联系。

c. OpenFOAM 各版本的字典输入文件可能格式上有差异，如果发生读入错误，请对应加以修改。

d. 当需要使用自编脚本完成批处理时，脚本中的作业提交命令（bsub …）务必使用-I 选项，使作业阻塞，以确保脚本命令按先后顺序执行。如果想后台运行可以使用“nohup 你的脚本 &”方式，但是要记住进程号以便出错时终止脚本。

e. 对于需要较长参数的可执行程序，如 transformPoints -scale “(0.001 0.001 0.001)”，直接用 bsub 提交可能出现参数传递错误。最好将该命令包裹在一个脚本中，然后用 bsub 命令提交该脚本。

三 OpenFOAM 用户扩展程序编译

1. 商用 x86 平台上扩展程序编译

原理上来讲，x86 是通用平台，可以编译任何版本的 OpenFOAM 及其扩展程序。但是限于系统和编译器版本的原因，这里仅仅针对 OpenFOAM-3.0.0 版本介绍扩展程序编译方法。

首先可以利用已经编译好的OpenFOAM-3.0.0库。运行of300x86配置好编译环境，配置后编译器为 icpc，选项为 OpenFOAM 默认编译选项。

使用 OpenFOAM 自有编译脚本进行编译：仿照 OpenFOAM 文件组织方式，首先建立一个目录包含生成某个对象（库文件或者可执行程序）所需要的源文件，并在该目录下建立 Make 子目录。在 Make 子目录下创建两个配置文件 files 和 options，分别指定源文件和创建目标、引用目录和链接库，示例如图 12：

```
EXE_INC = \
    -I$(LIB_SRC)/TurbulenceModels/turbulenceModels/lnInclude \
    -I$(LIB_SRC)/TurbulenceModels/incompressible/lnInclude \
    -I$(LIB_SRC)/transportModels \
    -I$(LIB_SRC)/transportModels/incompressible/singlePhaseTransportModel \
    -I$(LIB_SRC)/finiteVolume/lnInclude \
    -I$(LIB_SRC)/meshTools/lnInclude \
    -I$(LIB_SRC)/fvOptions/lnInclude \
    -I$(LIB_SRC)/sampling/lnInclude

EXE_LIBS = \
    -lturbulenceModels \
    -lincompressibleTurbulenceModels \
    -lincompressibleTransportModels \
    -lfiniteVolume \
    -lmeshTools \
    -lfvOptions \
    -lsampling
```

```
pisoFoam.C

EXE = $(FOAM_APPBIN)/pisoFoam
```

图 12 Make 子目录

右侧 options 里面 EXE_INC 定义了 include 目录，一般是相应模块下 lnInclude 目录，该目录将对应模块下所有源文件建立了符号链接。EXE_LIBS 定义了需要链接的库，右侧 files 文件中按行列出了需要编译的源文件，以及编译输出对象 EXE。这些为编译可执行程序需要配置的变量，编译库文件方法类似，只是库文件变量前缀为 LIB。另外，注意将最终输出对象的路径修改为本地路径，否则会导致错误，因为 FOAM_APPBIN 等默认路径一般用户没有写权限。完成文件组织后，在包含 Make 目录和所有源文件的目录下运行 wmake [type]进行编译，编译可执行程序 type 为空，编译动态库 type 为 libso，编译动态库 type 为 lib。

如果用户要使用 OpenFOAM-2.1.1 作为基本库，操作与 OpenFOAM-3.0.0 类似。用户也可自行编译不同版本 OpenFOAM 及扩展程序，但高版本中往往利用了 C++11 特

性，目前平台上编译器（ICC-4.0.3 和 GCC-4.4.7）对 C++11 支持并不好。

2. 国产神威平台扩展程序编译

国产神威平台上的编译与 x86 平台最大的差异在于所有库文件只能编译成静态库。在原本的 x86 体系中，动态库的依赖关系是分层次的，也就是说每次生成动态库时只需要链接与之直接有依赖的库，链接生成可执行程序亦然。但由于 OpenFOAM 使用输入参数来决定 C++子类类型，按照原始配置并使用静态链接，实际运行时会由于缺少子类定义而导致出错。因此，在神威平台上编译 OpenFOAM 必须建立依赖库文件补全机制，并将静态库中所有内容都链入可执行程序中。

编译国产神威版本的 OpenFOAM 与 x86 基本有相似之处。以 OpenFOAM-3.0.0 为例，先运行 of300sw 进行编译环境的配置，默认会使用 swg++453 进行编译，使用 swld453 进行链接。swg++453 和 swld453 并不是默认编译器，目前只用于编译 OpenFOAM。然后，参照 x86 文件组织方法组织好文件，但注意 options 文件要按依赖顺序尽量写全所有依赖的静态库（OpenFOAM 的库和用户扩展的库）。在正式编译之前先把 /home/export/online1/SHARE 下的 include、lib、和 bin 拷贝到自己的${HOME}目录，里面存有编译依赖的头文件、库和编译器，然后将${HOME}/bin 加入 PATH 环境变量。之后，使用 wmake 进行扩展模块的编译。

加速版本的 OpenFOAM 对编译器和链接器进行了优化，编译器仍然采用 swg++453，链接器采用 OFswld 进行链接，该链接器位于 wmake 文件夹下，用户无需修改 OpenFOAM 内的原始 options 文件，进入可执行程序所在目录使用 wmake 进行扩展模块的编译即可。

四 结语

如果您在使用过程中发现其他问题，请随时联系我们的工作人员。我们一直在对 OpenFOAM 进行优化，力争为用户提供一个高效稳定的版本。

联系人	邮箱
任虎	renhu@mail.nsccwx.cn
朱一西	zhuyx@mail.nsccwx.cn