关于 SDK-gcc 在 Linux 系统下的环境搭建

如有不足欢迎补充与完善

1、 首先打开官方提供的 SDK—gcc 目录，如图 1：

图 1

其中 readme.txt 文件是对整个目录文件的说明，如图 2：

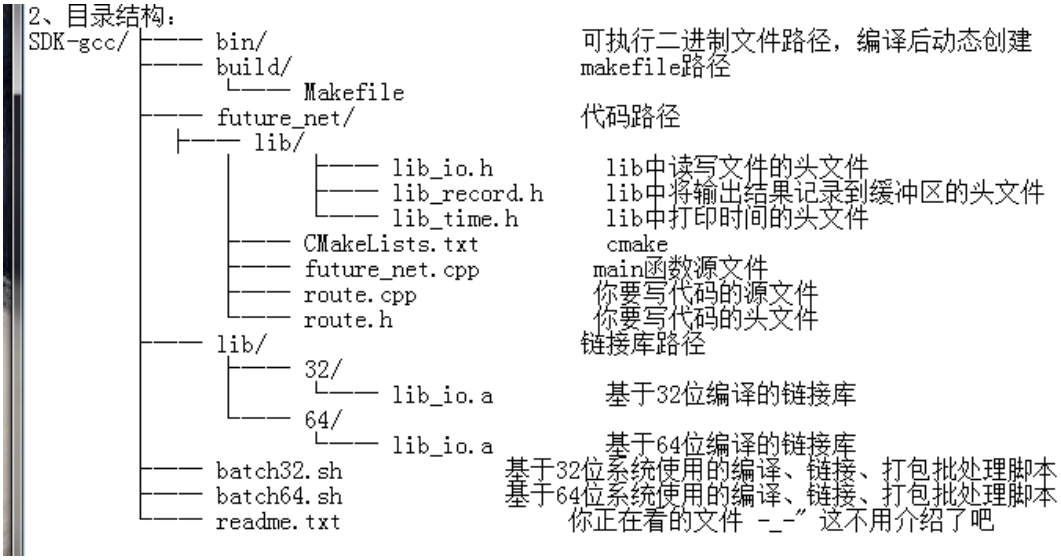


图 2

2、 关于 readme.txt 中，第一点和第四点作一下说明：

1、整体指引：

1)使用一键式脚本编译、链接并打包压缩，如果编译失败请自行解决编译问题；

2)如果编译成功会在 bin 路径下生成可执行二进制文件"future_net"；

3) 使用如下格式调用并调试程序"./future_net /xxx/topo.csv /xxx/demand.csv /xxx/result.csv"，其中 topo.csv 和 demand.csv 是输入文件，result.csv 是输出文件;

4)调试成功后到竞赛官网提交 SDK-gcc 路径下的压缩包"future_net.tar.gz"，稍后查询成绩。

4、手工操作说明：

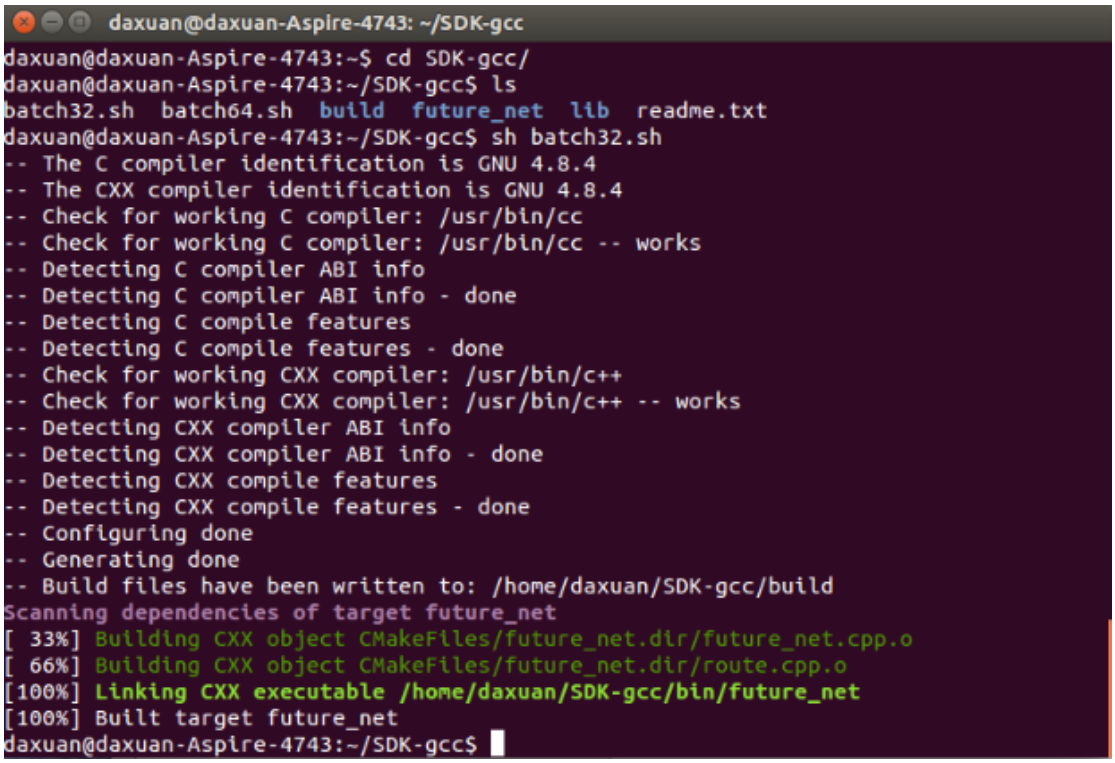
1)根据自己的系统选择 32 位或者 64 位的 lib_io.a，将其复制到 SDK/lib/下；

2)进入 build/下，执行 make 完成编译和链接。生成的二进制文件存放于 bin/下；

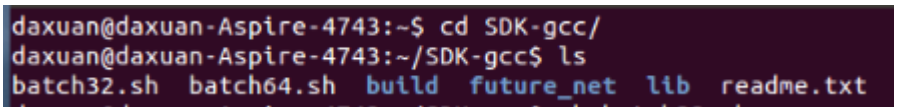
3) 将生成的二进制文件和代码路径置于同一级路径下，打包压缩生成"future_net.tar.gz"。

以下是一键式脚本编译、链接并打包压缩：

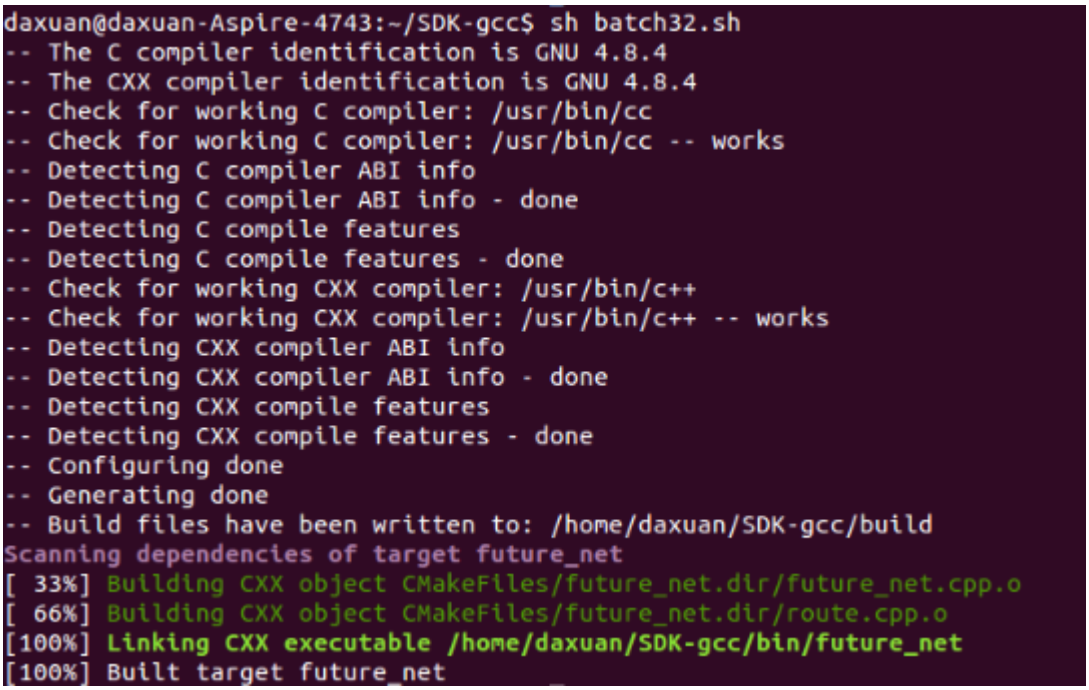
1、 安装好 linux 系统（本人的是 ubuntu 14.04 LTS 32-bit），将整个 SDK-gcc 包复制到 /home 目录下。



2、 cd　SDK-gcc 进入目录，ls 来显示目录中的文件



此时进行一键式脚本编译、链接并打包压缩，运行 sh batch32.sh,结果如下：



或者运用权限的方式执行如下：首先给脚本运行权限 chmod　+x batch32.sh。此时，

batch32.sh 为可执行文件，用 ls 命令显示，注意颜色不同。

```
daxuan@daxuan-Aspire-4743:~/SDK-gcc$ chmod +x batch32.sh
daxuan@daxuan-Aspire-4743:~/SDK-gcc$ ls
batch32.sh  batch64.sh  bin  build  future_net  future_net.tar.gz  lib  readme.txt
```

接着运行：./batch32.sh，结果一样如下：

```
daxuan@daxuan-Aspire-4743:~/SDK-gcc$ ./batch32.sh
-- The C compiler identification is GNU 4.8.4
-- The CXX compiler identification is GNU 4.8.4
-- Check for working C compiler: /usr/bin/cc
-- Check for working C compiler: /usr/bin/cc -- works
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Detecting C compile features
-- Detecting C compile features - done
-- Check for working CXX compiler: /usr/bin/c++
-- Check for working CXX compiler: /usr/bin/c++ -- works
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Detecting CXX compile features
-- Detecting CXX compile features - done
-- Configuring done
-- Generating done
-- Build files have been written to: /home/daxuan/SDK-gcc/build
Scanning dependencies of target future_net
[ 33%] Building CXX object CMakeFiles/future_net.dir/future_net.cpp.o
[ 66%] Building CXX object CMakeFiles/future_net.dir/route.cpp.o
[100%] Linking CXX executable /home/daxuan/SDK-gcc/bin/future_net
[100%] Built target future_net
```

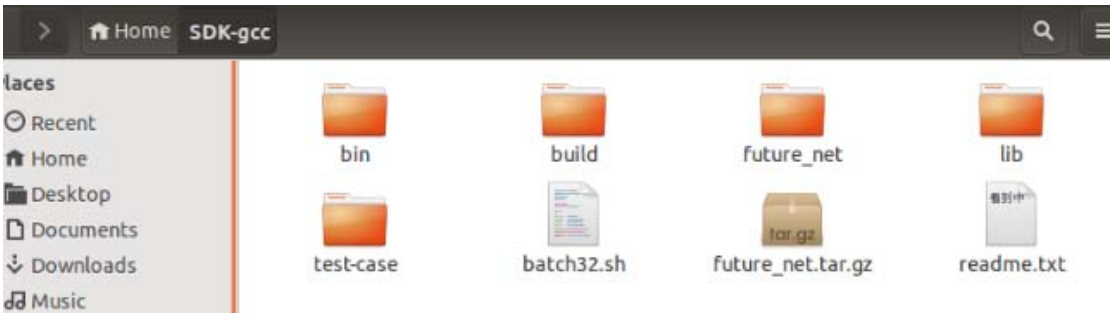
3、 此时进入 bin 目录（即题目中说的如果编译成功会在 bin 路径下生成可执行二进制文件"future_net"）及压缩包"future_net.tar.gz"，结果编译成功，运行 ls 会看见新生成的目录。

```
batch32.sh  batch64.sh  bin  build  future_net  future_net.tar.gz  lib  readme.txt
```

多了一个 bin 目录，进入 bin 目录就会有生成的二进制可执行文件 future_net。

```
daxuan@daxuan-Aspire-4743:~/SDK-gcc$ cd bin/
daxuan@daxuan-Aspire-4743:~/SDK-gcc/bin$ ls
future_net
```

4、 把测试用例 test-case,复制到 SDK-gcc 目录。



5、 调试程序"./future_net /xxx/topo.csv /xxx/demand.csv /xxx/result.csv"，其中 topo.csv 和 demand.csv 是输入文件，result.csv 是输出文件;

```
daxuan@daxuan-Aspire-4743:~/SDK-gcc/bin$ ./future_net  ../test-case/topo.csv ../test-case
/demand.csv ../test-case/sample_result.csv
Begin date/time is: Sun Mar  6 22:27:28 2016
        use time is 0 s 0 ms.
End date/time is: Sun Mar  6 22:27:28 2016
        use time is 0 s 26 ms.
daxuan@daxuan-Aspire-4743:~/SDK-gcc/bin$
```

此时，就会显示你的运行时间。
这时，就可以自己编写 search_route 接口，加入自己的算法了。

以下是手工操作：

1) 根据自己的系统选择 32 位或者 64 位的 lib_io.a，将其复制到 SDK/lib/下；
这步在你运行 batch32.sh 脚步时是没有复制的，该脚本自动做了。

2) 进入 build/下，执行 make 完成编译和链接。生成的二进制文件存放于 bin/下；
这是进入到 build/目录，执行 make 命令，编译结果与一键式脚本编译相同，此时生成了 bin/目录及 future_net 二进制可执行文件。没有压缩包"future_net.tar.gz"，此时就要进行下官方下面的一步进行打包。

3) 将生成的二进制文件和代码路径置于同一级路径下，打包压缩生成"future_net.tar.gz"。

综上，推荐一键式脚本编译、链接并打包压缩。
关于执行脚本遇到的问题：

1、 如果遇到 cmake 的问题，请输入 cmake --version 来检查自己的 cmake 版本，群里同学说要 3.4 版本以上，群里已经上传了 3.5 版本。请自行安装。

2、关于：

1)shell 脚本会清空 bin/下的所有文件和路径，以及 build/下除了 Makefile 外的所有文件和路径。请不要在此保存你的任何文档；

2)如果使用 shell 脚本一键功能，请不要修改任何路径名和文件名，并保持各路径和文件的位置关系不变。

如果你执行 batch32.sh 脚本没有在该脚本目录下执行，会产生以上效果，所有官方建议不复制到别的目录及桌面执行的原因。

本人曾经贪图便宜在 SDK-gcc/bin 目录下，执行了 sh ../batch32。 我就发现 SDK-gcc 目录下好多东西被删除了。

最后，祝大家超常发挥，取得优异成绩。