Астафьев А.В., Астафьев А.С., Кондрушин И.Е.
Муромский институт (филиал) федерального государственного образовательного учреждения высшего образования «Владимирский государственный университет имени Александра Григорьевича и Николая Григорьевича Столетовых»
602264, г. Муром, Владимирская обл., ул. Орловская, 23
E-mail: Alexandr.Astafiev@mail.ru

Организация позиционирования радиоустройств внутри помещений на основе информации о состоянии канала связи

При разработке современных прикладных систем всё большее внимание уделяется точности позиционирования разрабатываемых устройств или объектов интереса. Если в открытом пространстве большинство задач решается за счёт использования систем глобального позиционирования (GPS, ГЛОНАСС и т.д.), то в закрытых пространствах использование подобных методов дает менее точный результат в виду наличия большого количества физических преград и эффекта многолучевого распространения. Для решения задач внутреннего позиционирования и организации бесшовной навигации используются методы и алгоритмы, основанные на техническом зрении, анализе магнитного поля, данных с инерциальных датчиков и радиоустройств [1].

В работе рассматривается организация позиционирования объекта интереса с установленным радиоустройством, работающим на основе технологии WiFi. Большинство исследований в данном направлении рассматривают показатель уровня принимаемого сигнала RSSI для построения алгоритмов. Однако, в последние годы всё больше научных групп рассматривают информацию о состоянии канала связи (CSI) как более подробную информацию о характере распространения сигнала. Благодаря использованию модуляции OFDM можно получить информацию о 56 поднесущих при использовании стандарта WiFi4 [2], а использование технологии MIMO позволяет получать эти данные по каждой паре антенн. Визуально, процесс извлечения передачи информации о состоянии канала связи можно представить в виде рисунка 1.

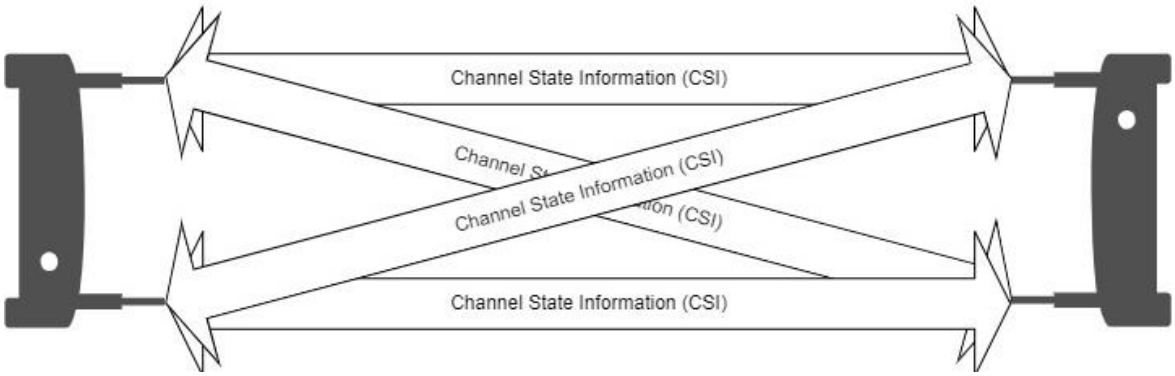

Рис. 1. Процесс передаци информации в сетях WiFi

Исходя из этого, целью исследования является разработка методов и алгоритмов для организации позиционирования внутри помещений с использованием информации о состоянии канала связи и искусственных нейронных сетей.

Для формирования обучающей выборки был проведен эксперимент в технической лаборатории размером 6х5 метров. В дальней часть лаборатории устанавливался стационарный принимающий маршрутизатор. В качестве объекта интереса выступал передающий маршрутизатор, который последовательно перемещался в различные точки наблюдения. Точки наблюдения располагались в прямой видимости принимающего маршрутизатора с шагом в 25 сантиметров. Для учёта углового смещения были проведены замеры со смещением в 50 сантиметров. Общее количество точек сбора информации о состоянии канала связи составило 90. Для каждого расстояния было собрано более 6000 измерений.

Для достижения сбалансированности обучающей выборки было принято решение использовать по 5000 измерений на каждой дальности. Тестовая выборка включает в себя по 800 измерений на каждое расстояние.

Таким образом, обучающая выборка включает в себя 70 000 измерений информации о состоянии канала связи: 15 680 000 значений фаз и 15 680 000 значений амплитуд.

Для организации позиционирования задача была представлена как задача классификации расстояния между маршрутизаторами. Для решения поставленной задачи был использован аппарата нейронных сетей. На вход нейронной сети подавались данные о фазе и амплитуде по всем парам антенн. Таким образом, количество нейронов входного слоя равно 448. В качестве скрытых слоёв были использованы полносвязные слои, состоящие из 256 нейронов с последующим слоем регуляризации (dropout). На выходе был применен полносвязный слой с функцией активации softmax. Количество нейронов на выходном слое соответствует количеству классов расстояний.

После проведенных экспериментальных исследований был сделан вывод, что величина ошибки определения расстояния варьируется в диапазоне 0,55-3,13%. Однако, в случае рассмотрения этих вероятностей как вероятностей ошибок независимых событий, то величина ошибки варьируется в пределах от $0{,}35*10^{-7}$ до $0{,}45*10^{-4}$.

Список литературы:

1. Astafiev A.V., Zhiznyakov A. L., Zakharov A. A., Privezentsev D. G. Algorithm for Preliminary Processing Channel State Information of the WIFI Communication Channel for Building Indoor Positioning Systems. 24th International Conference on Digital Signal Processing and its Applications (DSPA), 2022, pp. 1-4

2. Astafiev AV, Titov DV, Zhiznyakov AL, Demidov AA. A method for mobile device positioning using a sensor network of BLE beacons, approximation of the RSSI value and artificial neural networks. Computer Optics 2021; 45(2): 277-285. DOI: 10.18287/2412-6179-CO-826.

3. IEEE Std. 802.11n-2009: Enhancements for higher throughput. http://www.ieee802.org, 2009.

Болгак А.В.
Юго-Западный государственный университет
305040, г. Курск, ул. 50 лет Октября, 94

Алгоритмическая и высокоуровневая оптимизация в задаче умножения плотных квадратных вещественных матриц одинарной точности для однопоточной программной реализации с последующей оценкой реальной производительности

Высокопроизводительные вычисления используются во многих отраслях повседневной жизни. Областями применения таких вычислений являются оборонная промышленность, создание новейших лекарственных препаратов, высокочастотная торговля фьючерсами, моделирование различных деталей в машиностроении, проектирование роботизированных средств, томография, классификация бинарных отношений [1], прогнозирование смены климата, решение задач линейной алгебры и дифференциальных уравнений и так далее.

Умножение матриц – это одна из фундаментальных задач, решение которой позволяет эффективно задействовать основные вычислительные ресурсы современных процессоров и графических ускорителей, тем самым, увеличивая реальную производительность и уменьшая временные затраты на решение поставленной задачи за счет алгоритмической и высокоуровневой оптимизации соответствующей программной реализации.

Актуальность данной проблематики заключается в том, что в настоящее время многие алгоритмы решения прикладных задач сводятся именно к матричному умножению, при этом необходимо минимизировать дополнительные расходы, связанные с подготовкой данных, получаемых из оперативной памяти [2]. Время выполнения таких программных реализаций напрямую влияет на время решения той или иной прикладной задачи. Исходя из этого, разработано множество подходов, оптимизирующих выполняемые операции различными способами.

Целью данной работы является алгоритмическая и высокоуровневая оптимизация программной реализации задачи умножения квадратных матриц для однопоточной CPU-ориентированной программной реализации с последующей оценкой её производительности.

В работе рассмотрены основные аспекты анализа эффективности различных подходов к реализации решения задачи умножения квадратных матриц A и B размера N × N для однопоточной высокоуровневой программной реализации на современном поколении процессоров. Данная реализация ориентирована на использование процессоров семейства «x86» с оптимизацией работы кэш-памяти процессора.

Для решения поставленной задачи и оценки производительности каждого из алгоритмов были разработаны следующие программные реализации: «классическое» умножение, умножение с буферизацией столбца, блочное умножение. Исследование проводилось на базе процессора 11th Gen Intel(R) Core(TM) i5-11400H @ 2.70GHz, используемый компилятор – Microsoft Visual Studio 2017.

Анализ достигнутой реальной производительности вышеперечисленных программных реализаций позволяет сделать вывод о том, что умножение матриц с использованием блочного подхода является наиболее оптимальным и в каждом конкретном случае (размерность задачи, аппаратная конфигурация) имеет место вполне определенный оптимальный размер блока S, при котором достигается максимальная реальная производительность, а время умножения матриц становится минимальным.

Вероятнее всего, данный эффект связан с микроархитектурными особенностями реализации кэш-памяти для данного процессора (параметры оперативной памяти (латентность, пропускная способность), особенности размещения данных в КЭШах процессора различного уровня, их латентность, ассоциативность, пропускная способность связывающих их шин и так далее).

Также стоит отметить, что путем раскрутки внутреннего цикла (англ. loop unrolling) [3], формально являющейся высокоуровневой оптимизацией, удалось добиться дополнительного снижения времени выполнения умножения матриц. При этом значительно сокращается время обработки, а также изменяется оптимальный размер блока S (см. рис. 1). Раскрутка на разумное число итераций позволяет повысить параллелизм на уровне инструкций (сокр. ILP), тем самым

сокращая время выполнения умножения матриц. Чрезмерная раскрутка цикла, напротив, может привести к исчерпанию емкости кэша команд L1i, что существенно увеличивает время обработки алгоритма.

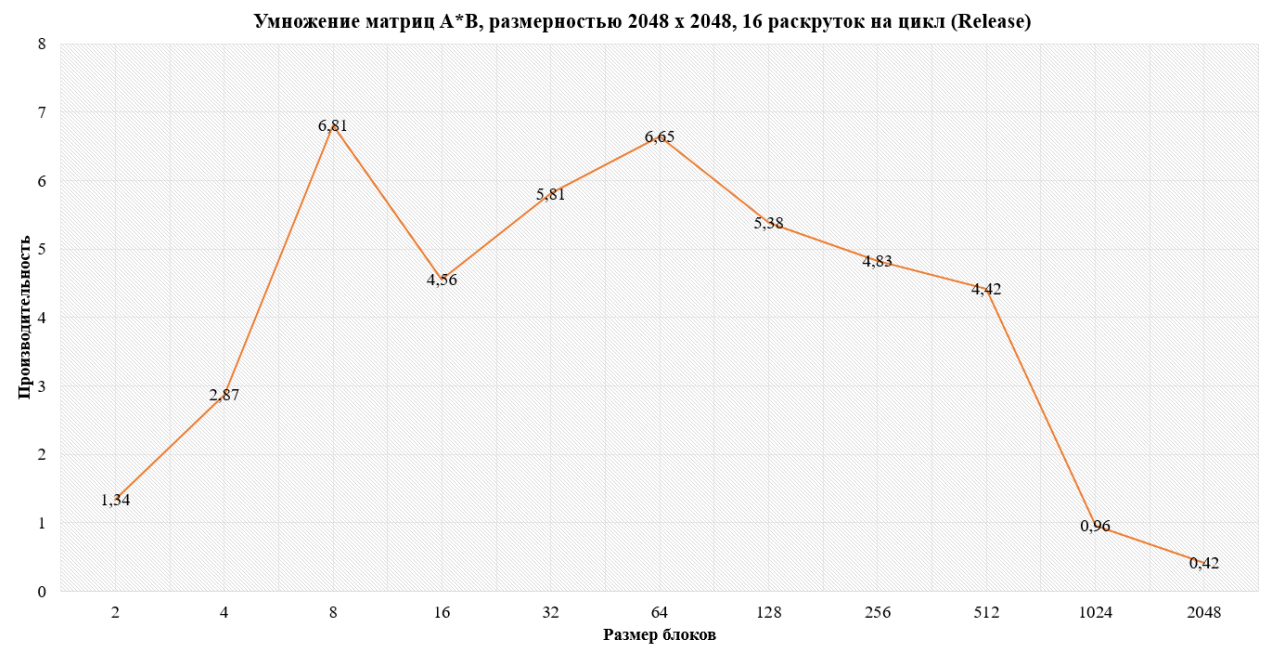

Рис. 1. График зависимости реальной производительности умножения вещественных матриц одинарной точности размером 2048 x 2048 от размера блока S (раскрутка внутреннего цикла – 16 итераций)

Сравнение блочного умножения и умножения с буферизацией столбца позволяет сделать вывод о том, что для процессора 11th Gen Intel(R) Core(TM) i5-11400H @ 2.70GHz оба варианта приводят к достижению сопоставимой производительности.

Таким образом, результаты исследования показали, что процессор 11th Gen Intel(R) Core(TM) i5-11400H @ 2.70GHz демонстрирует реальную производительность на уровне 2,1 – 6,8 GFLOP/s при однопоточной скалярной обработке данных, что в десятки раз превосходит производительность «классического» варианта умножения без оптимизации работы с памятью и сопоставимо с полученными ранее цифрами для процессоров предыдущих поколений [4]. В перспективе дальнейших исследований планируется анализ реальной производительности для матриц различного типа и разработка программных реализаций, ориентированных на выполнение на вычислительных средствах с параллельной архитектурой.

Литература

1. Ватутин Э.И., Зотов И.В. Построение матрицы отношений в задаче оптимального разбиения параллельных управляющих алгоритмов // Известия Курского государственного технического университета. Курск, 2004. № 2. С. 85–89.

2. Штейнберг Б.Я. Блочно-рекурсивное параллельное перемножение матриц // Известия высших учебных заведений. Приборостроение. 2009. Т. 52. № 10. С. 33–41.

3. Intel 64 and IA-32 Architectures Software Developer’s Manual. Volume1: Basic Architecture. Order number 253665-021.

4. Ватутин Э.И., Мартынов И.А., Титов В.С. Оценка реальной производительности современных процессоров в задаче умножения матриц для однопоточной программной реализации // Известия Юго-Западного государственного университета. Серия: Управление, вычислительная техника, информатика. Медицинское приборостроение. 2013. № 4. С. 11–20.

Быков А.А.
Муромский институт (филиал) федерального государственного образовательного учреждения высшего образования «Владимирский государственный университет имени Александра Григорьевича и Николая Григорьевича Столетовых»
602264, г. Муром, Владимирская обл., ул. Орловская, 23
E-mail: bykov_a_a@list.ru

Применение фазометрического метода в задачах геотехнического контроля процесса бурения

Проблема обеспечения населения чистой пресной водой уже давно является ключевой проблемой человечества. Для удаленных от водных артерий районов бурение артезианских скважин является решением этой проблемы. Водосодержащие горизонты могут находиться на разной глубине и для того, чтобы до них добраться следует использовать бурильные установки. Это дорогое оборудование, подверженное большим нагрузкам и часто выходящее из строя.

Повреждение бурильной установки приводит к большим денежным и временным затратам. К нему могут привести разные причины, но в основном они сводящиеся к переходу агрегатов бурильной установки во внештатный режим работы. Обычно это разнородные вибрации, вызванные разбалансированием работы агрегатов, переходом между слоями грунта разной плотности, установлением неправильного режима работы, и т.д.

Системы векторных измерений имеют высокую эффективность [1] и могут использоваться для обнаружения и локализации геодинамических процессов. Данный метод контроля обладает высокой помехоустойчивостью и чувствительностью по сравнению с фиксацией амплитудных параметров аномальных составляющих электромагнитного поля. В данном случае объектом контроля являются технологическое оборудование, скважина и окружающий ее грунт.

В работе [2] поясняется применение фазометрического метода геоэлектрического контроля, а именно использование нескольких источников зондирующих сигналов, расположенных в непосредственной близости от исследуемого объекта и необходимого количества векторных датчиков измерения электрического поля. При этом регистрация фазовых характеристик при фиксированном положении источников и измерительного базиса при возможности управления параметрами зондирующих сигналов основана на том, что первичное и вторичное электрические поля являются векторными величинами.

В соответствии с вышесказанным можно сделать вывод, что возможно организовать систему контроля процесса бурения с целью предотвращения процессов, вызывающих повреждения агрегатов бурильной установки. При этом имеется возможность повышения точности обнаружения водных горизонтов в соответствии с различными физико-химическими характеристиками грунта.

Фазометрический метод контроля в качестве зондирующего сигнала использует несколько источников, расположенных вблизи исследуемого объекта и необходимое количество векторных измерительных датчиков электрического поля.

Для оценки возможности использования фазометрического метода в задачах мониторинга грунта в процессе бурения скважин было проведено натурное исследование процесса прохождения бура через слои грунта с различными электрическими характеристиками во время бурения скважин на воду. Для этого была создана экспериментальная установка, в состав которой входит: источники зондирующих сигналов, устройства измерения и регистрации сигналов в среде, устройство обработки геодинамических данных. Изменение фазовых характеристик сигналов дает информацию о глубине погружения бура и о электрических характеристиках слоев грунта, в которые бур погружается, что позволяет судить о степени влагонасыщения грунта. При движении бура в однородной среде с постоянным влагонасыщением и с постоянной линейной скоростью изменение фазы во времени будет носить линейный характер.

Изменение влажности и вязкоупругих свойств грунта, приводит к изменению формы графика. Идея метода заключается в том, чтобы по виду графика определить процессы, которые протекают в данный момент на границе раздела бур-грунт.

Литература

1. Baknin, M.D., Bykov, A.A., Surzhik, D.I., Kuzichkin, O.R. Geotechnical monitoring of the foundations of structures based on integrated seismoelectric measurements in conditions of karst hazard. International Multidisciplinary Scientific GeoConference Surveying Geology and Mining Ecology Management, SGEMthis link is disabled, 2020, 2020-August(1.2), pp. 559–566

2. Bykov, A., Grecheneva, A., Kuzichkin, O., Surzhik, D., Vasilyev, G., & Yerbayev, Y. (2021). Mathematical description and laboratory study of electrophysical methods of localization of geodeformational changes during the control of the railway roadbed. Mathematics, 9(24), 3164.

Ватутин Э.И.
Юго-Западный государственный университет
305040, г. Курск, ул. 50 лет Октября, 94

Стратегия распределенной диагонализации латинских квадратов

В комбинаторике известным является понятие т.н. «комбинаторного взрыва» – ситуации, при которой с ростом размерности задачи N ее вычислительная сложность резко возрастает и применение ряда методов, работавших до этого для малых размерностей, становится невозможным из-за необходимости огромных вычислительных затрат. На примере решаемых в настоящее время задач, связанных с построением спектров быстровычислимых числовых характеристик диагональных латинских квадратов (ДЛК) [1], данная ситуация для диагонализации (построения всех возможных ДЛК, изоморфных заданному латинскому квадрату (ЛК)) и поквадратного обхода окрестностей [2–4] начинает проявляться начиная с размерности N=13. Поквадратный обход окрестностей при этом становится невозможен (приблизительная оценка необходимых вычислительных затрат – около 80 лет в проекте добровольных вычислений [3]), диагонализацию возможно реализовать по частям путем разработки специализированной распределенной программной реализации указанной процедуры.

В решаемых в настоящее время задачах, связанных с построением спектров, диагонализация применяется совместно с анализом окрестностей и позволяет как увеличить мощность результирующего спектра, так и сдвинуть его верхнюю и нижнюю границы (супремум и инфимум) в ситуациях, когда спектр близок в пределу и просто обходом окрестностей границы уже не сдвигаются. Для размерности N=12 диагонализация требует от нескольких часов до нескольких десятков часов на квадрат (для ДЛК с максимально известным числом трансверсалей (198 144) – более недели в 1 поток на Core i7 4770). Для размерности N=13 самые «легкие» по числу трансверсалей квадраты будут диагонализироваться несколько часов, «тяжелые» – до нескольких месяцев, что неприемлемо.

Схематично процедура диагонализации выглядит следующим образом: из множества трансверсалей ЛК находятся подходящие пары трансверсалей, по которым производится ряд целенаправленных перестановок строк и столбцов исходного ЛК с целью получения результирующего ДЛК, что на много порядков быстрее построения перестановок всех возможных пар строк и столбцов. Далеко не все пары трансверсалей подходят для выполнения преобразования, однако проверять необходимо все, при этом пары (T_i, T_j) и (T_j, T_i), $i \neq j$, проверять дважды смысла нет, что при изображении соответствия трансверсалей в виде бинарной матрицы (0 – не подходят для диагонализации, 1 – подходят) необходимо обойти не всю матрицу, а ее половину – верхнюю или нижнюю треугольную подматрицу (для определенности обходится верхняя). В программировании подобный обход применяется очень часто в ряде алгоритмов, а соответствующий псевдокод выглядит следующим образом:

```
for (int i = 0; i < NT; i++)
  for (int j = i+1; j < NT; j++)
    ...
```

Простейшим способом распараллеливания является разбиение по внешнему циклу (по i), однако при этом возникает проблема, связанная с тем, что в таком случае в каждом расчетном задании (англ. Work Unit, сокр. WU) потребуется хранить полное множество трансверсалей (для порядка N=13 их уже бывает более миллиона, для больших порядков их количество может быть сильно большим[1]), что потребует минимум сотен МБ – единиц–десятков ГБ оперативной памяти на WU. Более подходящей представляется следующая стратегия распараллеливания: матрица разбивается на квадраты заданного размера $K \times K$, в рамках одного WU производится обработка одного квадрата (см. рис., на нем изображен один из самых «легких» ДЛК с

[1] https://oeis.org/A287644

N_T » 43 тыс. трансверсалей и разбиение на WU по 20 тыс. × 20 тыс. трансверсалей в квадрате, всего 6 WU).

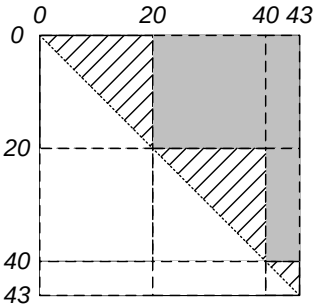

Рис. Схематичное изображение разбиения исходной задачи на подзадачи

При программной реализации в соответствии с данной стратегией возникает ряд особенностей.

1. Необходимость построения разбиения на квадраты, некоторые из которых частично попадают под главную диагональ матрицы – в составе соответствующих WU необходима проверка и отсечение пар трансверсалей из нижней треугольной подматрицы (обрабатываемая часть выделена штриховкой, полностью обрабатываемые квадраты/прямоугольники выделены серым).
2. Наличие маленьких прямоугольников и квадратов по краям (им будут соответствовать более короткие WU с рядом дополнительных проверок, чтобы не выйти за пределы анализируемой области вправо и вниз).
3. Необходимость хранения множества трансверсалей по частям (точнее, в двух частях в диапазонах $[x_{\min}, x_{\max}]$ и $[y_{\min}, y_{\max}]$, где $x_{\min}, y_{\min}$ – координаты верхнего левого угла анализируемого квадрата/прямоугольника, $x_{\max} = \min(x_{\min} + K, N_T)$, $y_{\max} = \min(y_{\min} + K, N_T)$, диапазоны иногда могут пересекаться (в данной задаче – совпадать полностью, в перспективе в ряде других практических применений в задачах по эвристической работе со спектрами – частично пересекаться).

Неоспоримым плюсом подхода является его универсальность: на квадраты можно разбить как «легкие», так и «тяжелые» ДЛК, в последнем случае возрастет лишь число подзадач (для ДЛК порядка 13 с максимально известным числом трансверсалей – около 2500 WU при текущем способе разбиения), при этом возможно управлять средним временем счета WU и затратами памяти путем изменения размера квадрата K (критичным в данной задаче является именно время счета, которое при организации расчета в проекте грид-вычислений не желательно делать как сильно маленьким (минуты и меньше), так и сильно большим (десятки часов)).

Кроме описанного выше распараллеливание также допустимо по парастрофическим преобразованиям (по 3 на ДЛК из 6 возможных за исключением транспонирования, т.к. оно дает ДЛК из того же главного класса и не представляет интереса) и т.н. slice'ам для каждого из них (также 3, итого 3г 3= 9 комбинаций).

В соответствии с вышеизложенным была разработана программная реализация, в которой для 200 «легких» ДЛК было сформировано около 10 тыс. WU, которые были обсчитаны в проекте добровольных распределенных вычислений RakeSearch[2]. При $K = 20Ч0^3$ среднее время счета составило 7–8 минут на WU, максимальное – 20 минут. В перспективе дальнейших исследований планируется увеличение значения K с целью увеличения времени счета WU и выполнение диагонализации всех ДЛК, входящих в состав спектров (на данный момент спектр числа диагональных трансверсалей включает в своем составе 12751 элемент, числа трансверсалей общего вида – 75265 элементов, числа интеркалятов – 152 элемента).

При организации подобного распределенного расчета в грид есть еще одна особенность, связанная с построением необходимого множества трансверсалей: его можно однократно получить в процессе генерации WU, а затем передать на клиент, а можно получить первым этапом

[2] https://rake.boincfast.ru/rakesearch/

при обсчете WU. В данном эксперименте выбран второй вариант (на генерацию нужных трансверсалей требуется 1–2 секунды в каждом WU, при этом происходит экономия нескольких сотен КБ исходных данных WU. В перспективе с ростом размерности задачи N вполне вероятна ситуация, в которой получение нужных подмножеств трансверсалей подобным образом в каждом WU станет неприемлемо долгим и потребуется переход к первому варианту.

Литература

1. Keedwell A.D., Dénes J. Latin Squares and their Applications. Elsevier, 2015. 438 p. DOI: 10.1016/C2014-0-03412-0.

2. Ватутин Э.И., Титов В.С. и др. Оценка мощностей спектров быстровычислимых числовых характеристик диагональных латинских квадратов порядков N>9 // Наука и образование в развитии промышленной, социальной и экономической сфер регионов России. Муром, 2022. С. 314–315.

3. Ватутин Э.И., Титов В.С. и др. Эвристический метод построения аппроксимаций спектров числовых характеристик диагональных латинских квадратов // Интеллектуальные информационные системы: тенденции, проблемы, перспективы (ИИС – 2022). Курск: изд-во ЮЗГУ, 2022. С. 35–41.

4. Ватутин Э.И., Никитина Н.Н. и др. Методы построения спектров быстровычислимых числовых характеристик диагональных латинских квадратов // Национальный суперкомпьютерный форум (НСКФ – 2022). Принята к опубликованию.

Долгов[1] А.А., Курочкин[1,2] И.И.
[1]*Национальный исследовательский технологический университет МИСИС*
119049, Москва, Ленинский пр-кт, д. 4, стр. 1
[2]*Институт проблем передачи информации им. А.А. Харкевича РАН*
127051, г. Москва, Большой Каретный переулок, д.19 стр. 1
E-mail: kurochkin@iitp.ru

Использование мобильных устройств в грид-системах из персональных компьютеров

Распределенные вычисления для решения научных вычислительных задач набрали популярность в последние десятилетия. Распределенные вычислительные системы или грид-системы позволяют агрегировать географически разнесенные и разнородные ресурсы от разных организаций для решения комплексных вычислительных задач. В связи с популярностью использования мобильных устройств в последние годы, а также стремительным ростом вычислительных возможностей этих устройств, смартфоны и планшеты способны проводить ресурсоемкие операции, что позволяет рассматривать их как поставщиков вычислительных ресурсов. В течение последнего десятилетия во многих исследовательских проектах была решена проблема включения мобильных устройств в сеть. Появилось различное программное обеспечение для организации грид-систем, например BOINC – программная платформа с открытым исходным кодом. Платформа BOINC позволяет разворачивать проекты распределенных вычислений, где в качестве вычислительных узлов могут использоваться, как персональные компьютеры, ноутбуки, так и мобильные устройства на базе ОС Android.

Был развернут тестовый проект распределенных вычислений parlea.ru/boinctest, в котором в качестве вычислительных узлов могут выступать как мобильные устройства, так и персональные компьютеры, и ноутбуки. Проект охватывает архитектуры процессоров: x86, aarch64, arm, x86-64. Вычислительное приложение, решающее одну из многочисленных задач по поиску ортогональных диагональных латинских квадратов [1], доступно для нескольких операционных систем, а именно: Windows, Linux, Android. На данном проекте было проведено несколько вычислительных экспериментов с различными серверными параметрами на грид-системе из персональных устройств разных типов. Произведена настройка параметров серверной части проекта распределенных вычислений, которая позволила уменьшить время проведения вычислительных экспериментов, увеличить процент загруженности устройств, уменьшить процент просроченных заданий [2].

На текущий момент на проекте запущен вычислительный эксперимент, длительностью в несколько недель, в котором активно участвует около 300 различных устройств. Динамику подключения узлов к проекту можно увидеть на рис. 1.

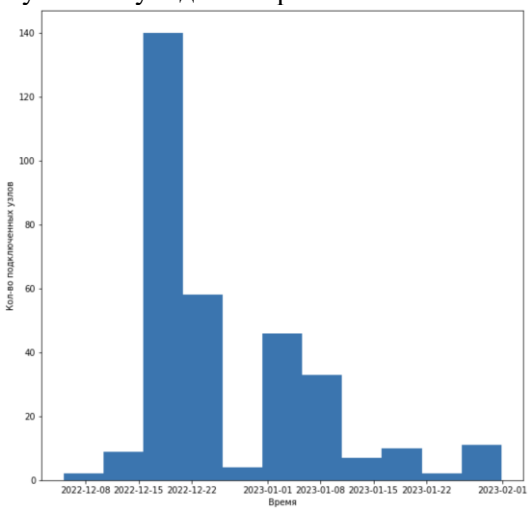

Рис.1. Динамика подключения узлов к проекту.

В текущем эксперименте необходимо просчитать более миллиона рабочих единиц разной сложности (от нескольких секунд до нескольких часов). Исходя из этого, необходимо увеличить количество получаемых и запасаемых задач на узлах. Для этого были убраны жесткие ограничения у параметров max_wus_to_send и max_wus_in_progress повысив их значение до 200 и 100 соответственно. Также в рамках текущего эксперимента была разблокирована возможность получения одним хостом одного юзера более одного результата конкретной рабочей единицы (параметры one_result_per_user_per_wu и one_result_per_host_per_wu) для того, чтобы повысить темп возвращаемых результатов. В публичном проекте этого делать не стоит, поскольку возможны намеренно испорченные результаты со стороны пользователей. Было установлено, что часть возвращаемых результатов (около 5%), были возвращены на сервер с ошибкой, связанной с неправильным ограничением максимальной вычислительной мощности для рабочей единицы. Параметр rsc_flops_bound был повышен на несколько порядков до $8.25*10^{25}$ для того, чтобы свести к минимуму влияние таких результатов на общий эксперимент. Минимальный размер кворума, а также параметр репликации: min_quorum, target_nresults равняются 2 соответственно, чтобы кратно не увеличивать множество рабочих единиц, но при этом сохранить возможность валидирования результатов. Параметр крайнего срока выполнения(deadline) delay_bound остался неизменным и численно равен 5 дням.

Чтобы избежать потери большого массива данных, полученных в результате вычислительного эксперимента, на проекте предусмотрен механизм двухэтапного бекапа, дважды в день архивированные результаты сохраняются на внешний жесткий диск и дважды в сутки результаты выгружаются с сервера проекта.

В процессе эксперимента дополнительно были использованы возможности пакета boinc2docker, который использует Docker контейнеры в качестве оберток для вычислительных приложений. Данное решение было интегрировано в проект и запущено асинхронно второе вычислительное приложение с несколькими сотнями простых задач. Все множество задач было успешно посчитано и не возникло конфликтов с первым вычислительным приложением. Этот механизм значительно упрощает работу, необходимую для разработки и развертывания приложений для BOINC. Он работает, комбинируя возможности boot2docker и vboxwrapper. Главные плюсы данного решения:

- позволяет избежать использования BOINC API в вычислительном приложении;
- контрольная точка автоматически предоставляется vboxwrapper;
- вычислительное приложение автоматически работает в Windows, Mac OS и Linux;
- можно удобно создавать свои приложения с помощью Dockerfiles.

На основе полученного результата дополнительно планируется реализация вычислительного приложения для решения задачи распределенного обучения нейронной сети.

Полученный опыт, а также подобранные параметры в процессе эксперимента, можно применять в организации комплексных вычислительных экспериментов на грид-системах на базе BOINC, с приоритетом не только на персональные компьютеры, но и на мобильные устройства.

Литература

1. Vatutin E., Belyshev A., Kochemazov S., Zaikin O., Nikitina N. Enumeration of isotopy classes of diagonal Latin squares of small order using volunteer computing // Communications in Computer and Information Science. Vol. 965, Springer, 2018. pp. 578–586.

2. Kurochkin, I., Dolgov, A., Manzyuk, M., Vatutin, E. Using mobile devices in a voluntary distributed computing project to solve combinatorial problems // In: Voevodin, V., Sobolev, S. (eds) Supercomputing. RuSCDays 2021. Communications in Computer and Information Science, Vol 1510, Springer, Cham, 2021. pp. 525-537.

Курочкин И.И.
Институт проблем передачи информации им. А.А. Харкевича РАН
127051, г. Москва, Большой Каретный переулок, д.19 стр. 1
E-mail: kurochkin@iitp.ru

Классификация большого множества текстурных изображений методами машинного обучения

Задача классификации изображений является достаточно распространенным типом задач. Для ее решения часто используются методы машинного обучения и, в частности, глубокие нейронные сети. Решение задач классификации изображений необходимо в различных областях от медицины и распознавания материалов, до анализа спутниковых снимков. Одним из преимуществ использования глубоких нейронных сетей в задаче классификации является возможность работы с большим количеством классов. Но это накладывает существенные ограничения на формирование датасета – множества изображений, которое используется для обучения глубокой нейронной сети. Для решения прикладных задач очень часто формируются достаточно большие множества изображений – от нескольких тысяч до сотен миллионов изображений.

Популярность использования глубоких нейронных сетей для решения широкого круга задач привело к возникновению большого количества архитектур нейронных сетей, разработанных специально для решения определенного вида прикладных задач. Одним из популярных видов изображений являются текстурные изображения, на которых представлены одна или несколько видов текстур. В отличие от изображений объектов, текстуры могут не иметь четких границ, а их форма может не нести дополнительной информации. Задачи обработки и анализа текстурных изображений часто возникают при обработке данных, полученных с помощью методов микроскопии. Для решения некоторых задач классификации текстур могут использоваться глубокие нейронные сети с небольшим числом слоев свертки. Это связано с отсутствием необходимости выделять высокоуровневые признаки, которые могут присутствовать в больших объектах, но их не будет в изображениях текстур. В качестве примера, можно привести глубокую нейронную сеть T-CNN [1] для классификации текстур.

Далее будет рассматриваться задача классификации изображений текстур, полученных с помощью оптического микроскопа. Изначально были получены порядка 650 исходных изображений размера 2272x1704 точек в градациях серого.

Опишем порядок формирования текстурного датасета для обучения нейронной сети. Пусть есть множество сходных изображений размером 2272x1704, каждое из которых имеет несколько числовых и категориальных признаков. При этом количество классов может варьироваться от 2 до 36, в зависимости от используемых признаков. Пусть глубокая нейронная сеть с архитектурой T-CNN имеет входной слой размерностью 256x256, тогда из каждого изображения 2272x1704 получится 9*7=63 непересекающихся фрагмента, при 50% наложении фрагментов получится 17*13=221 фрагмент. Так как ориентация текстуры на изображении неизвестна, то каждый фрагмент нужно представить в виде множества фрагментов повернутых на различные углы: при повороте одного фрагмента изображения на угол от 0^o до 345^o с шагом 15^o получается 24 фрагмента. В результате, датасет для обучения нейронной сети сформированный из 650 исходных изображений будет состоять из 3 447 600 фрагментов изображений.

Обучение даже небольшой глубокой нейронной сети с архитектурой T-CNN на датасете размера порядка 3.5 миллиона изображений является вычислительно сложной задачей. Такой объем вычислений провести за разумное время возможно только при использовании высокопроизводительных кластерных систем или распределенных систем. Для реализации обучения глубокой нейронной сети на распределенной системе нужно реализовать один из подходов к распределенному глубокому обучению [2]: синхронный, асинхронный или децентрализованный. При использовании грид-системы на базе платформы BOINC лучше использовать синхронный или асинхронный подход с параметрическим сервером [3]. В процессе распределенного глубокого обучения на параметрическом сервере будет находиться

актуальная глобальная модель нейронной сети. Процесс обучения будет проходить итеративно, от использования того или иного подхода будет меняться методика обновления глобальной модели с помощью результатов обучения локальных моделей после каждой итерации.

Однако при реализации любого из подходов к распределенному обучению глубокой нейронной сети будет необходимо разделять исходный датасет на части. При этом на вычислительном узле грид-системы локальная модель глубокой нейронной сети может быть обучена на одной итерации только на небольшом количестве (~1000) изображений. Если предположить, что грид-система имеет 200 вычислительных узлов, то для участия всех изображений датасета понадобится порядка 18 итераций. Это возможно в случае последовательного формирования локальных датасетов перед каждой итерацией. При использовании методики случайного формирования локальных датасетов количество итераций увеличивается. Данные оценки позволяют сделать вывод, что общее количество итераций глобальной модели обучения нейронной сети будет не менее 500-1000.

Формирование локальных датасетов может происходить несколькими способами:

- Стохастическое (случайное) формирование локального датасета;
- Разделение датасета на части (статическое) до конца обучения;
- Разделение датасета на части (динамическое) после каждой итерации;
- Гибридное формирование датасета с использованием как случайного выбора, так наличия части изображений из локального датасета прошлой итерации.

Стоит заметить, что при любом способе формирования локального датасета должен соблюдаться баланс классов. Динамическое разделение датасета между узлами предполагает добавление новых данных на каждой итерации. Схему формирования локального датасета при динамическом разделении можно увидеть на рис. 1.

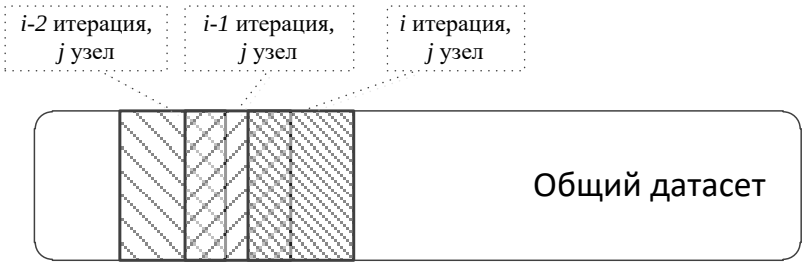

Рис.1. Формирование локального датасета при динамическом разделении.

В случае применения способа гибридного формирования в локальном датасете присутствует как динамическая часть, так и случайная часть (рис. 2).

Рис.2. Части локального датасета при гибридном подходе.

Гибридный способ формирования локального датасета позволяет с одной стороны задействовать весь общий датасет, а с другой позволяет не изменять полностью локальный датасет между соседними итерациями обучения. При применении гибридно способа скорость обучения возрастает.

Литература

1. Andrearczyk, Vincent, and Paul F. Whelan "Using filter banks in convolutional neural networks for texture classification." // Pattern Recognition Letters, Vol.84, 2016. pp.63-69.

2. Ben-Nun, Tal, and Torsten Hoefler. "Demystifying parallel and distributed deep learning: An in-depth concurrency analysis." // ACM Computing Surveys (CSUR) Vol.52(4), 2019. pp.1-43.

3. Kurochkin I.I., Kostylev I.S. Solving the problem of texture images classification using synchronous distributed deep learning on desktop grid system // Communications in Computer and Information Science, Vol 1331, Springer, Cham, 2020. pp. 647-657.

Курочкин[1,2] И.И., Богомолов[1] В.Ю., Кувшинов[1] М.Е.
[1]Национальный исследовательский технологический университет МИСИС
119049, Москва, Ленинский проспект, д. 4, стр. 1
[2]Институт проблем передачи информации им. А.А. Харкевича РАН
127051, г. Москва, Большой Каретный переулок, д.19, стр. 1
E-mail: kurochkin@iitp.ru

Методы анализа снимков сельскохозяйственных полей

Методы машинного обучения применяются в различных отраслях, в том числе и в сельском хозяйстве. Данные методы применяются для решения различных задач: обнаружение болезней растений, классификация культур, идентификация сорняков, определение спелости и подсчет плодов, анализ водных ресурсов и почвы [1]. В тоже время, в сельском хозяйстве нашли свое применение беспилотные летательные аппараты (БПЛА) и спутниковые системы. Регулярный мониторинг с помощью спутниковых систем и аэрофотосъемки позволяет своевременно реагировать на изменения на сельскохозяйственных полях.

Спутниковые снимки позволяют эффективно производить мониторинг полей, следить за состоянием посевов, выявлять потенциальные угрозы. Особенностью данной технологии является то, что она позволяет в режиме реального времени наблюдать за состоянием посевов на большой территории или даже полях, находящихся на расстоянии, в разных областях, регионах, странах, на разных континентах, а также возможность формировать обширную статистическую базу для сравнений на основе исторических данных мониторинга. Преимуществами спутникового мониторинга являются: возможность мониторинга большой площади полей, возможность организации высокого уровня автоматизации наблюдения за посевами, а также автоматический анализ и интерпретация данных.

Использование снимков аэрофотосъемки с помощью БПЛА с низких высот 10-50 метров дает возможность анализировать данные, которые не видны на спутниковых снимках. Так на снимках с БПЛА можно увидеть отдельные растения и их листья. Такая информация дает возможность определять на ранних стадиях болезни растений и отклонения их развития.

После обработки исходных снимков помимо получения основных типов данных: ортофотоплан и карт отражений, можно еще получить и карты вегетационных индексов, таких как NDVI, SAVI, WI, NDWI, WDRVI, ARI, TCARI, MSAVI и др. Для вычисления вегетационных индексов необходимо производить съемку с помощью мультиспектральных камер. Мультиспектральные снимки позволяют получать больше информации о состоянии растительности, на основе ее отражательной способности в различных диапазонах. С помощью мультиспектральных изображений моно решать задачи идентификации типов растительности и их состояния - ведь различное состояние листовой поверхности (обезвоженность, недостаток элементов питания, вирусная деградация, поражённость от вредителей и заболевания и т.п.). Одним из самых популярных вегетационных индексов является NDVI [2]. NDVI (Normalized Difference Vegetation Index) – показатель количества фотосинтетической активной биомассы на земной поверхности (оценка интенсивности вегетации растений). Для расчёта индекса NDVI необходимы данные с двух каналов на мультиспектральной камере, улавливающих отраженный спектр в красном и ближнем инфракрасном диапазонах. Формула вычисления NDVI:

$$\mathrm{NDVI} = \frac{(\mathrm{NIR} - \mathrm{Red})}{(\mathrm{NIR} + \mathrm{Red})}$$

где NIR – значение инфракрасного диапазона, Red – красного диапазона.

Использование вегетационных индексов для анализа состояния полей является достаточно популярным методом, но перед получением данных мониторинга полей для анализа требуется провести ряд этапов:

1. Планирование маршрута и организация полета БПЛА;
2. Обработка изображений с помощью дополнительных датчиков, в том числе проведение радиометрической коррекции;
3. Предобработка исходных изображений;

4. Создание ортофотоплана, путем склейки исходных изображений;
5. Сегментация изображения для отделения полей от прочих объектов;
6. Построение карты индексов с помощью мультиспектральных каналов камеры.

Исходные снимки плохо подходят для создания ортофотоплана, поскольку при склейке будут видны артефакты, а также достоверность значений во всех каналах изображения будет стоять под вопросом. К факторам, которые могут искажать исходные изображения, могут относиться:

- тени от облаков;
- погодные условия;
- температура;
- время дня и дата.

Влияние каждого фактора можно нивелировать с помощью алгоритмов и корректировкой организации съемки. К примеру, есть ряд алгоритмов для детекции и компенсации теней от объектов. Они делятся на пороговые методы и на методы классификации.

Методы, основанные на порогах, часто использовали либо предопределенные пороговые значения, либо адаптивные пороговые значения для пометки облаков на обработанных изображениях. Все пороговые методы имеют ряд недостатков – необходима точная настройка, так как пороги могут сильно меняться из-за локаций, камеры, и присутствует верхний порог по точности. Классификация изображений, основанная на извлечении признаков и методах машинного обучения, также является эффективным методом обнаружения облаков и их теней. Как правило, методы машинного обучения дают более точные результаты обнаружения облаков/теней, чем пороговые методы. Но следует учесть необходимость в обучении модели для конкретных условий съемки. В последние годы, часто встречается использование глубоких нейронных сетей, которые дают лучшие показатели точности классификации. На данный момент одним из лучших решений является многомасштабная глубокая нейронная сеть 3D-CNN [3], которая достигает около 96% точности для детекции облаков и около 96% точности для детекции теней облаков. Данная архитектура глубокой нейронной сети принимает на вход все мультиспектральные каналы изображения, что позволяет отличать облака от схожих объектов, таких как снег или белые здания.

Литература

1. Гатаулина Г. Г., Заренкова Н. В., Шитикова А. В. «О системном подходе к анализу формирования урожая зернобобовых культур» // Современное состояние и перспективы исследований сои. – 2020. – с. 119-131.

2. Hassan M. A. et al. A rapid monitoring of NDVI across the wheat growth cycle for grain yield prediction using a multi-spectral UAV platform // Plant science. – Vol. 282, 2019. pp. 95-103.

3. Chen Y. et al. Cloud and cloud shadow detection based on multiscale 3D-CNN for high resolution multispectral imagery // IEEE Access. – Vol. 8, 2020. pp. 16505-16516.