УДК 004.85

Абдрахманов Дмитрий Леватович, Коичеиков Владимир Игоревич,
Лукьянов Артем Олегович, Тарабанова Татьяна Владимировна

ИССЛЕДОВАНИЕ ВЫЧИСЛИТЕЛЬНОЙ ЭФФЕКТИВНОСТИ МОДЕЛИРОВАНИЯ В ПАКЕТЕ LAMMPS С ИСПОЛЬЗОВАНИЕМ ПОТЕНЦИАЛА МЕЖАТОМНОГО ВЗАИМОДЕЙСТВИЯ, ПОЛУЧЕННОГО СЕТЬЮ SCHNET

Описан опыт использования пакета моделирования методом классической молекулярной динамики LAMMPS с потенциалом межатомного взаимодействия, вычисляемым при помощи сверточной нейронной сети SchNet. Обсуждаются особенности сборки пакета LAMMPS для эффективного использования графического процессора в расчетах.

Молекулярная динамика, нейросетевой потенциал, графический процессор, вычислительный кластер.

Abdrakhmanov Dmitry Levatovich, Konchenkov Vladimir Igorevich,
Lukyanov Artyom Olegovich, Tarabanova Tatyana Vladimirovna

INVESTIGATION OF THE COMPUTATIONAL EFFICIENCY OF MODELING IN THE LAMMPS PACKAGE USING THE POTENTIAL OF INTERATOMIC INTERACTION OBTAINED BY THE SCHNET NETWORK

The experience of using the LAMMPS classical molecular dynamics simulation package with the potential of interatomic interaction calculated using the convolutional neural network SchNet is described. The features of the LAMMPS package assembly for efficient use of the GPU in calculations are discussed.

Molecular dynamics, neural network potential, graphics processor, computing cluster.

Введение

Моделирование методом молекулярной динамики является широко применяемым методом для исследования транспортных, тепловых и других физических свойств различных материалов, в том числе кристаллических, аморфных, а также низкоразмерных систем. Различают квантовую молекулярную динамику (англ. *ab initio* molecular dynamics, AIMD) и классическую молекулярную динамику (англ. classical molecular dynamics, CMD). Несмотря на высокую точность, AIMD подходит только для небольших систем (не более нескольких сотен атомов) и модельных временных промежутков не более сотен фемтосекунд. С другой стороны, CMD позволяет рассматривать системы, состоящие из миллионов атомов на модельных временах порядка наносекунд. В рамках классической молекулярной динамики необходимо, кроме начальных координат и скоростей частиц, знать силовое поле в образце. Подобрать подходящую, дающую физически обоснованные результаты модель силового поля – особое искусство. С развитием методов глубокого обучения появился следующий подход к получению потенциала межатомного взаимодействия, позволяющего сохранить точность AIMD-методов, но рассматривать размеры образцов и модельные времена, характерные для классической молекулярной динамики. Нейронная сеть обучается на данных AIMD (признаки – положения атомов, свойства – силы, действующие на каждый из атомов, и энергия всей системы), а затем обученная нейронная сеть встраивается в систему моделирования методом классической молекулярной динамики (например, LAMMPS [1]) в качестве функции, вычисляющей силовое поле. Одним из лидеров является пакет DeePMD, построенный на использовании полносвязной нейронной сети. В работе [2] рассматривается моделирование черного фосфорена с использованием пакетов DeePMD и LAMMPS, получены близкие к экспериментальным значения плотности и теплопроводности образца. В работе [3] рассматривается моделирование силового поля полимера – полифениленсульфида, показано, что из-за большого числа конфигураций, формируемых тепловым движением в этом материале (по сравнению с кристаллическим материалом) полносвязная нейронная сеть пакета DeePMD не в состоянии адекватно представить силовое поле. В связи с этим вызывают интерес исследования применимости нейронных

сетей других архитектур к задаче восстановления силовых полей кристаллических и аморфных материалов. В настоящей работе рассматривается сеть SchNet [4, 5] – сверточная сеть с непрерывной фильтрацией (англ. continuous-filter convolutional neural network). Как и DeePMD, пакет SchNetPack поддерживает интеграцию с пакетом LAMMPS. В работе приводится описание особенностей сборки LAMMPS совместно с SchNetPack на кластере Волгоградского государственного технического университета с поддержкой расчетов на видеокарте при помощи технологии NVidia CUDA.

Конфигурация вычислительного кластера

Кластер Волгоградского государственного технического университета, в числе прочего, оснащен вычислительными узлами, имеющими в своем составе видеокарты NVidia GeForce RTX 3060. На кластере работает очередь SLURM, используется операционная система Rocky 8, являющаяся бесплатным аналогом Red Heat Enterprise Linux. Вычислительные узлы связаны друг с другом по сети Infiniband. Согласно руководству по установке пакета SchNetPack 2.0 [4], для функционирования этого пакета совместно с LAMMPS необходимо установить пакет CUDA 11.7. Основной сложностью является корректная сборка LAMMPS с учетом настроек кластера. Ниже приведены основные настройки, хранящиеся в файле ~/.bashrc пользователя.

```
source /opt/rh/gcc-toolset-11/enable
```

Эта строка определяет стандартную для RHEL-подобных систем замену системного gcc-8 для версии Rocky 8.x на версию gcc-11.

```
export PATH="/opt/cuda/cuda-11.7/bin/:$PATH"
```

Данная команда включает в стандартные пути исполняемые файлы компилятора Nvidia Cuda (nvcc) и его отдельных компонентов

```
export PATH="/home/test_user/cmake/bin/:$PATH"
export PATH="/home/test_user/miniconda3/:$PATH"
```

Данные строки указывают пути к расположению miniconda3 и cmake. Эти программы установлены в домашней директории пользователя, что существенно облегчает настройку окружения, давая возможность пользователю собрать свои варианты пакета LAMMPS, не затрагивая файлы других пользователей кластера.

Сборка пакета LAMMPS

Сборка пакета LAMMPS осуществляется согласно разделу “LAMMPS Interface” руководства [4]: при помощи `conda` создается окружение

spk_lammps с нужными зависимостями, это окружение активируется, затем при помощи пакетного менеджера pip устанавливается пакет schnetpack.

```
conda create -n spk_lammps python=3.9 cuda-toolkit=11.7 pytorch
mkl-include numpy -c pytorch -c nvidia
conda activate spk_lammps
pip install schnetpack
```

После этого клонируем репозиторий LAMMPS и загружаем файлы pair_schnetpack.cpp, pair_schnetpack.h, patch_lammps.sh:

```
git clone --depth 1 git@github.com:lammps/lammps
cd ~/miniconda/envs/spk_lammps/lib/python3.9/site-packages/
schnetpack/interfaces/
mkdir spk_lammps
cd spk_lammps
…
wget https://raw.githubusercontent.com/atomistic-machine-
learning/ schnetpack/master/interfaces/lammps/patch_lammps.sh
chmod u+x patch_lammps.sh
```

Пакет SchNetPack 2.0 предполагает использование версии pytorch 1.13, а при описанных выше настройках устанавливается версия 2.x этого пакета. Проблема решается исправлением в файле patch_lammps.sh версии стандарта C++. Ниже приведен фрагмент файла patch_lammps.sh, в котором при использовании Torch заменяется версия 11 стандарта C++ на версию 14. Для использовании версий 2.x пакета pytorch необходимо в этой части файла patch_lammps.sh исправить значение параметра CMAKE_CXX_STANDARD с 14 на 17.

```
sed -i "s/set(CMAKE_CXX_STANDARD 11)/set(CMAKE_CXX_STANDARD
14)/" $lammps_dir/cmake/CMakeLists.txt
```

Далее применяем скрипт patch_lammps.sh:

```
./patch_lammps.sh <path/to/lammps>
cd <path/to/lammps>
mkdir build
cd build
```

В руководстве [6] подготовку сборки рекомендуется осуществлять следующей командой:

```
cmake ../cmake -DCMAKE_PREFIX_PATH=`python -c 'import
torch;print(torch.utils.cmake_prefix_path)'` -DMKL_INCLUDE_DIR=
"$CONDA_PREFIX/include"
```

На используемой вычислительной системе представленная команда стала работоспособной только с некоторыми дополнениями. Пакет LAMMPS может быть собран как с поддержкой вычислений на графических процессорах (GPU), так и без такой поддержки. Основным фактором возможности организации эффективных вычислений при помощи графического процессора является наличие потенциала межатомного взаимодействия, поддерживающего вычисления на видеокарте. В случае использования нейросетевого потенциала, формируемого сетью SchNet, такая поддержка есть. В пакете LAMMPS поддержка вычислений с использованием технологии CUDA может осуществляться различными способами. Ниже приведен скрипт, который осуществляет взаимодействие LAMMPS с видеокартой через пакет GPU.

```
cmake ../cmake -DCMAKE_MINIMUM_REQUIRED=3.8 -
DCMAKE_CXX_STANDARD=17 -DCMAKE_CXX_STANDARD_REQUIRED=ON -
DCMAKE_CXX_EXTENSIONS=OFF
-DCMAKE_C_COMPILER=/opt/rh/gcc-toolset-11/root/usr/bin/gcc
-DCMAKE_CXX_COMPILER=/opt/rh/gcc-toolset-11/root/usr/bin/g++
-DCMAKE_BUILD_TYPE=Release -DUSE_CUDA=ON
-DCMAKE_CUDA_ARCHITECTURES=86
-DCUDA_TOOLKIT_ROOT_DIR=/opt/cuda/cuda-11.7
-DCMAKE_CUDA_COMPILER=/opt/cuda/cuda-11.7/bin/nvcc
-DCAFFE2_USE_CUDNN=1 -DCAFFE2_USE_CUSPARSELT=ON
-DCUSPARSELT_INCLUDE_PATH=/home/test_user/cusparselt/include
-DCUSPARSELT_LIBRARY_PATH=/home/test_user/cusparselt/lib
-DCMAKE_PREFIX_PATH=
`python -c 'import torch;print(torch.utils.cmake_prefix_path)'`
-DMKL_INCLUDE_DIR="$CONDA_PREFIX/include" -DPKG_RIGID=yes
-DPKG_GPU=ON -DGPU_API=cuda -DGPU_PREC=mixed -DGPU_ARCH=sm_86
-DCUDPP_OPT=NO -DCUDA_MPS_SUPPORT=YES
```

В приведенном выше скрипте явно указаны пути к установке CUDA Toolkit, а также библиотек CuDNN, Caffe2, используемых Torch при вычислениях на видеокарте. После конфигурирования, выполняемого пакетом CMAKE, осуществляем сборку командой

```
make -j$(nproc)
```

Для использования нейросетевой модели, обученной при помощи пакета SchNetPack, необходимо выполнить команду

```
spkdeploy phosphorene_model_400 phosphorene_model
```

Во входном файле для LAMMPS нужно указать (16 – количество атомов в моделируемой элементарной ячейке фосфорена)

```
newton          off
pair_style  schnetpack
pair_coeff  * * phosphorene_model 16
```

Производительность вычислений

Пакет LAMMPS при сборке с использованием скрипта, представленного выше, может выполнять все расчеты на видеокарте. Основным ограничивающим фактором является объем памяти видеокарты, доступной для вычислений (в нашем случае – 12 Гб). Наиболее затратной по использованию памяти является операция по построению списка соседей. Список соседей строится в процессе обучения нейронной сети. Поскольку постоянные решетки у рассматриваемого в нашем примере черного фосфорена a_x = 4.376 Å, a_y = 3.314 Å, а радиус обрезки (радиус, в рамках которого учитывается взаимодействие выбранного атома с соседними атомами) должен быть не менее постоянной решетки, выбираем радиус обрезки r_c = 5.0 Å. При этом размер батча данных при обучении сети на видеокарте составляет batch_size = 200 … 300, при больших значениях наблюдается переполнение памяти видеокарты. Общее количество атомов в модели равно 5888. При использовании вычисляемого нейронной сетью потенциала при моделировании методом классической молекулярной динамики список соседей также строится, причем операции записи и чтения из памяти видеокарты занимают большую часть времени. Поэтому, хотя можно собрать LAMMPS без использования пакета GPU, чтобы собственно расчеты молекулярных траекторий велись центральным процессором, существенной разницы в производительности нет. Кроме того, проводились вычислительные эксперименты по обучению нейронной сети и выполнению классической молекулярной динамики на ПК, не имеющем в своем составе видеокарту с поддержкой CUDA (процессор AMD Ryzen 3 5425U 2.70 GHz, 16 ГБт ОЗУ). В этом случае скорость обучения нейронной сети, как и ожидалось, была в несколько раз ниже, а расчет молекулярных траекторий производился за время, сравнимое со временем расчета на видеокарте. Плюсом от переноса всех расчетов LAMMPS на видеокарту вычислительной ноды кластера является, прежде всего, экономия вычислительного времени процессорных ядер.

Выводы

В рамках работы произведена настройка пакета SchNetPack, вычисляющего силовое поле, которое затем используется в пакете LAMMPS для вычисления молекулярных траекторий на примере приведения к равновесному состоянию образца черного фосфорена. Обсуждаются особенности сборки пакета LAMMPS, вычислительная эффективность различных сборок. Показано, что использование видеокарты на этапе обучения нейронной сети и на этапе вычисления молекулярных траекторий является обоснованным решением.

Работа поддержана грантом РНФ 23-22-00461 «Исследование тепловых свойств упорядоченных и неупорядоченных низкоразмерных материалов методом молекулярного моделирования с потенциалами, полученными при помощи глубокого машинного обучения» (конкурс 2022 года «Проведение фундаментальных научных исследований и поисковых научных исследований малыми отдельными научными группами»).

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. https://github.com/lammps/lammps

2. *Шеин Д.В., Завьялов Д.В., Жариков Д.Н.* Моделирование фосфорена методом классической молекулярной динамики с использованием глубокого обучения // Физика. Технологии. Инновации. ФТИ-2022 : тез. докл. IX Междунар. молодеж. науч. конф., посвящ. 100-летию со дня рожд. проф. С. П. Распопина (г. Екатеринбург, 16-20 мая 2022 г.). Екатеринбург, 2022. - С. 330-331.

3. *Шеин Д.В., Завьялов Д.В., Конченков В.И.* Исследование применимости нейронных сетей с прямой связью для компьютерного моделирования полимеров // Журнал технической физики. 2023. Т. 93. Вып. 12. С. 1732 – 1735. DOI: 10.61011/JTF.2023.12.56806.f242-23

4. https://schnetpack.readthedocs.io/en/latest/getstarted.html

5. *Schütt K.T., Hessmann S.S.P., Gebauer N.W.A., Lederer J., Gastegger M.* SchNetPack 2.0: A neural network toolbox for atomistic machine learning // Journal of Chemical Physics. 2023. V. 158. Is. 14. P. 144801. DOI: 10.1063/5.0138367

Абдрахманов Дмитрий Леватович, преподаватель кафедры «Электронно-вычислительные машины и системы» Волгоградского государственного технического университета.

Конченков Владимир Игоревич, доцент кафедры «Электронно-вычислительные машины и системы» Волгоградского государственного технического университета, Россия, город Волгоград, пр. им. В.И. Ленина, 28, 400005, телефон: +7 904 756 86 41, e-mail: kontchenkov@yandex.ru.

Лукьянов Артем Олегович, магистрант Волгоградского государственного технического университета.

Тарабанова Татьяна Владимировна, магистрант Волгоградского государственного технического университета.

Abdrakhmanov Dmitry Levatovich, Lecturer at the Department of Electronic Computing Machines and Systems, Volgograd State Technical University .

Konchenkov Vladimir Igorevich, Associate Professor of the Department of Electronic Computing Machines and Systems, Volgograd State Technical University, 28 V.I. Lenin Ave., Volgograd, Russia, 400005, phone: +7 904 756 86 41 , e-mail: kontchenkov@yandex.ru.

Lukyanov Artyom Olegovich, undergraduate student at Volgograd State Technical University.

Tarabanova Tatyana Vladimirovna, undergraduate student at Volgograd State Technical University.