Group Contribution 3.2 dev Manual

1 概述

Group Contribution 是一款基于基团贡献法计算分子各种理化性质的程序包。该程序接受分子的SMILES作为输入，自动统计不同基团数目，然后输出各种理化性质。目前支持计算的理化性质包括闪点、冰点、沸点、临界温度、临界压力、临界体积、密度、生成吉布斯自由能、生成焓、蒸发焓、摩尔体积（298K）、燃烧热、质量热值、比冲。

1.1 背景

1.1.1 基团贡献法

自1923年Macleod提出计算液体等张比容的基团贡献法以后，六十年来基团贡献法得到了迅速的发展。基团贡献法是一种具有较高准确度的估算化合物物化性质的方法。该方法具有预测过程简便，适用范围广，通用性较好等优点。自上世纪中叶提出以来，被广泛应用于纯物质的物性估算。此方法也用于多相之间进行平衡估算中，尤其是气相和液相之间的平衡估算。基团贡献法是预测相平衡的唯一方法，不仅能对纯物质的物理性质与热力学性质进行估算预测，也可对混合物的热力学性质进行预测。现在，用基团贡献法不仅可以预测纯物质的各种物理性质(包括：临界参数、等张比容、折光率、热导系数、粘度、摩尔体积等)，也可以预测纯物质的各种热力学性质(包括：热容、蒸发热、饱和蒸气压、标准生成热、标准熵等)，还能预测混合物的热力学性质(如：活度系数、超额烩等)。

基团贡献法的基本假设是纯化合物或混合物的物性，等价于构成此化合物或混合物的各种集团对物性的贡献之和，换言之，假定同一基团在不同体系内对物性的贡献是相同的。集团贡献法的优点是具有最大的通用性，物质世界中存在的分子数量难以计数，但是构成常见有机物的基团仅几百种，因此，若能利用一些已有的实验值来估计不同集团对不同物性的贡献值，就可以利用他们取预测其他有机化合物的性质。

Group Contribution 3.2 dev 实现的是《Group-contribution based estimation of pure component properties》中提出的三顺序基团贡献法，同时加入了《Group-contribution+ (GC+) based estimation of properties of pure components: Improved property estimation and uncertainty analysis》中新增加的基团和参数。

1.1.2 SMILES

SMILES（Simplified molecular input line entry system），简化分子线性输入规范，是一种用ASCII字符串明确描述分子结构的规范。SMILES由Arthur Weininger和David Weininger于20世纪80年代晚期开发，并由其他人，尤其是日光化学信息系统有限公司（Daylight Chemical Information Systems Inc.），修改和扩展。

由于SMILES用一串字符来描述一个三维化学结构，它必然要将化学结构转化成一个生成树，此系统采用纵向优先遍历树算法。转化时，先要去掉氢，还要把环打开。表示时，被拆掉的键端的原子要用数字标记，支链写在小括号里。

典范SMILES保证每个化学分子只有一个SMILES表达式。典范SMILES常用于分子数据库的索引。 记法：

1. 原子用在方括号内的化学元素符号表示。 例如[Au]表示“金”，氢氧根离子是[OH-]。 有机物中的C、N、O、P、S、Br、Cl、I等原子可以省略方括号，其他元素必须包括在方括号之内。
2. 氢原子常被省略。对于省略了方括号的原子，用氢原子补足价数。 例如，水的SMILES就是O，乙醇是CCO。

3. 双键用“=”表示；三键用“#”表示。 含有双键的二氧化碳则表示为O=C=O，含有三键的氰化氢表示为C#N。
4. 如果结构中有环，则要打开。断开处的两个原子用同一个数字标记，表示原子间有键相连。 环己烷(C6H12)表示为C1CCCCC1。需要注意，标志应该是数字(在此例中为1)而不是“C1”这个组合。扩展的表示是(C1)-(C)-(C)-(C)-(C)-(C)-1而不是(C1)-(C)-(C)-(C)-(C)-(C)-(C1)。
5. 芳环中的C、O、S、N原子分别用小写字母c,o,s,n表示。
6. 碳链上的分支用圆括号表示。 比如丙酸表示为CCC(=O)O，FC(F)F或者C(F)(F)F表示三氟甲烷。
7. 在芳香结构中的N原子上连有一个H原子，用[nH]表示
8. 用@和@@表示手性

本程序关于SMILES的读写均是基于外部python库Rdkit实现的。

1.2 依赖的环境

本程序包的开发和测试环境为：

- python == 3.11.0
- tqdm == 4.66.1
- numpy == 1.26.0
- pandas == 2.1.1
- xlrd == 2.0.1
- rdkit == 2023.3.3
- mpi4py == 3.1.5 (optional)

上述依赖包的更旧或更新的版本应该也不会有太大问题，但我们没有做更多测试。mpi4py仅在需要多核并行时是必要的，若只需单核运行则不需安装mpi4py。我们在window11和ubuntu18.04都进行了测试，没有发现异常。

1.3 程序结构

Group Contribution 3.2 dev 的程序结构是这样的：

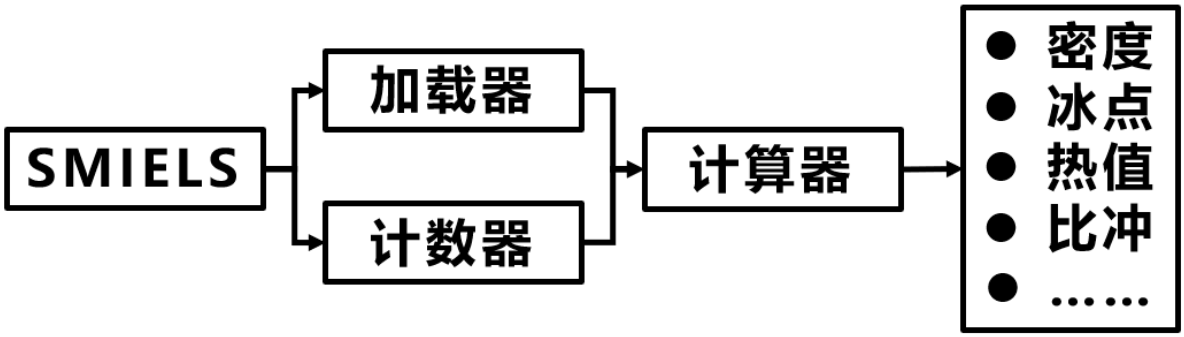
- `gp_3x_loader.py`：负责加载用于计算分子性质的基团参数和记录基团统计顺序的文件等内部数据。
- `gp_3x_counter.py`：负责统计给定分子的不同基团数目，还可以直接输出统计结果。
- `gp_3x_calculator.py`：接收`gp_3x_counter.py`中实现的Counter和`gp_3x_loader.py`中实现的Loader的结果，然后基于基团贡献法和内部基团参数计算分子性质。
- `gp_3x_mpirun.py`：基于外部python库mpi4py实现并行批量计算分子性质的脚本。
- `main.py`：主程序接口，用户可以在这里自行调用所需功能。
- `test_gp_3x_counter.ipynb`：测试用代码，主要用于测试`gp_3x_counter.py`中定义的函数的可靠性。

还有两个重要的文件夹：

- `gp_3x_internal_data`：保存了Group Contribution所必需的内部数据，其中，`group_contribution_parameters.xlsx`中记录了所有基团的模型参数，`group_order.xlsx`中记录了`gp_3x_counter.py`所需的基团统计顺序。
- `gp_3x_test_mol`：主要是一些测试程序的输入文件和结果，或可作为输入文件模板。

1.4 工作流程

工作流程如下图所示，程序接收分子的SMILES作为输入，加载器（loader）加载内置的基团贡献法参数，计数器（counter）根据内置的基团统计顺序统计不同基团的数目。计算器（calculator）接收加载器和计数器的结果计算给定分子的各种理化性质。



2 快速入门

本节介绍如何使用该程序计算分子的理化性质和统计给定分子的基团数目，以及一些其他的主要功能。

2.1 输入和输入文件

Group Contribution 支持两种运行模式，一种运行模式是计算单个分子，另一种是批量计算给定文件中记录的所有分子。

2.1.1 单个分子

计算单个分子时可以直接给出给定分子SMILES，如环己烷的SMILES为`C1CCCC1`

2.1.2 分子文件

当需要批量计算大量分子时，需要给出记录分子SMILES的文件，目前支持的文件格式包括：`.txt`,`.csv`,`.xlsx`

其格式可按如下书写

2.1.2.1 txt

```
CCCCCCCC
CCCC(C)C
CCC(C)(C)C
C=CCCCC
C/C=C/CCC
C=C(C)CC
CC=C(C)C
CC(C)=C(C)C
C=C=CC
C=C=C(C)C
```

每行只有一个分子的SMILES（不要有空格）。

2.1.2.2 csv

```
index,smiles,molar_mass
0,CC,30.069999999999993
1,C1CC1,42.08100000000002
2,CCC,44.09700000000002
3,C1CCC1,56.10800000000002
```

格式比较自由，只要有一列列名为`smiles`的列即可，其余数据会被忽视。

2.1.2.3 xlsx

index	smiles	molar_mass
0	CC	30.07
1	C1CC1	42.081
2	CCC	44.097
3	C1CCC1	56.108
4	CC1CC1	56.108

格式比较自由，只要有一列列名为`smiles`的列即可，其余数据会被忽视。

2.2 运行方式

在使用本程序时，用户可以直接使用我们提供的主程序`main.py`。同时，为了保留python本身最大的可拓展性，用户也可以将本程序作为外部库导入进用户自行编写的python脚本中。

下面分别介绍这两种使用方式。

2.2.1 GC作为独立程序

当配置好1.2小节中的环境后，在本程序的主目录下进入终端后，只需输入：

注意，笔者在编写手册时使用的系统时Windows 11，终端是Anaconda提供的Anaconda Powershell Prompt，其中支持一些Linux中的常用命令，而且同时支持Linux和Windows的路径格式，请用户在使用时自行区分不同系统的文件路径格式！

```
python main.py
```

即可进入GC的主程序，用户将看到如图2.1所示的主界面：

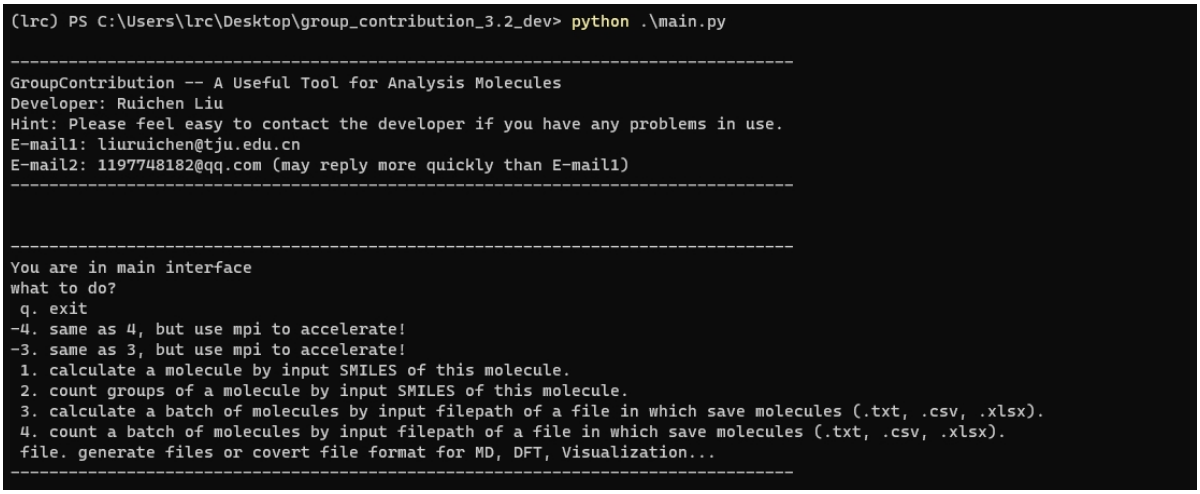


图2.1 GC程序的主界面

程序最开始显示一段基本信息，包括程序名、开发者以及开发者的联系方式（用户在使用中遇到任何问题都可联系开发者，开发者会在能力范围内尽可能提供帮助）。然后显示用户所处位置（主界面），程序询问用户要做什么操作，用户只需输入对应的序号即可命令程序执行相应的任务。

2.2.1.1 退出(q)

若要优雅地退出本程序，只需在主界面内输入`q`然后在键盘上敲击`Enter`即可，如图2.2所示。

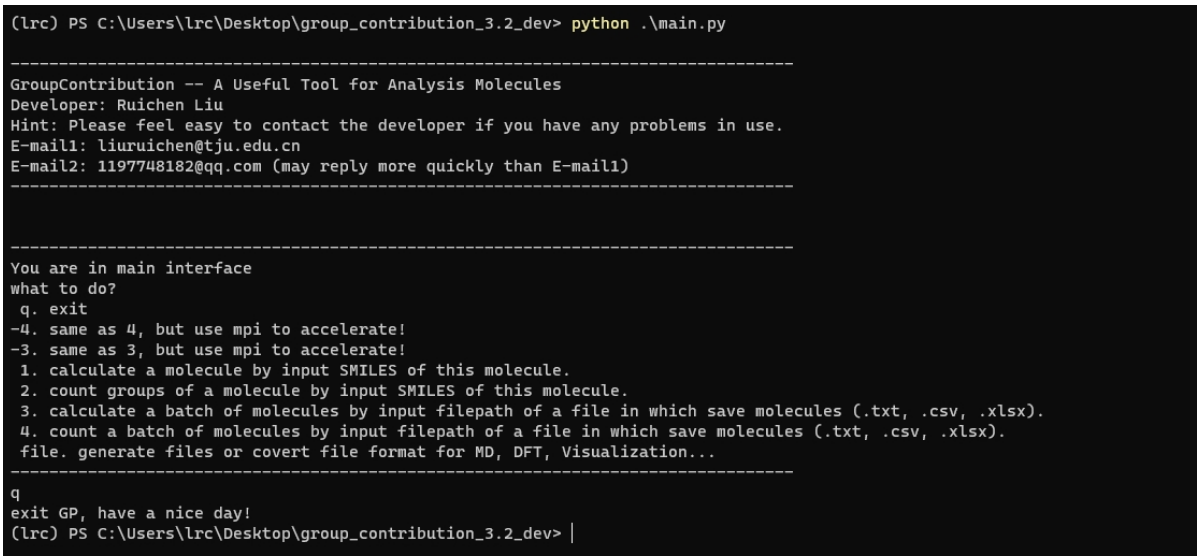


图2.2 GC程序的退出

程序正常退出后，自动退回了终端。

2.2.1.2 计算单个分子的理化性质(1)

若要计算单个分子的理化性质，只需在主界面内输入`1`，然后在键盘上敲击`Enter`，根据后续的提示输入所需指令即可，如图2.3所示。

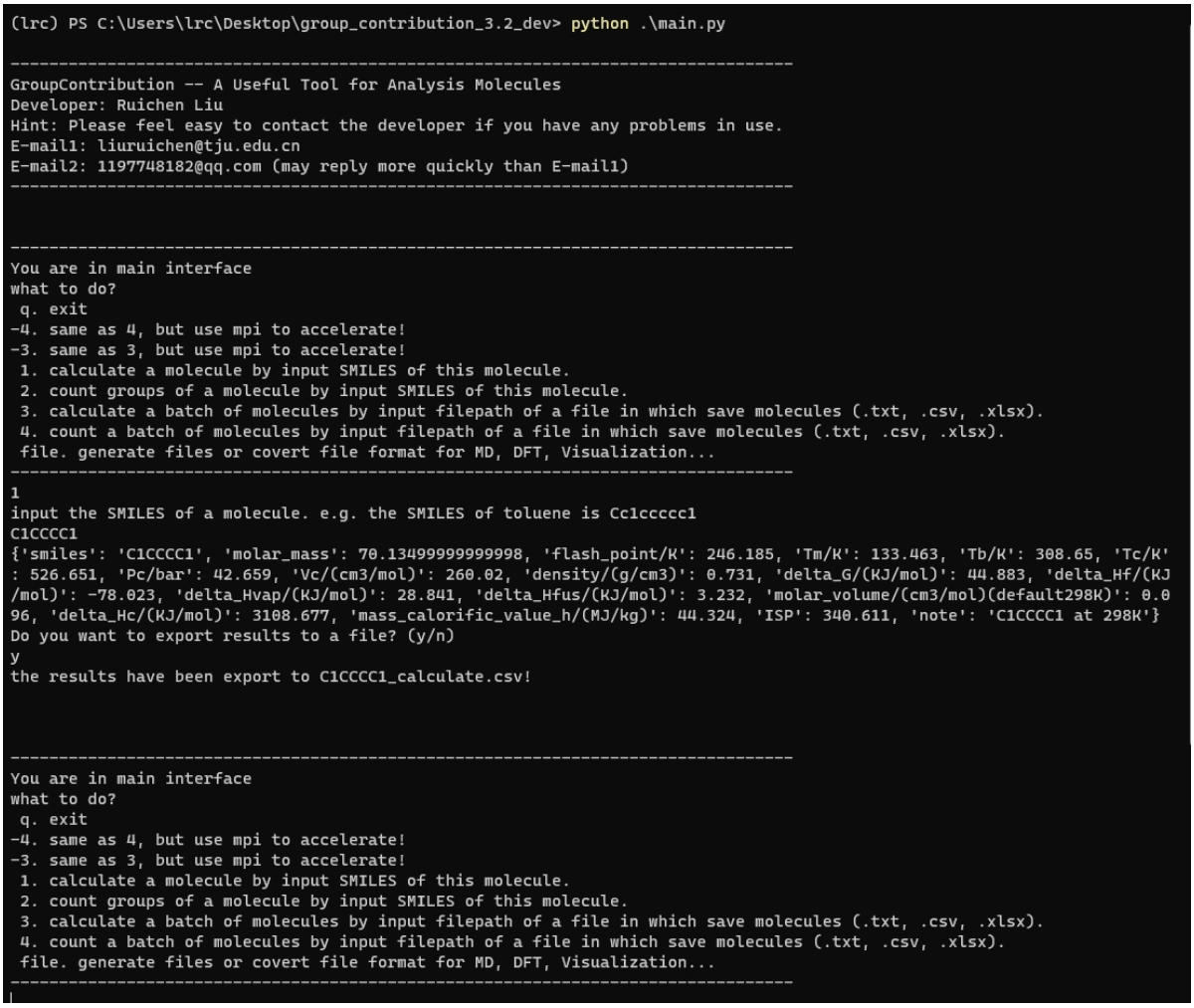


图2.3 GC程序的主功能1

当通过在主界面输入`1`进入主功能1（计算单个分子的理化性质后），程序会提示用户键入目标分子的SMILES表达式。当用户输入要计算的分子的SMILES（如环戊烷C1CCCC1）并敲击`Enter`后，程序开始进行计算，然后将计算结果输出到屏幕上。之后程序会询问用户是否需要将此结果导出为一个`.csv`文件，若用户确实想要导出，则键入`y`并敲击`Enter`，一个名为`{当前计算的分子的SMILES}_calculate.csv`的文件将产生在程序的主目录下。若用户不需将结果输出为文件，键入`n`并敲击`Enter`即可。

之后程序会再次返回主界面，此时若键入`q`并敲击`Enter`即可优雅地退出程序。

2.2.1.3 统计单个分子的基团数目(2)

若要统计单个分子的基团数目，只需在主界面内输入`2`，然后在键盘上敲击`Enter`，根据后续的提示输入所需指令即可，如图2.4所示。

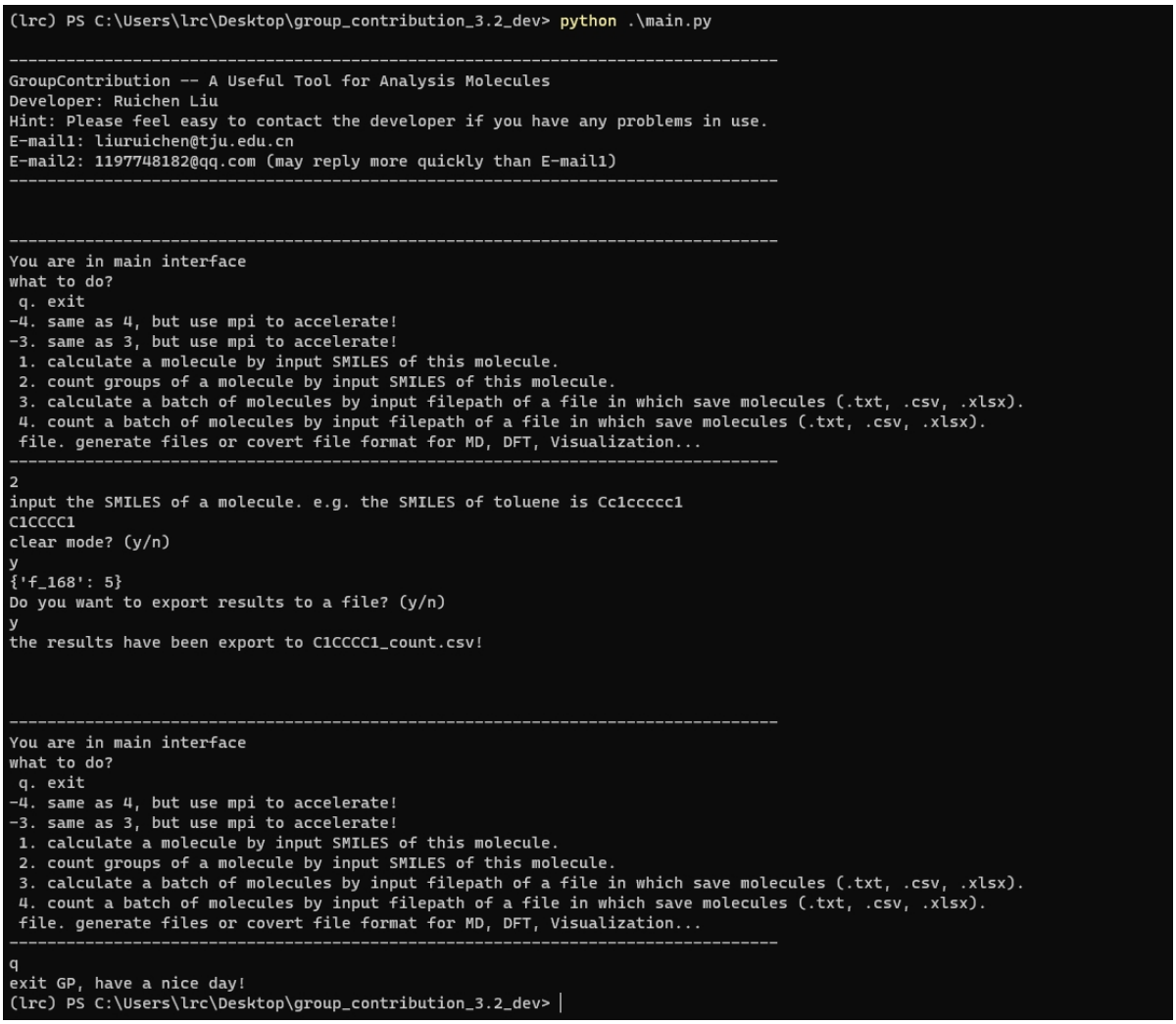


图2.4 GC程序的主功能2，使用清爽模式输出结果

当通过在主界面输入`2`进入主功能2（统计单个分子的基团数目后），程序会提示用户键入目标分子的SMILES表达式。当用户输入要计算的分子的SMILES（如环戊烷C1CCCC1）并敲击`Enter`后，程序会询问用户是否使用清爽模式，清爽模式是指仅输出数目不为零的基团结果，而那些数目为零的基团则不会输出其结果（因为已经是0了）。当用户想要以清爽模式输出统计结果时，键入`y`然后敲击`Enter`，程序就开始进行统计，然后将统计结果输出到屏幕上。统计完成后，程序会询问用户是否需要将此结果导出为一个`.csv`文件，若用户确实想要导出，则键入`y`并敲击`Enter`，一个名为`{当前计算的分子的SMILES}_count.csv`的文件将产生在程序的主目录下。若用户不需将结果输出为文件，键入`n`并敲击`Enter`即可。

之后程序会再次返回主界面，此时若键入`q`并敲击`Enter`即可优雅地退出程序。

若用户希望将全部统计结果都输出，则不应使用清爽模式（这在希望获取基团贡献法风格的分子指纹时很常见），此时用户的输入和程序的输出见图2.5。

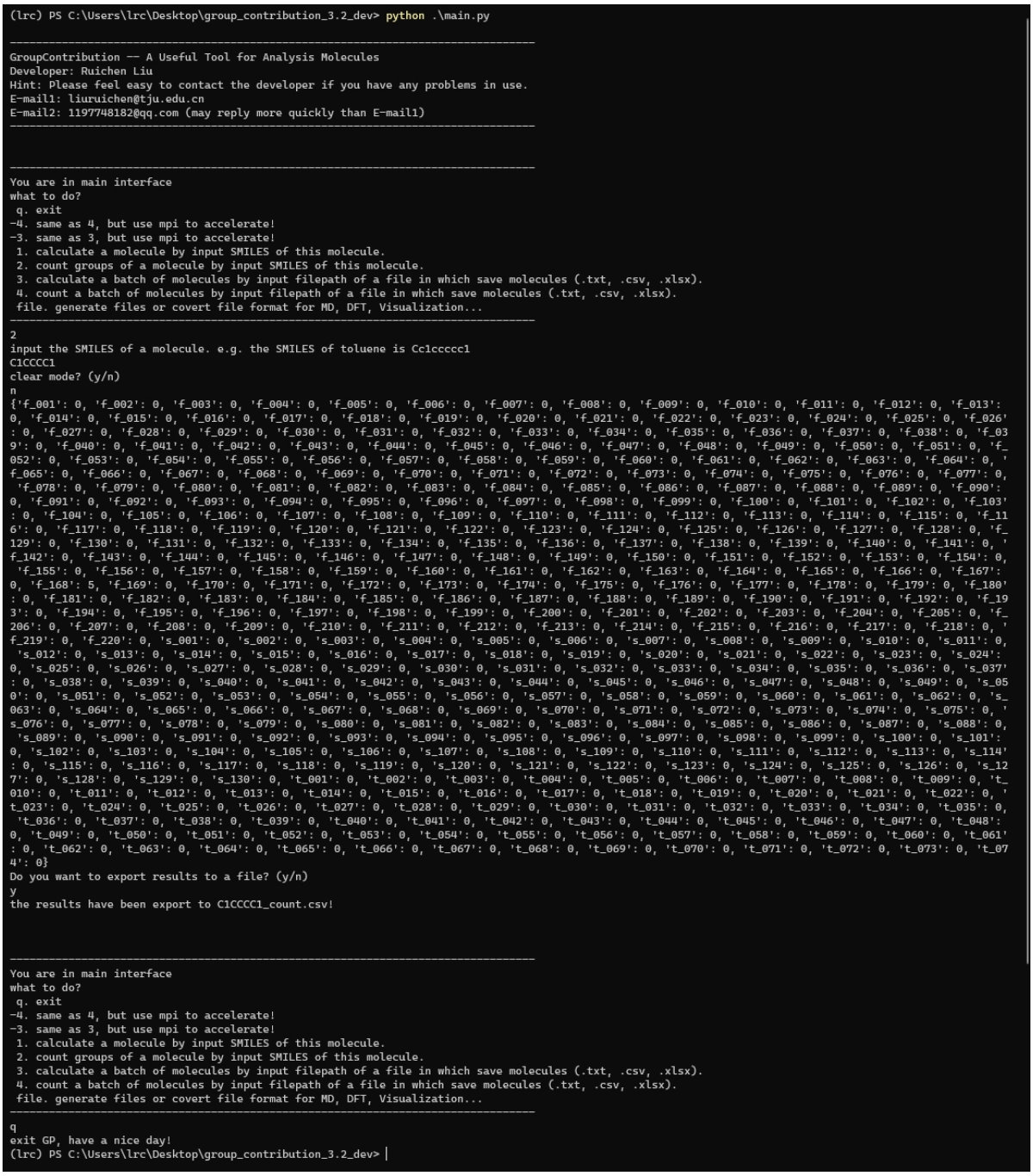


图2.5 GC程序的主功能2，不使用清爽模式输出结果

2.2.1.4 批量计算分子的理化性质(3)

若用户需要计算一批分子的理化性质，需要先准备一个2.1.2小节中介绍的分子文件，然后在主界面内输入`3`，然后在键盘上敲击`Enter`，根据后续的提示输入所需指令即可，如图2.6所示。

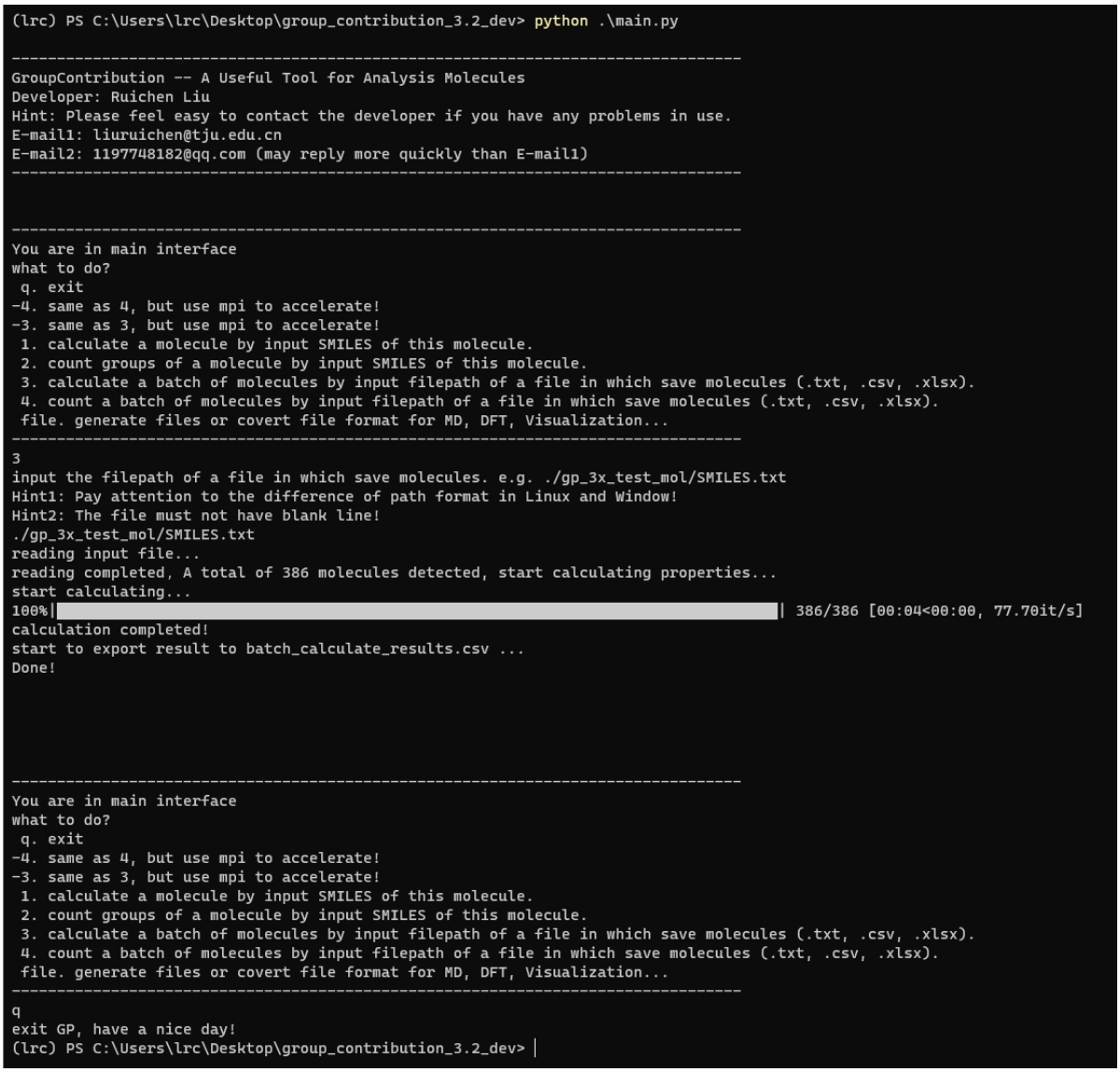


图2.6 GC程序的主功能3

当通过在主界面输入`3`进入主功能3（批量计算给定分子文件中的分子的理化性质），程序会提示用户键入保存了分子SMILES的文件路径。用户输入文件路径（特别提醒：Windows和Linux下路径的格式略有不同！分子文件中不要有空行！）然后敲击`Enter`后，程序会自动读取分子文件中记录的SMILES，然后进行计算。当计算完成后，结果将写进程序主目录下的`batch_results.csv`（自动生成），其格式与2.2.1.2中导出的文件是一样的。

2.2.1.5 基于MPI并行批量计算分子的理化性质(-3)

现代计算机的中央处理器（CPU）往往是多核的，若在批量计算分子性质时希望充分利用CPU的多核性能，可以使用MPI并计算。

若用户需要并行计算一批分子的理化性质，同2.2.1.4小节中一样，需要先准备一个2.1.2小节中介绍的分子文件，然后在主界面内输入`-3`，然后在键盘上敲击`Enter`，根据后续的提示输入所需指令即可，如图2.7所示。

图2.7 GC程序的主功能-3

当通过在主界面输入`-3`进入主功能-3（批量计算给定分子文件中的分子的理化性质），程序会提示用户键入保存了分子SMILES的文件路径。用户输入文件路径（特别提醒：Windows和Linux下路径的格式略有不同！分子文件中不要有空行！）然后敲击`Enter`后，程序会询问用户需要使用多少个进程（核）来并行计算，当用户输入想要调用的核数并敲击`Enter`后，程序会给出提示：**请键入q以优雅地退出程序，然后在终端输入下列命令**，用户需要根据所使用的系统选择输入哪条命令，以Windows为例，在终端输入`mpiexec -np 4 python .\gp_3x_mpirun.py -smiles_file_path ./gp_3x_test_mol/SMILES.txt -result_file_path mpi_batch_calculate_results.csv -task calculate`（**这里的命令只是例子，根据分子文件路径和想要调用的核数不同，屏幕上输出的命令也会有所不同，请根据实际情况随机应变**），会调用本程序中的`gp_3x_mpirun.py`模块进行并行计算，然后将结果写入生成在程序主目录的`mpi_batch_calculate_results.csv`中，其格式与2.2.1.2中导出的文件是一样的。

注意：基于MPI并行批量计算分子的理化性质的速度优势仅在要计算的分子非常多时（如上万）才能体现，这是因为MPI多进程计算时进程之间相互通信会增加耗时，若要计算的分子数目不多（如几百上千），那么并行计算所减少的耗时无法抵消进程间相互通信所增加的耗时。更具体的例子，可以参考表2.1

我们在 13th Gen Intel(R) Core(TM) i9-13900KF上对并行效率进行了测试，输入的分子文件使用的是Group Contribution内部数据文件夹中的`gdb.txt`(记录了大约三十万个饱和碳氢分子的SMILES)。结果如下：

表2.1 GC程序主功能4的并行效率测试

并行条件	耗时/s
single core	1471
mpi-4 core	1030
mpi-8 core	581
mpi-16 core	351

2.2.1.6 批量统计分子的基团数目(4)

若用户需要统计一批分子的基团数目（这在用户希望使用基团贡献法风格的分子指纹作为下游机器学习模型的输入时非常有用），需要先准备一个2.1.2小节中介绍的分子文件，然后在主界面内输入`4`，然后在键盘上敲击`Enter`，根据后续的提示输入所需指令即可，如图2.8所示。

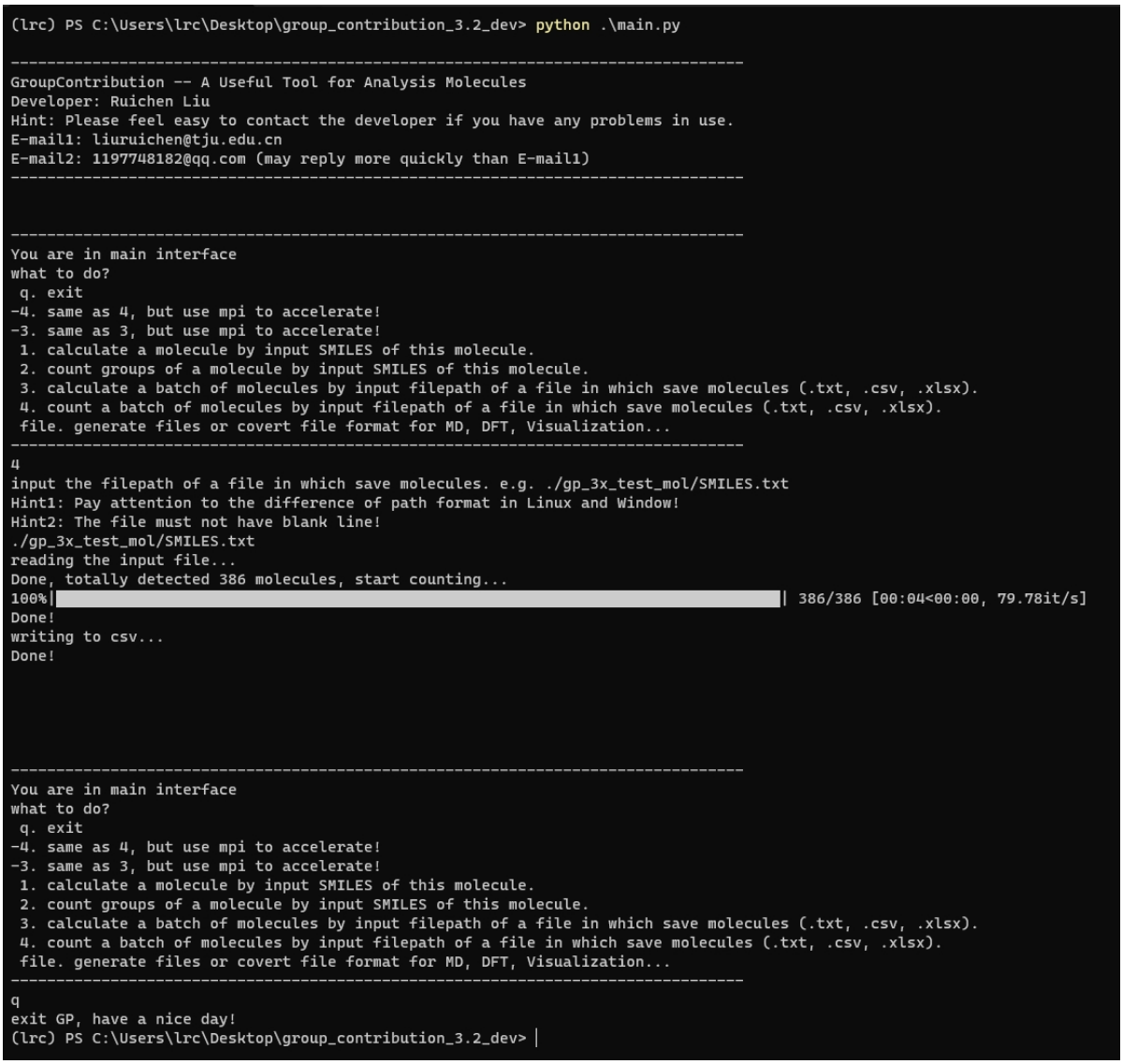


图2.8 GC程序的主功能4

当通过在主界面输入`4`进入主功能4（批量统计给定分子文件中的分子的基团数目），程序会提示用户键入保存了分子SMILES的文件路径。用户输入文件路径（特别提醒：Windows和Linux下路径的格式略有不同！分子文件中不要有空行！）然后敲击`Enter`后，程序会自动读取分子文件中记录的SMILES，然后进行计算。当计算完成后，结果将写进程序主目录下的`batch_count_result.csv`（自动生成），其格式与2.2.1.3中导出的文件是一样的。

2.2.1.7 基于MPI并行批量统计分子的基团数目(-4)

现代计算机的中央处理器（CPU）往往是多核的，若在批量统计分子基团数目时希望充分利用CPU的性能，可以使用MPI并计算。

若用户需要统计一批分子的基团数目（这在用户希望使用基团贡献法风格的分子指纹作为下游机器学习模型的输入时非常有用），需要先准备一个2.1.2小节中介绍的分子文件，然后在主界面内输入`-4`，然后在键盘上敲击`Enter`，根据后续的提示输入所需指令即可，如图2.9所示。

图2.9 GC程序的主功能-4

当通过在主界面输入`-4`进入主功能-4（批量统计给定分子文件中的分子的基团数目），程序会提示用户键入保存了分子SMILES的文件路径。用户输入文件路径（特别提醒：Windows和Linux下路径的格式略有不同！分子文件中不要有空行！）然后敲击`Enter`后，程序会询问用户需要使用多少个进程（核）来并行计算，当用户输入想要调用的核数并敲击`Enter`后，程序会给出提示：**请键入q以优雅地退出程序，然后在终端输入下列命令**，用户需要根据所使用的系统选择输入哪条命令，以Windows为例，在终端输入`mpiexec -np 4 python .\gp_3x_mpirun.py -smiles_file_path ./gp_3x_test_mol/SMILES.txt -result_file_path mpi_batch_count_results.csv -task count`（**这里的命令只是例子，根据分子文件路径和想要调用的核数不同，屏幕上输出的命令也会有所不同，请根据实际情况随机应变**），会调用本程序中的`gp_3x_mpirun.py`模块进行并行计算，然后将结果写入生成在程序主目录的`mpi_batch_count_results.csv`中，其格式与2.2.1.3中导出的文件是一样的。

注意：基于MPI并行批量计算分子的理化性质的速度优势仅在要计算的分子非常多时（如上万）才能体现，这是因为MPI多进程计算时进程之间相互通信会增加耗时，若要计算的分子数目不多（如几百上千），那么并行计算所减少的耗时无法抵消进程间相互通信所增加的耗时。更具体的例子，可以参考表2.1

2.2.1.8 与文件相关的操作(file)

基团贡献法虽然通用性强，计算速度快，但是其精度有限。因此，除了基团贡献法本身，本程序还向用户提供了一些用于可视化和生成分子动力学、量子化学计算（这在使用基团贡献法初筛，然后用高精度方法进一步筛选时非常有用）所需要的文件的功能。

2.2.1.8.1 基于SMILES生成给定分子的xyz文件(file_1)

xyz文件（http://sobereva.com/477）几乎是最简单的记录分子三维结构的文件格式，几乎所有的可视化程序都可以打开它（如gaussview、MS、VESTA、VMD等）

若用户希望通过输入给定分子的SMILES从而得到对应分子的xyz文件，只需在主界面内输入`file`，然后在键盘上敲击`Enter`，进一步输入`1`在键盘上敲击`Enter`。然后根据后续的提示输入所需指令即可，如图2.10所示。

图2.10 GC程序的主功能file_1

用户在输入目标分子的SMILES后，程序会要求用户指定输出的xyz文件的路径，若用户直接敲`Enter`，则将会在程序的主目录下输出与输入SMILES同名的xyz文件（**注意：SMILES语法中的一些符号如：/，#，：不能出现在文件名，请自行修改**）。成功生成xyz文件后，用户可以输入`0`返回至主界面，然后输入`q`优雅地退出。

2.2.1.8.2 批量生成xyz文件(file_2)

若用户需要生成一批分子的xyz文件，需要先准备一个2.1.2小节中介绍的分子文件，然后在主界面内输入`file`，然后在键盘上敲击`Enter`，然后输入`2`并敲击`Enter`，根据后续的提示输入所需指令即可，如图2.11所示。

```
(lrc) PS C:\Users\lrc\Desktop\group_contribution_3.2_dev> python .\main.py

--------------------------------------------------------------------------------
GroupContribution -- A Useful Tool for Analysis Molecules
Developer: Ruichen Liu
Hint: Please feel easy to contact the developer if you have any problems in use.
E-mail1: liuruichen@tju.edu.cn
E-mail2: 1197748182@qq.com (may reply more quickly than E-mail1)
--------------------------------------------------------------------------------



--------------------------------------------------------------------------------
You are in main interface
what to do?
 q. exit
-4. same as 4, but use mpi to accelerate!
-3. same as 3, but use mpi to accelerate!
 1. calculate a molecule by input SMILES of this molecule.
 2. count groups of a molecule by input SMILES of this molecule.
 3. calculate a batch of molecules by input filepath of a file in which save molecules (.txt, .csv, .xlsx).
 4. count a batch of molecules by input filepath of a file in which save molecules (.txt, .csv, .xlsx).
 file. generate files or covert file format for MD, DFT, Visualization...
--------------------------------------------------------------------------------
file

--------------------------------------------------------------------------------
You are in primary function file
what to do?
-6. same as 6, but use mpi to accelerate!
-2. same as 2, but use mpi to accelerate!
 0. return to main interface.
 1. generate .xyz file by input SMILES of a molecule.
 2. generate a batch of .xyz files by input filepath of a file in which save molecules (.txt, .csv, .xlsx).
 3. convert a file to other format (e.g. xyz, mol, mol2, pdb...)
 4. convert a batch of file to other format (e.g. xyz, mol, mol2, pdb...) by input a filepath list
 5. generate .gjf(input file of gaussian) file by input SMILES of a molecule.
 6. generate a batch of .gjf(input file of gaussian) files by input filepath of a file in which save molecules (.txt, .c
sv, .xlsx).
--------------------------------------------------------------------------------
2
input the filepath of a file in which save molecules. e.g. ./gp_3x_test_mol/SMILES.txt
Hint1: Pay attention to the difference of path format in Linux and Window!
Hint2: The file must not have blank line!
./gp_3x_test_mol/SMILES.txt
input the root path of output xyz files, that is, all the output xyz files will be make in this path. e.g. test_xyz
Hint1: Pay attention to the difference of path format in Linux and Window!
test_xyz
reading input file...
reading completed, A total of 386 molecules detected, start making xyz files...
xyz_root_path "test_xyz" has not been detected, I will create it for you
386it [00:02, 160.43it/s]
done! all .xyz files has been saved in test_xyz

--------------------------------------------------------------------------------
```

图2.11 GC程序的主功能file_2

当用户进入主功能file的子功能2时，程序要求用户输入记录了一批分子的SMILES的文件，在用户输入文件路径（特别提醒：Windows和Linux下路径的格式略有不同！分子文件中不要有空行！）然后敲击`Enter`后，程序会自动读取分子文件中记录的SMILES。然后程序要求用户输入保存被生成出的xyz文件的根目录（即本次任务所有生成的xyz文件都保存在那里），用户输入之后开始生成xyz文件。在生成结束后，程序会在主目录下产生两个文件，分别是记录成功生成xyz文件的SMILES（xyz_succeed.txt）和没有成功生成xyz文件的SMILES（xyz_fail.txt）。

2.2.1.8.3 基于MPI并行批量生成xyz文件(file_-2)

现代计算机的中央处理器（CPU）往往是多核的，若在批量生成xyz文件时希望充分利用CPU的性能，可以使用MPI并计算。

若用户需要生成一批分子的xyz文件，需要先准备一个2.1.2小节中介绍的分子文件，然后在主界面内输入`file`，然后在键盘上敲击`Enter`，在输入`-2`并敲击`Enter`，根据后续的提示输入所需指令即可，如图2.12所示。

图2.12 GC程序的主功能file_-2 (生成命令阶段)

程序会提示用户键入保存了分子SMILES的文件路径。用户输入文件路径（特别提醒：Windows和Linux下路径的格式略有不同！分子文件中不要有空行！）然后敲击`Enter`。之后程序要求用户输入保存被生成出的xyz文件的根目录（即本次任务所有生成的xyz文件都保存在那里），用户输入之后开始生成xyz文件。接着程序会询问用户需要使用多少个进程（核）来并行计算，当用户输入想要调用的核数并敲击`Enter`后，程序会给出提示：**请键入q以优雅地退出程序，然后在终端输入下列命令**（如图2.13所示），用户需要根据所使用的系统选择输入哪条命令，以Windows为例，在终端输入`mpiexec -np 4 python .\gp_3x_mpirun.py -smiles_file_path ./gp_3x_test_mol/SMILES.txt -out_root_path mpi_test_xyz -task xyz`（**这里的命令只是例子，根据分子文件路径和想要调用的核数不同，屏幕上输出的命令也会有所不同，请根据实际情况随机应变**），会调用本程序中的`gp_3x_mpirun.py`模块进行并行计算，然后将所有生成xyz文件生成在程序主目录的`mpi_test_xyz`目录中。在生成结束后，程序会在主目录下产生两个文件，分别是记录成功生成xyz文件的SMILES（xyz_succeed.txt）和没有成功生成xyz文件的SMILES（xyz_fail.txt）。

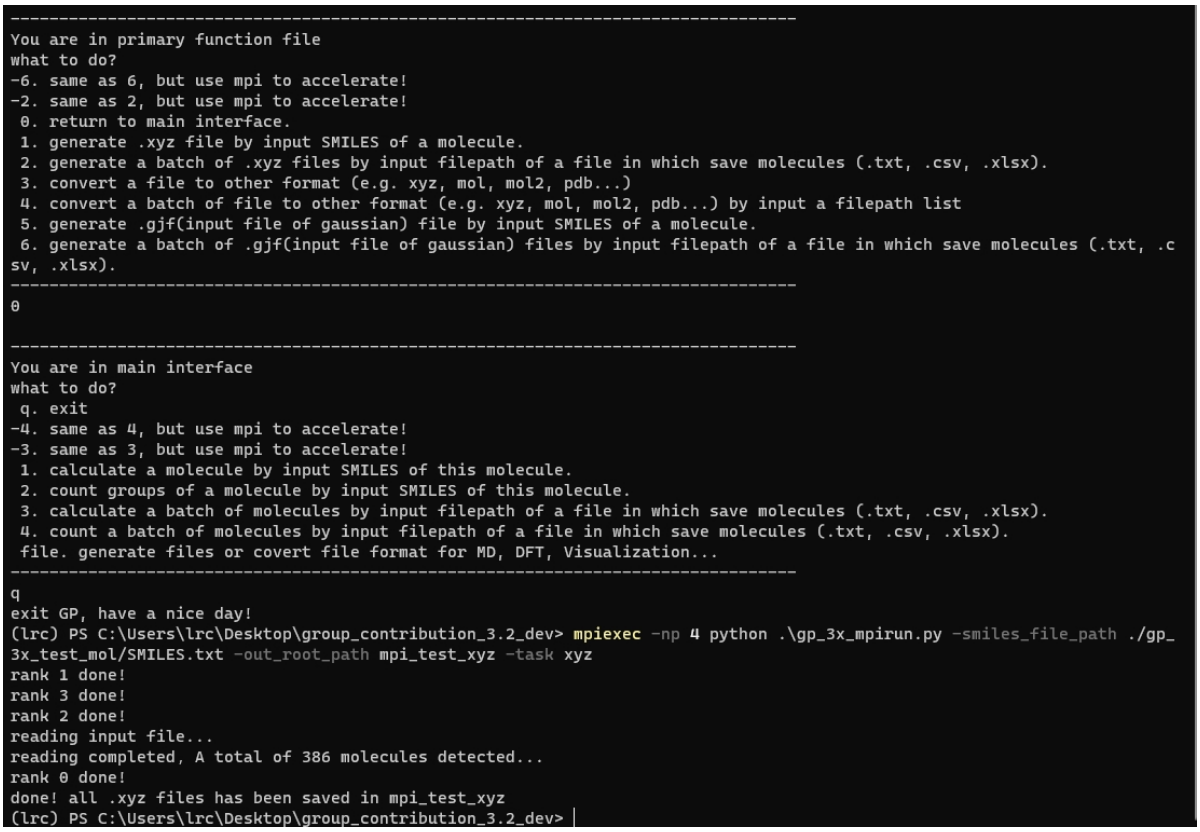


图2.13 GC程序的主功能file_-2 (运行命令阶段)

先输入`0`返回至主界面，然后再输入`q`优雅地退出程序。在终端输入刚才程序给我们生成的命令即可。

2.2.1.8.4 转化给定文件的文件格式(file_3)

本程序还提供了转换文件格式的功能，用户只需提供要被转化为原文件格式和路径，然后指定需要的文件格式和路径即可。**本程序此功能基于openbabel，因此但凡openbabel支持的文件格式，本功能都支持，如：xyz，mol，mol2，pdb...**

使用例子见图2.14。

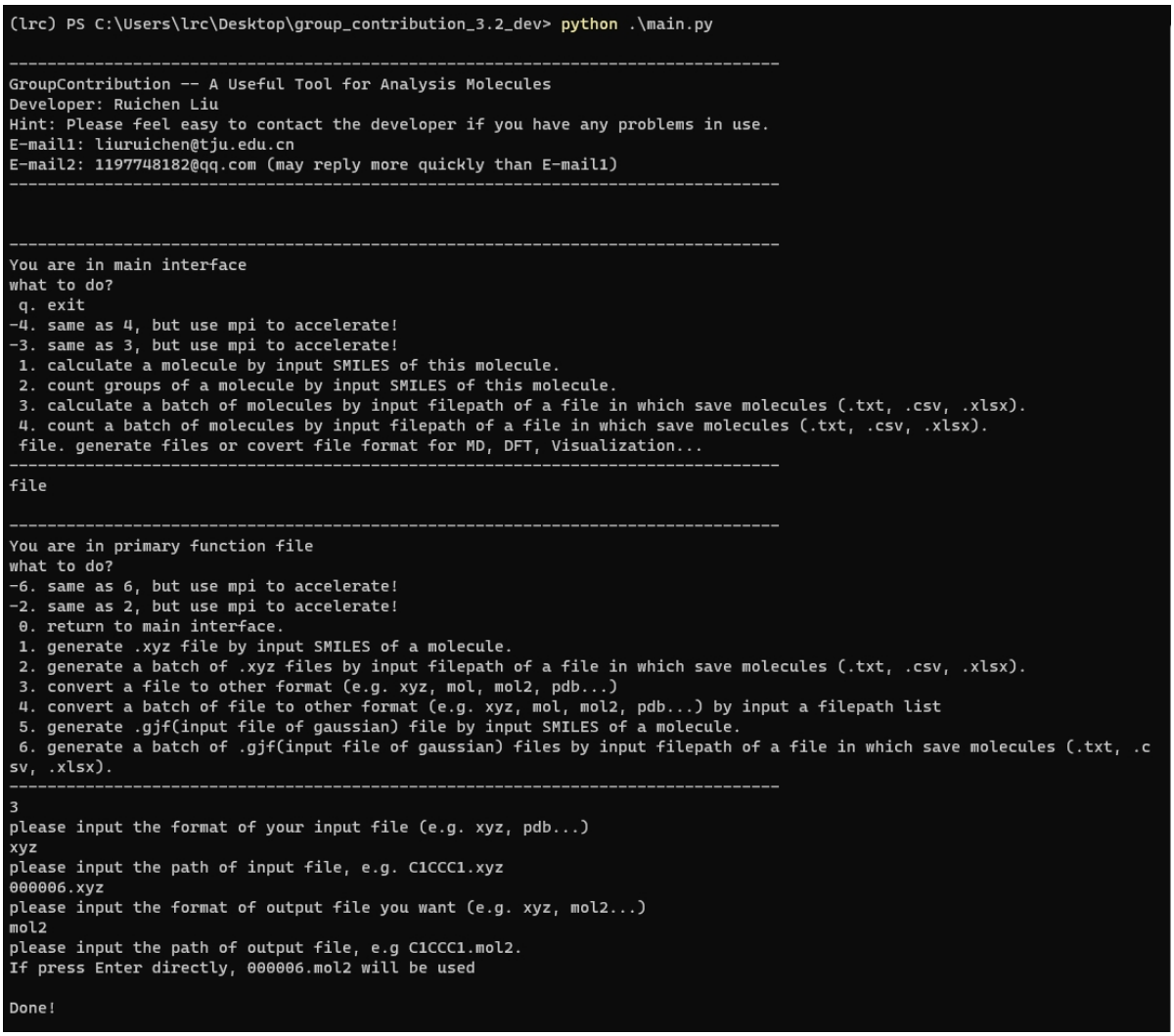


图2.14 GC程序的主功能file_3

注意，当指定目标文件路径时，若直接敲`Enter`，则会在程序主目录下生成与原文件同名的文件。

2.2.1.8.5 批量转化给定文件的文件格式(file_4)

当需要转化一批文件的文件格式时，首先应把待转换格式的文件放入同一个文件夹。然后进入主功能`file`的子功能`4`，首先程序会询问用户待转换的文件格式如何，然后还需输入待转换格式的文件的根目录（即保存了所有带转换格式的文件的目录）。接着程序问用户希望将文件转化为什么格式，用户输入之后（如xyz，mol，mol2，pdb，gro等），程序要求用户指定转化之后的新文件的根目录（即所有新产生的文件都将保存在那里），若直接敲击`Enter`，将会使用待转换格式的文件的根目录。具体操作见图2.15。

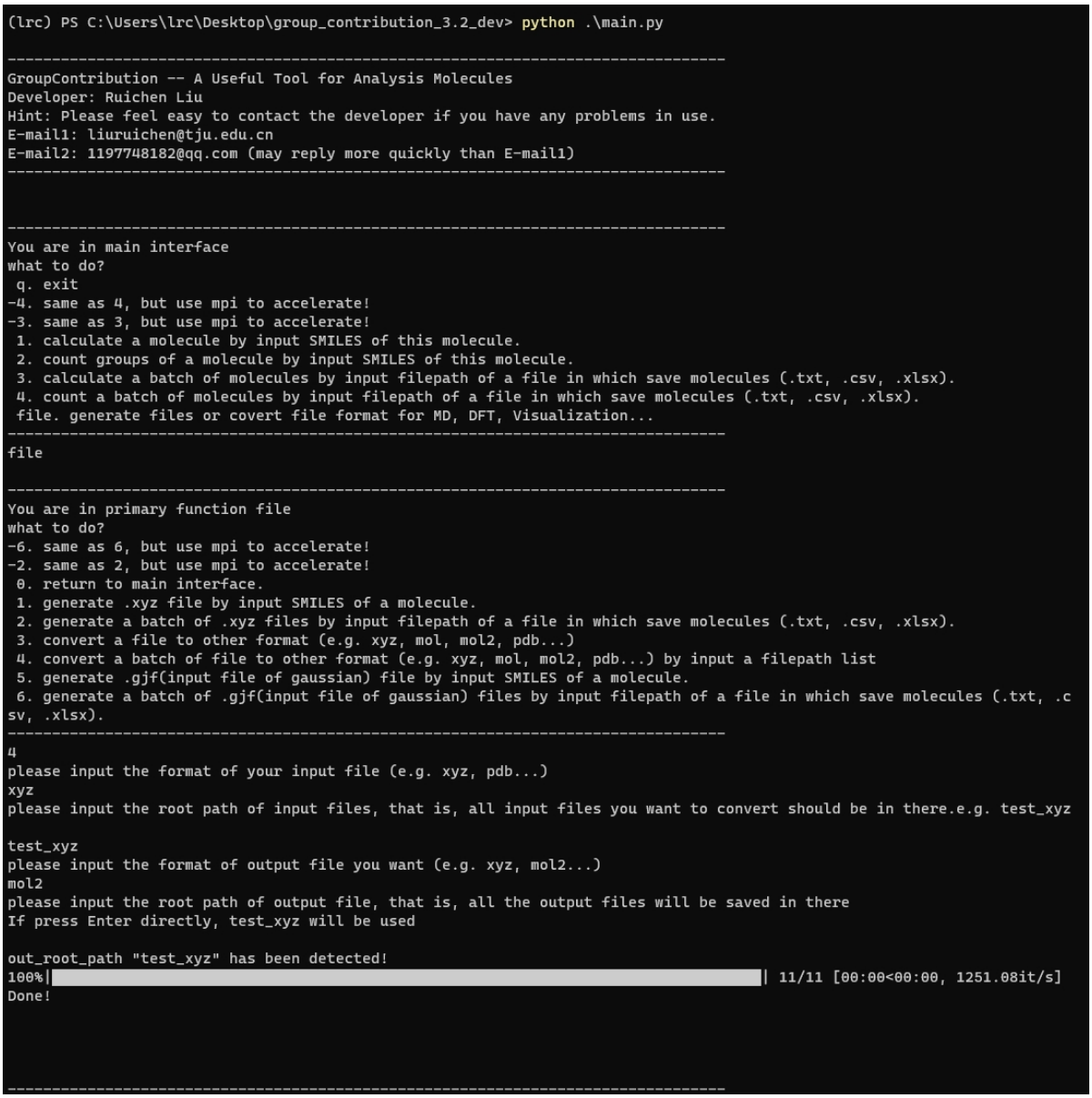


图2.15 GC程序的主功能file_4

2.2.1.8.6 基于SMILES生成gjf文件(file_5)

本程序还为用户提供了基于分子的SMILES生成对应的gjf文件（量子化学计算软件Gaussian的输入文件）。用户只需进入主功能file中的子功能5，然后根据提示操作即可，见图2.16。

图2.16 GC程序的主功能file_5

若用户对这里提供的接口（如高斯调用的CPU核数、内存数目、关键词、chk文件路径）不甚了解，可以参考量子化学软件Gaussian的用户手册或相关论文，也可联系开发者，或可得到一些帮助。

2.2.1.8.7 批量生成gjf文件(file_6)

还可以批量生成分子的gjf文件，程序要求用户输入记录了一批分子的SMILES的文件，在用户输入文件路径（特别提醒：Windows和Linux下路径的格式略有不同！分子文件中不要有空行！）然后敲击`Enter`后，程序会自动读取分子文件中记录的SMILES。然后程序要求用户输入保存被生成出的gjf文件的根目录（即本次任务所有生成的gjf文件都保存在那里），用户输入之后开始生成gjf文件。在生成结束后，程序会在主目录下产生两个文件，分别是记录成功生成gjf文件的SMILES（gjf_succeed.txt）和没有成功生成gjf文件的SMILES（gjf_fail.txt）。具体操作步骤见图2.17。

图2.17 GC程序的主功能file_6

2.2.1.8.8 基于MPI并行批量生成生成gjf文件(file_-6)

现代计算机的中央处理器（CPU）往往是多核的，若在批量生成gjf文件时希望充分利用CPU的性能，可以使用MPI并计算。

若用户需要生成一批分子的gjf文件，需要先准备一个2.1.2小节中介绍的分子文件，然后在主界面内输入`file`，然后在键盘上敲击`Enter`，在输入`-6`并敲击`Enter`，根据后续的提示输入所需指令即可，如图2.18所示。

图2.18 GC程序的主功能file_-6 （生成命令阶段）

图2.19 GC程序的主功能file_-6 （执行命令阶段）

先输入`0`返回至主界面，然后再输入`q`优雅地退出程序。在终端输入刚才程序给我们生成的命令即可。

2.2.2 GC作为外部库

本程序还可作为外部库导入进用户自行编写的python脚本中，下面分别介绍。

2.2.2.1 计算单个分子的理化性质

只需按如下方式编写python脚本即可：

```python
from gp_3x_calculator import Calculator  # 从Group Contribution导入计算器


calculator = Calculator()  # 实例化一个Calculator对象
print(calculator.calculate_a_mol('CC1(C2)CC(C3)CC2CC3(C)C1', debug=True))  # 打印计算结果，这里设置debug=True是要求程序同时打印基团数目
```

2.2.2.2 统计单个分子的基团数目

只需按如下方式编写python脚本即可：

```python
from gp_3x_counter import Counter  # 从Group Contribution导入统计器


counter = Counter()  # 实例化一个Counter对象
result = c.count_a_mol(m, clear_mode=True)  # 统计结果，使用清爽模式
print(result)  # 打印计算结果
```

2.2.2.3 批量计算分子

我们这里以Group Contribution内部数据文件夹中的`gdb.txt`(记录了大约三十万个饱和碳氢分子的SMILES)为例，对其进行批量计算。假定在当前工作目录输出名为`test.csv`的结果文件。

```
import os
from gp_3x_calculator import Calculator  # 从Group Contribution导入计算器


calculator = Calculator()  # 实例化一个Calculator对象
input_file_path = os.path.join('gp_3x_internal_data', 'gdb.txt')  # 说明输入文件路径
output_file_path = os.path.join('test.csv')  # 说明输出文件路径
calculator.calculate_mols(input_file_path, output_file_path)  # 开始计算
```

2.2.2.4 基于MPI并行批量计算分子

除了单核运行模型外，Group Contribution还支持多核并行批量计算分子性质。我们给出名为`gp_3x_mpirun.py`的运行脚本

以Group Contribution内部数据文件夹中的`gdb.txt`(记录了大约三十万个饱和碳氢分子的SMILES)为例，对其进行批量计算，且在当前工作目录输出名为`mpi_batch_calculate_results.csv`的结果文件。

必须在终端输入以下命令（强烈建议在Linux系统中运行）：

```
# 使用8个进程并行计算
# 注意Windows和Linux路径的区别
mpirun -np 8 python ./gp_3x_mpirun.py -smiles_file_path ./gp_3x_internal_data/gdb.txt -result_file_path mpi_batch_calculate_results.csv -task calculate  # Linux系统中输入这一行
mpiexec -np 8 python .\gp_3x_mpirun.py -smiles_file_path .\gp_3x_internal_data\gdb.txt -result_file_path mpi_batch_calculate_results.csv -task calculate  # Windows系统中输入这一行
```

程序将会在当前目录输出名为`mpi_batch_calculate_results.csv`的结果文件。

2.2.2.5 批量统计分子基团数目

我们这里以Group Contribution内部数据文件夹中的`SMILES.txt`(记录了几百个各类分子的SMILES)为例，对其进行批量计算。假定在当前工作目录输出名为`count_result.csv`的结果文件。

```python
import os
from gp_3x_counter import Counter  # 从Group Contribution导入统计器


c = Counter()  # 实例化一个Counter对象
c.count_mols(smiles_file_path=os.path.join('gp_3x_test_mol', 'SMILES.txt'),
             count_result_file_path='count_result.csv',
             add_note=True,
             add_smiles=True
            )  # 批量统计分子基团数目
```

2.2.2.6 基于MPI并行批量统计分子基团数目

除了单核运行模型外，Group Contribution还支持多核并行批量计算分子性质。我们给出名为`gp_3x_mpirun.py`的运行脚本

以Group Contribution内部数据文件夹中的`gdb.txt`(记录了大约三十万个饱和碳氢分子的SMILES)为例，对其进行批量计算，且在当前工作目录输出名为`result_mpi.csv`的结果文件。

必须在终端输入以下命令（强烈建议在Linux系统中运行）：

```
# 使用8个进程并行计算
# 注意Windows和Linux路径的区别
mpirun -np 8 python ./gp_3x_mpirun.py -smiles_file_path
./gp_3x_internal_data/gdb.txt -result_file_path mpi_batch_count_results.csv -
task count  # Linux系统中输入这一行
mpiexec -np 8 python .\gp_3x_mpirun.py -smiles_file_path
.\gp_3x_internal_data\gdb.txt -result_file_path mpi_batch_count_results.csv -
task count  # Windows系统中输入这一行
```

程序将会在当前目录输出名为`mpi_batch_count_results.csv`的结果文件。

2.2.2.7 与文件相关的操作

2.2.2.7.1 基于SMILES生成给定分子的xyz文件

```python
from gp_3x_tool import Tool  # 从Group Contribution中导入工具箱Tool


t = Tool()  # 实例化一个Tool对象
t.smi_to_xyz('C1CCCC1', xyz_path='test.xyz')  # 生成C1CCCC1的xyz文件，路径为
test.xyz
```

2.2.2.7.2 批量生成xyz文件

```
from gp_3x_tool import Tool  # 从Group Contribution中导入工具箱Tool


t = Tool()  # 实例化一个Tool对象
t.batch_smi_to_xyz(smiles_file_path='SMILES.txt', xyz_root_path='test_xyz')  # 批量生成xyz文件，生成的xyz都保存在test_xyz中
```

2.2.2.7.3 基于MPI并行批量生成xyz文件

除了单核运行模型外，Group Contribution还支持多核并行批量生成xyz文件。我们给出名为`gp_3x_mpirun.py`的运行脚本

以Group Contribution内部数据文件夹中的`SMILES.txt`(记录了几百个各类分子的SMILES)为例，批量计算xyz文件，且在当前工作目录中的mpi_test_xyz目录下输出所有xyz文件。

必须在终端输入以下命令（强烈建议在Linux系统中运行）：

```
# 使用8个进程并行计算
# 注意Windows和Linux路径的区别
mpirun -np 8 python .\gp_3x_mpirun.py -smiles_file_path ./gp_3x_test_mol/SMILES.txt -out_root_path mpi_test_xyz -task xyz  # Linux系统中输入这一行
mpiexec -np 8 python .\gp_3x_mpirun.py -smiles_file_path ./gp_3x_test_mol/SMILES.txt -out_root_path mpi_test_xyz -task xyz  # Windows系统中输入这一行
```

2.2.2.7.4 转化给定文件的文件格式

```
from gp_3x_tool import Tool  # 从Group Contribution中导入工具箱Tool


t = Tool()  # 实例化一个Tool对象
t.convert_file_type(in_format='xyz', in_path='in.xyz',
                    out_format='mol2', out_path='out.mol2')
```

2.2.2.7.5 批量转化给定文件的文件格式

```
from gp_3x_tool import Tool  # 从Group Contribution中导入工具箱Tool


t = Tool()  # 实例化一个Tool对象
t.batch_convert_file_type(in_format='xyz', in_root_path='in_xyz',
                          out_format='mol2', out_root_path='out_mol2')
```

2.2.2.7.6 基于SMILES生成gjf文件

```
from gp_3x_tool import Tool  # 从Group Contribution中导入工具箱Tool


t = Tool()  # 实例化一个Tool对象
t.smi_to_gjf(smi='C1CCC1', nproc='12', mem='12GB',
             gaussian_keywords='#p opt freq b3lyp/6-31g*',
             chk_path='C1CCC1.chk',
             gjf_path='C1CCC1.gjf',
             add_other_std_tasks=False)
```

2.2.2.7.7 批量生成gjf文件

```
from gp_3x_tool import Tool  # 从Group Contribution中导入工具箱Tool


t = Tool()  # 实例化一个Tool对象
t.batch_smi_to_gjf(smiles_file_path='SMILES.txt', gjf_root_path='test_gjf',
                   nproc='12', mem='12GB',
                   gaussian_keywords='#p opt freq b3lyp/6-31g*',
                   add_other_std_tasks=False)
```

2.2.2.7.8 基于MPI并行批量生成生成gjf文件

除了单核运行模型外，Group Contribution还支持多核并行批量生成gjf文件。我们给出名为`gp_3x_mpirun.py`的运行脚本

以Group Contribution内部数据文件夹中的`SMILES.txt`(记录了几百个各类分子的SMILES)为例，批量计算gjf文件，且在当前工作目录中的mpi_test_gjf目录下输出所有gjf文件。

必须在终端输入以下命令（强烈建议在Linux系统中运行）：

```
# 使用8个进程并行计算
# 注意Windows和Linux路径的区别
mpirun -np 8 python gp_3x_mpirun.py -smiles_file_path
./gp_3x_test_mol/SMILES.txt -out_root_path mpi_test_gjf -nproc 12 -mem 12GB -
gaussian_keywords "#p opt freq b3lyp/6-31g*" --add_other_std_tasks -task gjf
 # Linux系统中输入这一行
mpiexec -np 8 python gp_3x_mpirun.py -smiles_file_path
./gp_3x_test_mol/SMILES.txt -out_root_path mpi_test_gjf -nproc 12 -mem 12GB -
gaussian_keywords "#p opt freq b3lyp/6-31g*" --add_other_std_tasks -task gjf
 # Windows系统中输入这一行
```

3 高级

3.1 导出基团贡献法风格的分子指纹

除了计算分子性质，Group Contribution还支持直接导出分子不同基团数目的统计结果，即输出基团贡献法风格的分子指纹。

```
from gp_3x_counter import Counter  # 从Group Contribution中导入计数器Counter


c = Counter()  # 实例化一个Counter对象
result = c.count_a_mol(m, clear_mode=False)  # 输出基团数目的统计结果。clear_mode=False是指不省略数目为0的基团
print(result)

print(c.get_group_fingerprint(m))  # 直接输出列表形式的统计结果，如：[1,2,1,0,0,5.....]。该列表的长度为所有基团的种类数（220 + 130 + 74）
```

3.1.1 基于基团贡献法风格的分子指纹建立机器学习模型（todo）

3.2 重新拟合基团参数（todo）