RUNNER v.1

Краткое руководство пользователя

Назначение программы

RUNNER – программа для глобального поиска оптимальной структуры атомных кластеров и структуры их низколежащих изомеров, в ходе которой энергия кластеров оценивается на основе квантовохимического расчета (в т.ч. DFT и *ab initio*) или расчетов с использованием классических межатомных потенциалов или силовых полей.

RUNNER выполняет собственно поиск глобального минимума, используя специализированные программы-серверы для расчета энергии и локальной оптимизации геометрии кластеров. Серверами для вычисления энергии и локальной оптимизации в настоящее время могут быть программы:

- Gaussian 03/09/16
- NWChem v. 6 / 7
- GLOBUS v.1

Метод глобального поиска, реализованный в программе, представляет собой эволюционный (генетический) алгоритм в комбинации с табу-поиском. Подробное описание реализованного в программе алгоритма глобальной оптимизации дано в разделе *Алгоритм глобальной оптимизации, реализованный в RUNNER.*

Получение и установка программы

Программа, управляющий скрипт и примеры входных файлов могут быть загружены с ресурса https://github.com/skignatov/runner-release

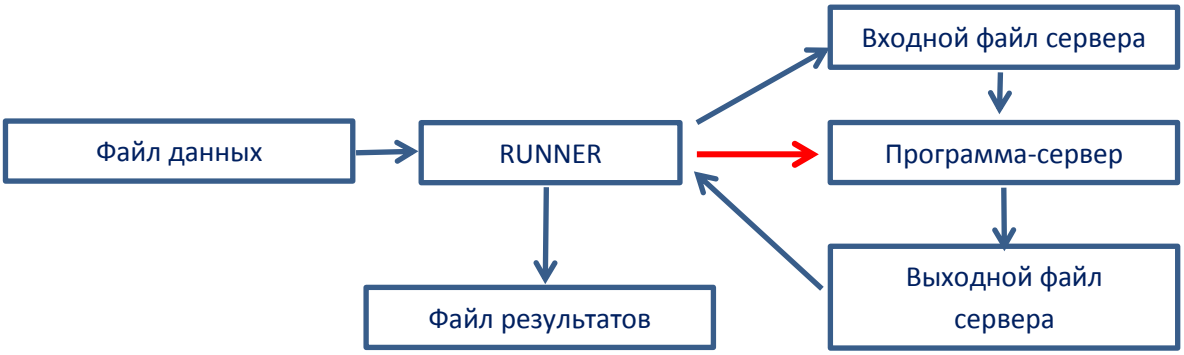
Для начала работы достаточно на своем компьютере создать отдельную директорию, поместить в нее программу, управляющий скрипт (при его необходимости, см. раздел *Режимы работы программы*) и файлы данных, необходимые при выбранном режиме работы (см. раздел *Файлы, необходимые для запуска*). Кроме того, на используемом компьютере (или компьютерах, играющих роль удаленных вычислительных узлов) должна быть также правильно установлена выбранная программа-сервер (получается и устанавливается отдельно). Команды запуска RUNNER описаны в разделе *Запуск программы*.

Режимы работы программы

Программа может работать в нескольких режимах:

1. *Режим прямого вызова* (Windows). В этом режиме программа читает основной файл входных данных, последовательно формирует файлы исходных данных для

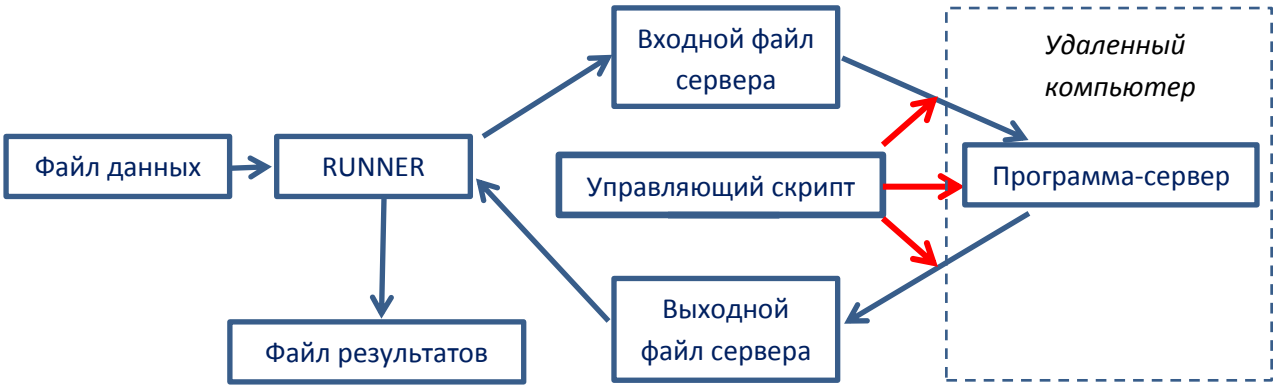
программы-сервера и запускает программу-сервер на используемом компьютере для расчета со сформированным файлом. После окончания расчета, выходной файл программы-сервера анализируется, из него извлекается информация о структуре и энергии кластера:



Режим прямого вызова реализует только последовательный расчет исследуемых структур. Его рекомендуется применять только для тех расчетов, в которых расчет энергии занимает небольшое время (например, расчет на основе эмпирических потенциалов и силовых полей программой GLOBUS, квнтовохимический расчет полуэмпирическим методом). Этот режим апробирован только в среде Windows.

2. *Режим SSH* (LINUX). В этом режиме программа читает файл входных данных, формирует файлы исходных данных для программы-сервера (параллельно несколько файлов, число которых контролируется командой *MaxJobs*). Генерируемые файлы помещаются в директорию ./inp , которая автоматически создается в текущей директории. Одновременно работающий управляющий скрипт (УС) *runner_ssh_<version>*.sh читает файлы в директории *./inp* и запускает расчеты сервера на одном или нескольких удаленных компьютерах (узлах), которые связаны с центральным узлом по протоколу *ssh*. Центральный компьютер также может выступать в качестве удаленного узла, т.е. задачи могут быть запущены и на нем. При успешном запуске данного файла УС перемещает его из ./inp в директорию ./out, одновременно создавая копию в директории ./sub. Файлы в этой директории ./sub получают название <file>.<ext>*_X*, где *X* – номер узла, на котором запущен расчет. УС циклически проверяет состояние расчетов на удаленных узлах и при завершении расчета (успешном или аварийном) скачивает файл результатов в директорию ./out, удаляя соответствующий файл из ./sub. RUNNER, обнаруживая новый выходной файл в директории ./out, анализирует его, извлекая оптимизированные структуры и энергии и перемещает входной и выходной файлы из ./out в директорию ./arc. В зависимости от причины завершения данного расчета, RUNNER формирует новый входной файл для продолжения незавершенного расчета или переходит к генерации новой точки для глобального поиска. Генерируемые файлы имеют формат имени geo-XXXXXX-YY.<ext>, где XXXXXX – номер точки глобального поиска, YY – номер продолжения незавершенного расчета (YY=00 для нового файла). Список узлов, на которых могут запускаться программы серверы и выполняться расчеты содержится в файле *nodes.txt*, который должен присутствовать в директории запуска. УС может

запускаться и останавливаться вручную независимо от RUNNER (до или после него), либо может быть запущен, перезапущен (в случае аварийного завершения УС) и остановлен программой автоматически (см. команду *ShellControl*). Режим SSH рекомендуется для задач с длительным расчетом энергии (DFT, *ab initio*) на рабочих станциях, которые не объединены в кластер. Поскольку квантовохимические расчеты выполняются на удаленных узлах параллельно, то увеличение числа доступных узлов увеличивает общую производительность ГО.



3. *Режим BATCH, пакетный режим.* (LINUX на суперкомпьютерах или кластерах). В этом режиме работа полностью аналогична режиму SSH, однако управляющий скрипт *runner_batch_<version>.sh* запускает задачи на суперкомпьютере (СК) или кластере с помощью планировщика задач – *slurm* или *pbs.* Управляющий скрипт для *pbs* отличается от скрипта *slurm* и называется *runner_batch_pbs_<version>.sh.* В отличие от режима *ssh*, в *batch*-режиме не требуется файл *nodes.txt*, однако требуются два других файла: файл *workdirs.txt,* который содержит список и описание рабочих директорий, где будут появляться результаты отдельных задач, и файлы типа *batch.template,* являющиеся формами для формирования скриптов пакетного запуска расчетных заданий, сформированными по правилам используемого СК. Файлы типа *batch.template* могут быть различными и/или иметь различные имена для каждой рабочей директории. Их соответствующие имена указываются в *workdirs.txt* для каждой рабочей директории. При работе в этом режиме в директории запуска программы создаются одна или несколько директорий (их имена описываются в *workdirs.txt* , обычно *wd0, wd1,wd2 ...*) так, что в каждой директории параллельно запускаются задачи для расчета отдельных точек глобального поиска. В этих директориях содержатся также выходные файлы данного расчета и все временные файлы программы-сервера. УС анализирует файлы в рабочих директориях и при завершении расчета перемещает их в директорию ./out. После этого в данной рабочей директории запускается новая задача, входной файл которой берется из директории ./inp. Этот режим работы является основным и наиболее удобным при расчетах на СК. Увеличение числа рабочих директорий приводит к ускорению расчета только в определенных пределах, поскольку большое количество одновременно запущенных задач может привести к тому, что часть из них будет стоять в очереди или будут заблокированы планировщиком из-за превышения лимита задач для одного пользователя. Часто оптимальным является 8-12 рабочих директорий для одной системы.

4. *Режим CRON, пакетный режим с периодическим запуском УС*. (LINUX на суперкомпьютерах). На некоторых СК администратором запрещена непрерывная работа УС на центральном узле. Для работы на таких компьютерах используется режим CRON, который полностью аналогичен режиму BATCH, однако УС и программа RUNNER не работают непрерывно, а запускаются периодически по команде *cron* с периодом в несколько минут. Настройка *cron* производится самим УС при первоначальном запуске. Этот режим менее устойчив, чем режим BATCH и его рекомендуется применять только в случае строгих ограничений на работу УС.

Файлы, необходимые для запуска

Здесь и далее в качестве примера рассматривается глобальная оптимизация кластера Mg_{10}. Во всех примерах в качестве названия файла приведено *mg10*. При работе это имя следует заменить на имя файла, соответствующее конкретной задаче.

Названия файлов и другие параметры, указанные в <…> могут варьироваться по желанию пользователя. Файлы без таких скобок должны иметь названия, точно соответствующие указанным в таблице.

Все файлы данных должны находиться в директории запуска. Программы могут находиться в директории, к которым задан путь поиска.

1. *Режим прямого вызова*

Файл	**Назначение**
runner.exe (Windows) *runner.x* (LINUX)	Программа RUNNER
<mg10>.inp	Основной файл исходных данных. Описывает оптимизируемую систему и параметры оптимизации
<mg.gj0>	Форма (template) файла данных программы-сервера

2. *Режим SSH*

Файл	**Назначение**
runner.x	Программа RUNNER
runner_ssh_<version>.sh	Управляющий скрипт
<mg10>.inp	Основной файл исходных данных. Описывает оптимизируемую систему и параметры оптимизации
<mg.gj0>	Форма (template) файла данных программы-сервера
nodes.txt	Файл, описывающий *ssh*-узлы, на которых будут работать программы-серверы.

3. *Режим BATCH*

Файл	**Назначение**
runner.x	Программа RUNNER

runner_batch_<version>.sh	Управляющий скрипт (планировщик *slurm*). В случае планировщика *pbs* название скрипта *runner_batch_pbs<_version>.sh*
<mg10>.inp	Основной файл исходных данных. Описывает оптимизируемую систему и параметры оптимизации
<mg.gj0>	Форма (template) файла данных программы-сервера
workdirs.txt	Файл, описывающий директории, в которых будут запускаться отдельные задания. Для каждой директории указываются ее имя, активна ли она (участвует ли в работе в данное время) и имя формы скрипта запуска задания.
<batch.template>	Форма (template) скрипта запуска заданий. Таких файлов может быть несколько, они могут иметь разные имена. Эти файлы назначаются отдельным рабочим директориям в файле *workdirs.txt*

4. *Режим CRON*
   Все файлы полностью идентичны режиму BATCH, за исключением управляющего скрипта, который в данном случае называется *runner_batch_cron<_version>.sh* (планировщик *slurm*).

Файлы данных, необходимые для работы RUNNER

1. *Основной файл входных данных программы RUNNER <mg10>.inp*

Основной файл входных данных программы RUNNER представляет собой текстовый файл в формате ASCII. Ключевые слова располагаются на строках, первый непустой символ которых #. Пустые строки и строки, не имеющие символа # в начале, игнорируются. Символ ! означает начало комментария, он может располагаться в любом месте строки. Вся информация на данной строке справа от ! считается комментарием.

Порядок ключевых слов, как на строке, так и на разных строках, может быть любым. Большие и малые буквы не различаются (за исключением символов химических элементов в команде Stoichiometry)

Основные команды и ключевые слова (значения в скобках <…> являются примерами):

Ключевое слово	**Значение**	**По умолчанию**
Stoichiometry=<*Mg10*>	Задает стехиометрию оптимизируемого кластера. Символы химического элемента должны быть записаны так, как даны в таблице Менделеева. Большие и малые буквы различаются, поэтому записывать mg10 или MG10 неправильно, правильно только Mg10.	Нет значения по умолчанию, команда должна присутствовать всегда
Nstruct=<100>	Количество циклов глобального поиска, т.е. количество сгенерированных и оптимизированных структур. Оптимизированными считаются только те	100

	структуры, оптимизация которых успешно достигла критериев останова. Общее число оптимизаций, часть которых не дошла до конца или завершилась аварийно, может превышать *Nstruct*.	
MaxCyc=<10000>	Число рабочих циклов, производимых программой RUNNER при просмотре рабочих директорий. Предотвращает бесконечное зацикливание, если число оптимизированных структур по каким-то причинам не может достигнуть *Nstruct*. Максимальное время работы программы равно *MaxCyc*IdleTime*	100000
IdleTime=<120> Idle=<120>	Время ожидания в конце каждого рабочего цикла, в секундах	300
Server=<g09>	Тип программы-сервера, который проводит локальную оптимизацию.	g09
FilPat=<*mg.gj0*>	Имя файла-формы (template) входных данных для программы-сервера. На месте размещения геометрии должна помещаться метка %GEO%. Если сервером является Gaussian, должен присутствовать параметр %chk=…	Нет значения по умолчанию
Seed=(<12345>,<67890>)	Зерно для инициализации генератора случайных чисел. При отсутствии команды *Seed* будут использованы внутренние значения по умолчанию генератора. Используется для повторения результатов предыдущего расчета.	Внутренние значения по умолчанию генератора случайных чисел
MaxPool=<30>	Максимальный размер пула. При его превышении находимые локальные минимумы, энергия которых превышает значения всех структур пула, не могут участвовать в генерации новых структур.	50
ActivePool=<0.5>	Активная часть пула, участвующая в генерации новых структур.	0.7
MinPool=<5>	Минимальный размер пула, при котором начинается основная фаза ГО	10
MaxRestarts=<20>	Максимальное число рестартов локальной оптимизации (ЛО) программы-сервера, т.е. запусков ЛО, оборвавшихся по причине превышения числа циклов, внутренних ошибок и т.д. Введено, чтобы исключить бесконечную работу сервера при повторении одной и той же ошибки. Если число рестартов превышено, структура помещается в директорию ./hlp. Такая структура не учитывается в *Nstruct*.	10
NoSaveArchive	Отключить сохранение результатов работы программы-сервера в архиве. В обычном режиме работы, при каждом завершении	отсутствует (результаты сохраняются в

	ЛО программой-сервером, входной и выходной файлы этой оптимизации (например, *.gjf и *.log) отправляются в директорию ./arc (архив результатов оптимизации). Если файлы очень большие, а пространство на диске ограничено, команда NoSaveArchive отменяет это сохранение, экономя место на диске.	директории ./arc)
BondLimits=(<1.>,<4>)	Пределы межатомных расстояний в Å, при которых атомы считаются связанными. Используется при генерации новых структур операторами генерации.	1 Å – 4 Å
Sphere=7.	Радиус сферы в Å, в которой генерируются атомы оператором RAND	7 Å
MaxTry=10000	Максимальное число попыток генерации кластера, в котором длины связи удовлетворяют BondLimits	5000
Init=(<RAND>,<RND1>) Init=(<RAND=30,RND1=70)	Операторы, применяемые для генерации начальной популяции на начальной фазе оптимизации (Initial Phase). Числа после = показывают вероятность использования данного оператора. Если вероятности не указаны, они считаются равными (т.е. для двух опереаторов – 50% и 50%). Список доступных операторов см. в разделе «Алгоритм оптимизации».	RAND, RND1
Operators=(<PCCR=50, RAND=10,RND1=10,TPCR=20,IMOV=10>)	Операторы, применяемые для генерации начальной популяции на главной фазе оптимизации (Main Production Phase). Числа после = показывают вероятность использования данного оператора (в %, автоматически нормируются к 100). Если вероятности не указаны, они считаются равными для всех операторов, у которых вероятности не указаны. Список доступных операторов см. в разделе «Алгоритм оптимизации».	PCCR=50, RAND=10,RND1=10,TPCR=20,IMOV=10
Continue	Продолжить завершенную ранее оптимизацию (т.е. оптимизацию, которая достигла предельного значения *Nstruct*). Для продолжения будут использованы файлы *.base_, *pool_, *.sbas_, *.info_ предыдущей оптимизации, они должны находиться в директории запуска. При наличии команды Continue параметр *Nstruct* рассматривается как дополнительное количество структур, которые должны быть исследованы в ходе продолжающейся оптимизации.	не присутствует
ShellControl= (*StartOnBeg,StartIfDown, StopOnFinal ,StopOnExit*)	Управление запуском, перезапуском и остановкой УС. *StartOnBeg* – запустить УС в начале работы	При отсутствии любого ключевого слова

	программы. *StartIfDown* – запустить УС если он перестал работать. *StopOnFinal* – остановить УС после завершения оптимизации *StopOnExit* – остановить УС при любом завершении работы программы	(или всей команды) его действие не выполняется.

В этом и последующих разделах метки и ключевые слова, показанные красным в примерах, обязательно должны присутствовать в файлах. Остальные ключевые слова и команды – опционально.

Пример файла <Mg10>.inp:

```
# Stoichiometry=Mg10   Nstruct=300
# Server=g09   FilPat=mg.gj0   Idle=60  NoAutoStart
!# SEED=(674127515,48622252)

# MaxPool=50 ActivePool=0.7 MinPool=10 MaxRestarts=10  !NoSaveArchive
# BondLIMIts=(1.,4.) Sphere=6.  MaxTry=5000

# Init=(RND1=10,RAND=10)
# Operators=(PCCR=50,RAND=10,RND1=10,TPCR=20,IMOV=10)
```

2. *Template-файл программы-сервера <mg.gj0>*

Файл-форма для создания входного файла программы-сервера представляет собой несколько измененный обычный файл этой программы, в котором место, где размещаются декартовы координаты, отмечено меткой %GEO%. Кроме того, в файлах для программы Gaussian должна присутствовать ключевое слово %chk=TTT (TTT – произвольный текст). В процессе работы текст TTT будет автоматически заменяться на имя временного файла geo-XXXXX-YY.chk, где XXXXXX и YY соответствуют номеру структуры и номеру рестарта при ее оптимизации.

Пример:

```
%Nproc=4
%Mem=6000MB
%chk=geo.chk      ← обязательное ключевое слово %chk
%NoSave
# bp86/6-31G(d,p) Fopt=(MaxCycle=300)  SCF=(xqc,MaxCycle=300)

gjf-pattern file

0 1
%GEO%             ← обязательная метка %GEO% координат атомов
```

3. *Файл nodes.txt*

Файл представляет собой текстовый ASCII-файл, в котором каждая строка описывает узел, где работает программа-сервер. Пустые строки и строки, начинающиеся с символа #, игнорируются. Часть строки после # рассматривается как комментарий.

Каждая строка имеет формат:

<IP > <Active> <User> <Nproc> <Mem> <Prog> <Dir>

IP	IP-адрес узла, по которому узел доступен через *ssh*
Active	Будет ли узел использоваться в текущей оптимизации (0 или 1)
User	Username для доступа по ssh
Nproc	Число ядер, которые можно задействовать при расчете на данном узле
Mem	Объем памяти в МБ, которые можно задействовать на данном узле
Prog	Команда запуска программы сервера
Dir	Директория, в которой будет проводиться расчет

Пример:

```
# ip          Active  User      Nproc   Mem     Prog    Dir
45.3.23.189    1     user1      4    16000MB   g16    /home/user1/Documents/gwork     # 0
45.3.23.190    1     user1      4    16000MB   g16    /home/user1/Documents/gwork     # 1
45.3.23.135    1     student    4    16000MB   g16    /srv1/gwork                     # 2
45.3.23.158    1     student    4    16000MB   g16    /home/student/Documents/gwork   # 3
45.3.23.157    1     student    4    12000MB   g09    /home/student/Documents/gwork   # 4
45.3.23.153    1     student    8     6000MB   g09    /home/student/Documents/gwork   # 5
45.3.23.167    1     student    4     6000MB   g16    /srv1/gwork                     # 6
45.3.23.165    1     student    4     6000MB   g09    /home/student/Documents/gwork   # 7
45.3.23.131    1     student    4     6000MB   g09    /home/student/Documents/gwork   # 8
45.3.23.134    1     student    4     6000MB   g09    /home/student/Documents/gwork   # 9
45.3.23.143    0     student    4     6000MB   g09    /home/student/Documents/gwork   # 10
45.3.23.144    1     student    4     6000MB   g09    /home/student/Documents/gwork   # 11
45.3.23.148    1     student    4     6000MB   g09    /home/student/Documents/gwork   # 12
```

4. *Файл workdirs.txt*

Файл представляет собой текстовый ASCII-файл, в котором каждая строка описывает директорию, в которой будет запущено задание методом BATCH с использованием *slurm* или *pbs* . Пустые строки и строки, начинающиеся с символа #, игнорируются. Часть строки после # рассматривается как комментарий.

Каждая строка имеет формат:

<Dirname > <Active> <No.> <Script>

Dirname	Имя директории
Active	Будет ли узел использоваться в текущей оптимизации (0 или 1)
No.	Порядковый номер директории. Не влияет на работу, играет характер

	комментария.
Script	Форма для BATCH-скрипта, используемый для запуска заданий в данной директории. На основе этой формы УС создает окончательный BATCH-script в данной директории. В форме должны находиться метки %%% и @@@, на месте которых УС помещает, соответственно, уникальное имя задания и input-файл программы-сервера.

Пример:

```
#Dirname  Active?    No.   Script
wd0         1         0      batch.run0
wd1         1         1      batch.run0
wd2         1         1      batch.run0
wd3         1         3      batch.run0
wd4         1         4      batch.run0
wd5         1         5      batch.run0
wd6         1         6      batch.run0
wd7         1         7      batch.run0
wd8         1         8      batch.run0
wd9         1         9      batch.run0
wd10        1        10      batch.run_skx
```

5. *Файл <batch.template>*

Файл-форма для создания скрипта запуска задач в режимах BATCH и CRON представляет собой несколько измененный обычный скрипт этой программы, в котором место, где размещаются название задачи отмечено меткой %%%, а место, где размещается название входного файла программы-сервера – меткой @@@. В процессе работы текст %%% будет автоматически заменятся на имя задачи в виде SXXXXX, где S – буква, указанная пользователем в параметре *jletter* в файле управляющего скрипта (см. раздел *Запуск программы*), а XXXXX соответствует номеру структуры при ее оптимизации. Использование различных параметров *jletter*=”<*S*>” для разных оптимизируемых систем позволяет различать одновременно работающие задания, отображаемые командами *squeue* и *qstat.*

Пример -- система *slurm:*

```
#!/bin/bash
#SBATCH -J %%%          ← обязательная метка %%% названия задачи
#SBATCH -o %%%.o%j      ← (может быть не одна)
#SBATCH -e %%%.e%       ← (может быть не одна)
#SBATCH -p normal
#SBATCH -N 1
#SBATCH -n 63
#SBATCH -t 12:00:00

module load gaussian/16rA.03

InputFile=@@@           ← обязательная метка @@@ названия input-файла
```

```
g16 ${InputFile}
```

Пример -- система *pbs:*

```
#!/bin/bash
#PBS -N %%%              ← обязательная метка %%% названия задачи
#PBS -l nodes=1:ppn=12

export GAUSS_LFLAGS='-opt "Tsnet.Node.Lindarsharg:ssh"'
cd $PBS_O_WORKDIR

inputfile=@@@            ← обязательная метка @@@ названия input-файла
output=${inputfile%.gjf}.log

g09 <$PWD/${inputfile} >$PWD/${output}
```

Запуск программы

1. *Режим прямого вызова*

Команда запуска (Windows):

```
runner.exe mg10.inp
```

2. *Режим SSH*

Команда запуска программы в фоновом режиме:

```
./runner.x mg10.inp &
disown
```

УС запускается автоматически либо вручную до или после запуска программы, в зависимости от параметра ShellControl (см. раздел *Ключевые слова входного файла*). При запуске вручную команда запуска УС:

```
./runner_ssh.sh >res.log
```

При запуске УС запрашивает пароли для всех удаленных узлов, которые должны быть введены пользователем в ответ на запросы:

```
>Password for node 1, user student:  absde123
>Password for node 2, user user1:    jhkk12h
…
```

После ввода паролей УС может быть переведен в фоновый режим:

```
Ctrl-Z
bg
disown
```

В ходе работы УС файл *res.log* будет содержать обновляемую информацию о его работе.

3. *Режим BATCH*

В этом режиме рекомендуется запускать сначала УС, который далее автоматически запускает программу. Команда запуска:

```
./runner_batch.sh >res.log &
```

Перед запуском УС, его следует отредактировать с помощью любого редактора (mc, vi …), задав несколько обязательных опций, описанных в начале файла `runner_batch.sh`:

```
########################################
#### Define your defaults here: #########
########################################

# These are important parameters! Check it carefully!
user="student"          ← указать свой login (username)
InputFile=mg7.inp       ← указать входной файл данных для runner.x
jletter="y"             ← указать букву для отличия заданий друг от друга
curdir=$PWD
idle_time=300           ← опционально: время сна между проверкой файлов, с
maxcyc=1000000          ← опционально: число циклов работы УС

#for PBS only  (prefix added by qsub to JobNo)
compname=".master1.cyberia.tsu.ru"  ← для pbs обязательно указать суффикс
                                      номера задания, выдаваемый командой qstat
```

4. *Режим CRON*

Запуск программы (а также настройка УС перед запуском) полностью аналогичны режиму BATCH. Команда запуска:

```
./runner_batch_cron.sh >res.log &
```

Алгоритм глобальной оптимизации, реализованный в RUNNER

Алгоритм глобальной оптимизации, реализованный в RUNNER, основан на эволюционном (генетическом) алгоритме в комбинации с табу-поиском. Генетический алгоритм базируется на применении генерирующих операторов (генераторов) к начальному и обновляемому пулу структур для создания структур-потомков. Метод отбора «лучших потомков» заключается в вытеснении структур пула оптимизированными структурами с более высоким фитнесом (с низшей полной энергией).

В программе реализованы 10 типов генераторов:

RAND	Случайная генерация атомов в сфере радиусом, задаваемым командой *Sphere*

RND1	Генерация атомов методом конденсации: новый атом, случайно генерируемый в сфере радиуса *Sphere*, движется к центру структуры, пока расстояние до ближайшего атома не окажется в пределах, указанных в команде *BondLimits*
PCCR	*Plane-Cut Crossover*: Две случайно выбранные из пула структуры разрезаются случайной плоскостью так, чтобы соотношение числа атомов в обеих частях двух структур было равным, и первые части двух структур меняются местами.
TPCR	*Two-Point Crossover:* Декартовы координаты двух структур выстраиваются в одномерные векторы и делятся на две части в случайном положении, одинаковом для обеих структур. Затем первые части двух векторов меняются местами.
IMOV	*Internal Move:* Случайно выбранный атом случайно выбранной структуры пула перемещается на случайное расстояние в направлении центра масс структуры.
AMOV	*Angular Move:* Случайно выбранный атом случайно выбранной структуры пула поворачивается на случайный угол вокруг центра масс структуры.
TWST	*Twist operator:* Случайно выбранная из пула структура разрезается случайной плоскостью на две части и обе части поворачиваются друг относительно друга на случайный угол вокруг оси, проходящей через центр масс и перпендикулярной плоскости разреза.
HINT	*Hint operator:* новая структура выбирается на основе «подсказки», т.е. структуры, предоставленной пользователем. В случае RUNNER файл со структурами-подсказками помещается в директорию *./add* в любое время работы программы. После генерации входных файлов, файлы-подсказки перемещаются в директорию ./usd. Форматами файлов являются (1) выходные файлы программы-сервера с оптимизированной геометрией; (2) текстовые файлы, в которых структуры-подсказки приведены в формате NXYZ с метками начала координат @geo, окончание координат – пустая строка.

Глобальная оптимизация начинается с фазы формирования начального набора структур («начальной популяции»). Эта фаза называется «начальная фаза» (Initial Phase). В ходе нее накапливается набор структур, на основе которых далее будут генерироваться «структуры-потомки». Формирование начальной популяции производится, как правило, двумя операторами RAND и RND1 («случайная генерация» и «случайная конденсация»). Предпочтительность одного из двух операторов можно изменить, используя команду *Init*, указывая в ней применяемые операторы и вероятность их использования. Кроме того, для начальной фазы можно использовать структуры-подсказки (оператор HINT). Начальная фаза заканчивается, когда формируется минимальная популяция («пул») структур, способных образовывать «потомков». Число таких структур определяется командой *MinPool* (по умолчанию 10). Входящие в пул структуры сортируются по увеличению их абсолютных энергий и хранятся в файле **.pool_*.

По окончании начальной фазы начинается основная фаза алгоритма (Main Production Phase). В ее ходе на каждом рабочем цикле алгоритма формируются новые структуры-«потомки» путем применения заданного набора генераторов к случайно выбранным одной или нескольким структурам из «активного пула». Активный пул – это часть общего пула, объединяющая наиболее выгодные структуры (структуры с низшими энергиями). Доля наиболее выгодных структур от общего объема пула задается командой *ActivePool* (по умолчанию 0.7, т.е. 70% от всех структур пула). Уменьшение этого числа может привести

к ускорению оптимизации, если ГМ близок по структуре к уже найденным структурам. В то же время это может привести к потере «биоразнообразия» и неудаче при поиске ГМ необычной структуры. Избыточное увеличение *ActivePool* может привести к низкой скорости оптимизации.

В случае, когда в процессе оптимизации возникают структуры, сходные по структуре и энергии со структурами, входящими в пул, в пуле остается только та, которая имеет более низкую энергию. Сходство структур определяется алгоритмом на основе анализа сортированных по возрастанию межатомных расстояний в обеих структурах. Структуры считаются сходными, если все такие расстояний в двух структурах отличаются менее чем на 5%. Если в процессе оптимизации число различных оптимизированных структур превышает максимальный объем пула (он задается командой *MaxPool*), новая структура занимает свое место в порядке сортированных по возрастанию энергий, а структуры, номера которых превышают размер пула, выходят из него.

Выход структуры из пула не означает, что она не используется при дальнейшей работе. При генерации новой структуры программа проверяет сходство новой структуры со всеми структурами, для которых когда-либо в текущем запуске рассчитывалась энергия. Эти структуры хранятся в файле **.base_*. В этом файле хранятся не только оптимизированные структуры, но и все структуры, найденные на промежуточных циклах локальной оптимизации, выполняемых программой-сервером. Перед тем как начать расчет энергии (локальную оптимизацию) новой структуры, она проверяется на сходство со всеми структурами, хранящимися в **.base_* и расчет начинается, только если сходных структур не найдено. В противном случае, новая структура отбрасывается и генерируется заново. Кроме того, структура отбрасывается, если ее длины связи выходят за пределы, определенные командой *BondLimits*. Число попыток генерации определяется командой *MaxTry* (по умолчанию 5000). Если за указанное число попыток данный генератор не смог создать новую структуру, генерация выполняется оператором RAND. Кроме файла *.base_ RUNNER создает его укороченную версию, файл **.sbas_*. В нем содержатся только начальные (генерированные) и конечные (оптимизированные либо недооптимизирванные) структуры каждого расчета, промежуточные циклы оптимизации опускаются. Файл **.sbas_* удобнее анализировать, чем **.base_*. В файлах **.base_* и **.sbas_* начальные структуры помечены меткой **gen*, конечные (в данном файле) – меткой **end*, оптимизированные – меткой **opt*. Кроме того, при выполненном расчете частот, структура помечается меткой **freq*, и, если отсутствуют мнимые частоты, меткой **lm*. Если структура имеет *n* мнимых частот, она помечается меткой **nneg=<n>*.

Генерированные структуры записываются в файл входных данных программы-сервера, форма которого задается командой *FilPat*. Новый файл входных данных помещается в директорию ./inp. Из этой директории УС (или сама программа RUNNER в случае режима прямого вызова) отправляет входной файл программе-серверу и перемещает его в директорию ./out. По окончании расчета сервером УС помещает в эту же директорию файл результатов. Обнаруживая в директории ./out файл результатов, RUNNER анализирует результаты, извлекает их него структуры каждого шага оптимизации и их энергии и записывает их в файлы **.base_* и **.sbas_*. Одновременно обновляется пул и его файл **.pool_* , а входной и выходной файлы из директории ./out перемещаются в

директорию ./arc (архив всех проведенных оптимизаций). Сохранение этих данных в директории ./arc можно отменить командой *NoSaveArchive*. При этом экономится дисковое пространство, но в случае возникновения ошибок оптимизации будет невозможно установить их причину.

Работа программы RUNNER прекращается в нескольких случаях:

1. Проведена локальная оптимизация всех структур, число которых задано командой *Nstruct;*
2. Превышено число рабочих циклов, заданное командой *MaxCyc*;
3. В директории запуска появился файл *runner_stop* или *runner_prog_stop.*

Работа УС прекращается в случаях:

1. Превышено число рабочих циклов УС, заданное внутри скрипта параметром *maxcyc* (все буквы в нижнем регистре!);
2. В директории запуска появился файл *runner_stop* или *runner_shell_stop.*

Присутствие файлов *runner_stop* одновременно останавливает программу и УС, присутствие *runner_prog_stop* или *runner_shell_stop* – только программу или только УС. Содержимое всех этих файлов не имеет значения, они могут быть пустыми. В случае присутствия этих файлов в директории запуска, работа программы или УС невозможна. Для начала работы эти файлы должны быть удалены.

В случае, если работа программы или УС ошибочно завершена до завершения ГО, рестарт осуществляется командой, аналогичной обычному запуску. При этом RUNNER автоматически определяет свое предшествующее состояние и продолжает ГО. Аналогично действует УС. Обязательным условие успешного рестарта является сохранение всех файлов в директориях ./inp, ./out, ./sub и файла **.info_* в директории запуска. Перед рестартом рекомендуется сохранить копии файлов **.info_*, **.out* и *res.log*. Файл **.info_* хранит информацию о текущем состоянии проводимой оптимизации, его ручное редактирование не рекомендуется.

В случае необходимости продолжить завершенную ГО (ГО, в которой исследовано *Nstruct* структур), необходимо использовать команду *Continue* в файле **.inp*. В этом случае текущее значение *Nstruct* определяет дополнительное количество структур, которые должны быть исследованы.

Автор, лицензии и отзывы

Автор программы:

Проф. Игнатов Станислав Константинович,

Группа теоретической химии,

Нижегородский государственный университет им. Н.И. Лобачевского,

skignatov@gmail.com , http://qchem.unn.ru

г. Нижний Новгород, 2020 г.

Вопросы о работе с программой, сообщения о возможных неточностях и ошибках, отзывы и пожелания направлять по адресу skignatov@gmail.com