ISSN 2079-3316 ПРОГРАММНЫЕ СИСТЕМЫ: ТЕОРИЯ И ПРИЛОЖЕНИЯ № ?(??), 2015, с.??—??

УДК 51-76

О. С. Алтухова, П. И. Боровиков

Использование технологий параллельных вычислений для анализа данных NGS в задаче определения таксономической принадлежности прокариот

Аннотация. В задаче определения таксономической принадлежности прокариот возникает необходимость вычисления множества парных выравниваний последовательностей, например, при кластеризации или поиске в базе данных. В работе описан подход к распараллеливанию алгоритма такой задачи, с точки зрения эффективного использования ресурсов центрального процессора и графических ускорителей.

Ключевые слова и фразы: парное выравнивание последовательностей, динамическое программирование, кластеризация последовательностей, GPU, CUDA, OpenMP.

Введение

Изучение микробного сообщества (микробиоты) важно с точки зрения выработки эффективных подходов к профилактике, лечению и диагностике различных заболеваний. Классический метод изучения основан на изолировании бактерии на селективной среде, получении штамма и его культивировании. Однако такой метод неточно отражает структуру бактериального сообщества. Кроме того, только 10% бактерий может быть культивировано стандартными технологиями. Преодолеть эти недостатки позволило развитие молекулярных методов изучения бактерий, опирающихся на их генетическую информацию.

Технология высокопроизводительного секвенирования (NGS) способствовала огромному скачку в объеме генерируемой информации. За запуск прибора генерируется от 10^5 до 10^7 ридов (прочтений), при этом снизилась цена секвенирования. В следствие это-

го нарастает потребность в новых биоинформатических алгоритмах, позволяющих обрабатывать большие массивы данных за приемлемое время. Кроме того, для различных платформ высокопроизводительного секвенирования характерны разные ошибки секвенирования и возникает необходимость возможности настройки алгоритмов в зависимости от используемой платформы и участков ДНК, используемых для определения видовой принадлежности.

1. Метод динамического программирования для выравнивания последовательностей

Основная задача анализа последовательностей — решить, родственны ли две последовательности. Для этого две данные последовательности (или их части) выравниваются, а затем решается, говорит ли данное выравнивания о родственности последовательностей, или такой результат можно ожидать, если выровнять две случайно взятые последовательности [1]. В молекулярной биологии такая задача решается для цепочек ДНК, РНК и т. д. Математической моделью таких последовательностей является случайная цепочка символов над конечным алфавитом. В данной работе алфавит состоит из четырех символов, обозначающих нуклеотиды: A — аденин, T — тимин, G — гуанин, C — цитозин.

Элементарными операциями, с помощью которых можно получить одну последовательность из другой, являются замены, вставки и делеции (разрывы). Общий вес выравнивания при использовании аддитивной схемы вычисляется следующим образом:

$$s = \alpha m + \beta n + \gamma k\,,$$

где α, β, γ — весовые коэффициенты, m, n, k — количество совпадений, замен и разрывов соответственно.

Для того, чтобы найти лучшее выравнивание с помощью метода динамического программирования, необходимо максимизиро-

вать общий вес выравнивания. Далее приводится алгоритм глобального выравнивания в случае перекрывающихся выравниваний.

(1) Инициализация матрицы динамического программирования F размером (n+1)×(m+1), где n и m длины последовательностей:

$$\begin{cases} F_{i,0} = 0 \\ F_{0,j} = 0 \end{cases}$$

(2) Заполнение матрицы F:

$$F_{i,j} = max \begin{cases} F_{i-1,j-1} + S_{A_i B_j} \\ F_{i-1,j} + d \\ F_{i,j-1} + d \end{cases},$$

где A_i — i-тый символ последовательности A, B_j — j-тый символ последовательности B, S — матрица замен, d — штраф за разрыв.

(3) Нахождение веса наилучшего выравнивания:

$$s = \max(F_{i,m}, F_{n,j})$$

(4) Построение выравнивания с помощью процедуры обратного прохода.

Сложность алгоритма по времени и используемой памяти O(nm).

2. Применение технологий OpenMP и CUDA в задаче о множестве парных выравниваний последовательностей

Вычисление степени схожести последовательностей можно разделить на три этапа:

- заполнение матрицы динамического программирования,
- построение выравнивания с помощью процедуры обратного прохода,
- вычисление степени схожести по выравниванию.

Самая большая вычислительная сложность — квадратичная возникает на первом этапе. Т.к. матрица динамического програм-

мирования заполняется рекурсивно, невозможно вычислять каждый элемент параллельно. На рисунке 1 показаны зависимости элементов матрицы динамического программирования, а также выделены элементы, которые могут быть вычислены независимо друг от друга.

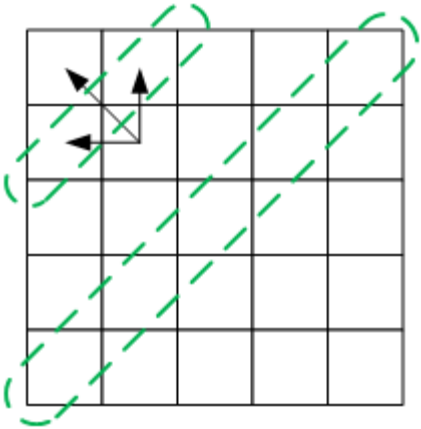

РИС. 1. Матрица динамического программирования

В случае, когда необходимо выполнить множество парных выравниваний возможно два подхода:

- параллельно вычислять некоторые элементы одной матрицы,
- параллельно вычислять некоторое количество матриц.

При втором подходе используется больше памяти, но он более эффективен по быстродействию [2]. Т.к. в процедуре обратного прохода содержится множество условных переходов, что может понизить быстродействие при вычислении на графических ускорителях, целесообразнее проводить вычисления второго и третьего этапа средствами CPU. Для вычислений на GPU используется технология CUDA, на CPU OpenMP.

Вследствие этого распределение задач между CPU и GPU выглядит следующим образом:

- заполнение матриц динамического программирования производить средствами CPU и GPU;
- выполнение процедур обратного прохода и вычисление степеней схожести последовательностей производить средствами CPU.

Для того, чтобы эффективно задействовать ресурсы и центрального процессора, и графических ускорителей предлагается схема вычислений, представленная на рисунке 2.

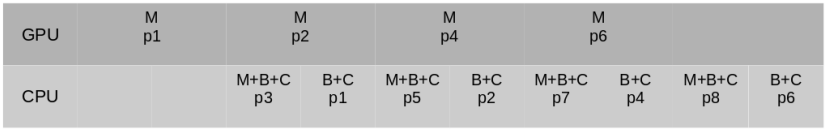

РИС. 2. Схема вычислений

Также была реализована процедура тестирующая производительность системы и определяющая параметры запуска алгоритма вычисления множества парных выравниваний, такие как:

- размер сетки из блоков и нитей для GPU,
- количество выравниваний для расчета на CPU.

Далее представлен алгоритм вычисления оптимального размера сетки из блоков и нитей.

(1) Считывание свойств устройства GPU, участвующих в дальнейших расчетах.
(2) Расчет количества блоков, запускаемых в сетке:

$$b = \frac{B_{SM} * C_{SM}}{q},$$

где B_{SM} — количество блоков на мультипроцессор (зависит от версии спецификации), C_{SM} — количество мультипроцессоров, q — количество одновременно исполняемых процедур.
(3) Определение максимального количества нитей, запускаемых в сетке:

$$T_{max} = \frac{DP_{max}}{2qb},$$

где DP_{max} — максимально возможное количество матриц динамического программирования.

(4) Запуск процедуры множества парных выравниваний с фиксированием времени выполнения для различных конфигураций сеток (числа, являющиеся степенью двойки в диапазоне $[8, T_{max}]$).

При распределении нагрузки между CPU и GPU удалось добиться производительности 1,2 GCUPS (количество вычисляемых элементов матрицы в секунду, деленное на 10^9). Расчеты производились на двух графических картах Tesla k20m и двух 12-ти ядерных процессорах Intel Xeon E5-2697 v2. Для сравнения на рисунке 3 представлена производительность для задачи множества парных выравниваний отдельно на GPU и CPU:

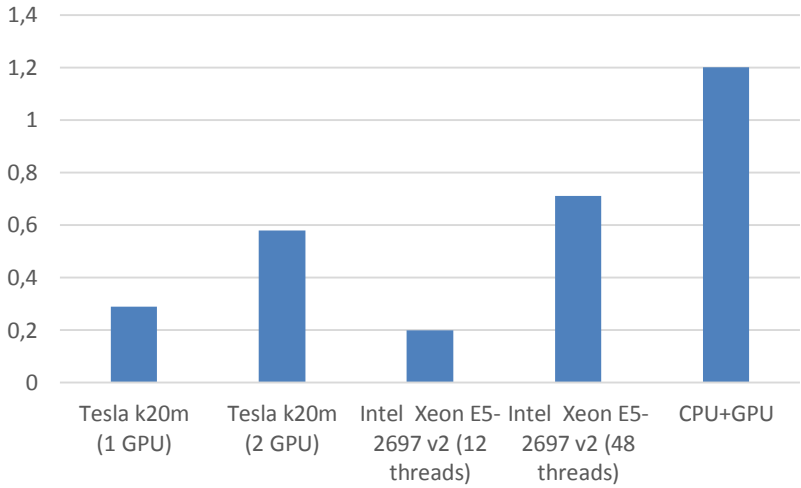

РИС. 3. Производительность в GCUPS

Необходимость вычисления множества выравниваний возникает при кластеризации последовательностей и дальнейшего поиска в базе данных.

3. Кластеризация последовательностей

Одним из популярных жадных алгоритмов кластеризации биологических последовательностей является CD-HIT [3]. Последовательности сортируются по убыванию длины и самая длинная выбирается представителем кластера. Затем каждая оставшаяся последовательность сравнивается с представителями кластеров. Если степень схожести превышает заданный порог, то последовательность добавляется в кластер, иначе становится представителем нового кластера. Такой алгоритм работает быстрее, чем алгоритмы, в которых строится матрица смежности между последовательностями.

Чтобы использовать массивный параллелизм в множестве выравниваний в данной работе используется измененный подход. На каждой итерации выбирается последовательность и выравнивается со всеми остальными последовательностями. Все последовательности, степень схожести которых при выравнивании превышает заданный порог, объединяются в один кластер и исключаются из дальнейшей кластеризации. Таким образом, количество исходных последовательностей уменьшается.

4. Заключение

В данной работе был разработан программный комплекс, эффективно использующий ресурсы ЭВМ для определения класса таксономической принадлежности прокариот с помощью поиска наиболее близких последовательностей в базах данных, с использованием метода динамического программирования для выравнивания последовательностей. С помощью кластеризации последовательностей и дальнейшего построения консенсусов удалось добиться значимого уменьшения исходного объема данных и исключения

ошибок секвенирования. Был использован «жадный» алгоритм кластеризации, который работает быстрее, чем алгоритмы, в которых строится матрица смежности между последовательностями. Благодаря изменяемым параметрам, а именно весовых коэффициентов за совпадения, замены, разрывы, минимальной длины выравнивания, способу расчета степени схожести, можно настроить алгоритм под конкретную платформу высокопроизводительного секвенирования.

Реализованная процедура поиска наиболее схожих последовательностей в базе данных использует выравнивание с помощью динамического программирования, что гарантирует построение оптимального выравнивания. Т.к. матрица динамического программирования заполняется рекурсивно, невозможно вычислить все её элементы параллельно, однако существует способ расчета, при котором последовательно вычисляются группы элементов матрицы, внутри которых вычисления производятся независимо друг от друга. Но в данной задаче такой способ уступает по производительности методу, при котором элементы матрицы рассчитываются последовательно, а параллелизм используется при расчете множества таких матриц.

Разработанная методика кластеризации может применяться и для других задач, требующих классификации большого объема полученных данных. Например, для анализа данных практически любого таргетного секвенирования, как универсальных генов каких-либо организмов с последующим определением таксономической принадлежности, так и отдельных генов человека с поиском полиморфизмов, характерных для данного образца и определением аллельных вариантов. Причем, в отличие от классического выравнивания на референсный геном, данный подход позволяет определять сцепленность полиморфизмов на длине рида. Кроме того, методика может применяться для поиска нежелательных часто встречающихся последовательностей в данных высокопроизводительного секвенирования и определения их источника, что помога-

ет увеличить долю полезных данных в эксперименте и снизить стоимость исследований.

Программный комплекс был адаптирован и используется также для задачи типирования HLA (человеческих лейкоцитарных антигенов).

Список литературы

[1] Vilo C., Dong Q. Evaluation of the RDP Classifier accuracy using 16S rRNA gene variable regions // Metagenomics.2012.1.C.1-5

[2] Liu, D. Maskell, and B. Schmidt. CUDASW++: Optimizing Smith-Waterman Sequence Database Searches for CUDA-enabled Graphics Processing Units // BMC Research Notes, 2(1):73, 2009.

[3] Chen W, Zhang CK, Cheng Y, Zhang S, Zhao H A Comparison of Methods for Clustering 16S rRNA Sequences into OTUs // PLoS ONE.2013.8(8):e70837.