УДК 004.421:004.912

ПАРАЛЛЕЛЬНАЯ РЕАЛИЗАЦИЯ МУЛЬТИЭВРИСТИЧЕСКОГО ПОДХОДА В ЗАДАЧЕ СРАВНЕНИЯ ГЕНЕТИЧЕСКИХ ПОСЛЕДОВАТЕЛЬНОСТЕЙ

Б.Ф. Мельников, доктор физико-математических наук, профессор, профессор кафедры «Прикладная математика и информатика»

А.Г. Панин, кандидат физико-математических наук, старший преподаватель кафедры «Прикладная математика и информатика»

Тольяттинский государственный университет, Тольятти (Россия)

Ключевые слова: мультиэвристический подход; анализ генетических последовательностей; расстояние Левенштейна; выравнивание строк; параллельное программирование.

Аннотация: В работе рассматривается параллельная реализация мультиэвристического подхода к проблеме определения схожести строк, кодирующих молекулы ДНК. Данный подход позволяет получить целый класс метрик на множестве строк, некоторые из которых могут дать интересные результаты при применении их к сравнению ДНК. Предложены несколько эвристик, проведено сравнение результатов их применения, а так же сравнение результатов работы параллельной версии алгоритма с последовательной.

ВВЕДЕНИЕ

Задача определения схожести ДНК является частным случаем задачи неточного сопоставления последовательностей [1]. «Неточность» заключается в том, что при сравнении строк имеется возможность распознать схожие последовательности даже несмотря на возможные ошибки и искажения в них, например, изменение, удаление или вставку нескольких символов. Количество таких искажений задаёт *метрику* на множестве строк, которая определяется по минимальному количеству операций редактирования, позволяющих получить из одной строки другую. Эта задача встречается во многих областях. Например, сравнение генов, хромосом и белков является одной из важнейших задачей и одним из основных инструментов молекулярной биологии и биоинформатики [1, 12]. *Точное* сравнение цепочек нуклеотидов здесь неприемлемо из-за наличия ошибок в данных, а так же из-за возможных мутаций. Неточное сопоставление осуществляется так же при обработке обычного текста. Одна из метрик, получаемая при сравнении слов, – расстояние Левенштейна – используется для исправления ошибок, для повышения качества распознавания отсканированных документов, для поиска в информационных системах, и базах данных [1].

К решению этой задачи существует несколько подходов. Точное решение могут дать алгоритмы, основанные на динамическом программировании – например, алгоритм Ханта-Шиманского, Хиршберга, Вагнера-Фишера, Смита-Ватермана и другие [1, 6, 7, 9, 13]. Как правило, эти алгоритмы имеют квадратичную сложность в худшем случае и оказываются слишком медленными в тех случаях, когда точность результата не так важна, как производительность – например, при поиске информации в базе данных. Для нахождения приближённого решения существуют различные алгоритмы в разных предметных областях, например, для поиска в базах данных генетической информации широко применяется алгоритм BLAST [14], аппроксимирующий алгоритм Нидлмана-Вунша.

Применение мультиэвристического подхода к задаче определения схожести последовательностей ДНК впервые было описано в работе [4]. Степень сходства последовательностей, полученная с помощью мультиэвристического подхода, имеет сходство с расстоянием Левенштейна, но не равна ему и не аппроксимирует его. Алгоритм позволяет получить множество альтернативных метрик (в зависимости от используемых эвристик) на пространстве цепочек нуклеотидов. Эти метрики могут отражать не только различные способы определения степени сходства, но и различное понимание самого понятия сходства.

РЕАЛИЗАЦИЯ МУЛЬТИЭВРИСТИЧЕСКОГО ПОДХОДА

Мультиэвристический подход – подход к решению задач дискретной оптимизации, который заключается в сочетании незавершённого метода ветвей и границ с комбинацией различных эвристик, на результатах работы которых основывается выбор очередного шага. Оценки, полученные эвристиками, усредняются с помощью динамических функций риска. Для подбора коэффициентов усреднения применяются генетические алгоритмы, упрощённое самообучение которыми применяется также и для старта незавершённого метода ветвей и границ [2, 3].

Для решения данной задачи он был применён следующим образом. Пусть x, y – исходные строки, i, j – индексы символов строк x и y соответственно, r – значение метрики, которое требуется найти. Под сдвигом строки понимается увеличение на единицу соответствующего индекса. Алгоритм можно описать следующей схемой.

Вход: Строки x и y.
Шаг 1: i := 0,j := 0,r:=0;
Шаг 2: **if** *x[i] = y[j]* **then begin**
сдвигаем обе строки;
r := *r* + стоимость совпадения символов *x[i]* и *y[j]*;

end
else begin
применяем эвристики для генерации возможных «траекторий» сдвига в позиции *i'* и *j'* таких, что *x[i'] = y[j']*;
оцениваем их с помощью других эвристик;
усредняем полученные оценки, используя функцию риска;
осуществляем сдвиг (при этом может измениться значение *r*);
end;

Шаг 3: повторяем второй шаг до тех пор, пока не достигнут конец одной из строк.

Стоимость совпадения двух символов в простейшем случае равна 1; для ДНК можно определять её с помощью матрицы весов аминокислотных замен BLOSUM [5] или другой подобной матрицы.

Были использованы следующие эвристики:

1. Выбираем траектории, для которых выражение *(i' - i) + (j' - j)* принимает минимальное, либо близкое к минимальному значение. Например, сначала рассматриваем все траектории со сдвигом только одной из строк на один символ; затем – со сдвигом одной из строк на два символа или обеих на один символ и т.д.

2. Сдвигаем ту строку, текущий символ которой реже встречается в другой строке. Для этой эвристики желательно знать вероятности появления символов в каждой из строк. Если они заранее не известны, можно считать их равными. В процессе работы можно корректировать вероятности или использовать алгоритм старения, чтобы вероятность символа определялась по некоторому фрагменту перед текущим символом, а не по целой строке. Если вероятности для текущих символов окажутся равными, сдвигается строка, в которой осталось больше символов.

3. Комбинация двух предыдущих: результирующая оценка позиции складывается из её оценок первой и второй эвристиками. Для определения оценки второй эвристики суммируются вероятности появления в другой строке для всех символов, которые придётся пропустить при сдвиге.

4. Используем алгоритм для поиска наибольшей общей подпоследовательности строк *x[i..i+k]* и *y[j..j+k]*, где *k* ~ 15. Для сдвига выбираем такие индексы *i'*, *j'*, в которых заканчивается наибольшая общая подпоследовательность. Если не будет найдено ни одной пары одинаковых символов, область поиска увеличивается. При использовании этой эвристики результат будет близок к значению наибольшей общей подпоследовательности.

5. Комбинация третьей и четвёртой эвристик: оценка позиции складывается из её оценок обеими эвристиками. Оценка позиции *(i', j')* четвёртой эвристикой является отношением длины наибольшей общей подпоследовательности строк *x[i..i']* и *y[j..j']* к средней длине сдвига строк из позиции *(i, j)* в позицию *(i', j')*.

6. Используем алгоритм Нидлмана-Вунша [10] для строк *x[i..i+k]* и *y[j..j+k]*, где *k* ~ 15. Сдвигаем строки в позицию *(i', j')*, для которой соответствующее значение в таблице алгоритма Нидлмана-Вунша является наибольшим.

7. Комбинация третьей и шестой эвристик: оценка позиции складывается из её оценок обеими эвристиками. Оценка позиции *(i', j')* шестой эвристикой является отношением значения в таблице алгоритма Нидлмана-Вунша, соответствующего этой позиции, к средней длине сдвига строк из позиции *(i, j)* в позицию *(i', j')*.

ПОДХОД К РЕАЛИЗАЦИИ ПАРАЛЛЕЛЬНОЙ ВЕРСИИ АЛГОРИТМА

Основным шагом при разработке параллельного алгоритма является определение информационных зависимостей и выявление независимых подзадач.

Разработанный алгоритм основан на итеративном сдвиге строк в некоторую позицию, определяемую парой индексов (i, j) для первой и второй строки соответственно. Определение очередной позиции осуществляется посредством оценивания некоторого небольшого множества ближайших позиций и выбор позиции с наибольшей оценкой. Очевидно, выполнять сдвиги можно только последовательно, и единственная часть алгоритма, в которой можно выделить независимые подзадачи – применение эвристик для выбора позиции. Можно предложить два способа параллелизации этой части алгоритма: выполнение различных эвристик в различных вычислительных потоках и параллелизация самих эвристик.

Первый способ имеет следующие недостатки. Во-первых, не всегда результат совместного применения нескольких эвристик является их «линейной суммой». В некоторых случаях для эффективного сочетания эвристик их необходимо интегрировать в одну комплексную эвристику. Во-вторых, вычислительная сложность некоторых эвристик является достаточно низкой, и временные затраты на управление вычислительными потоками оказываются слишком значительными.

Основные недостатки второго способа связаны с необходимостью разработки параллельных версий эвристик. В некоторых случаях это оказывается весьма затруднительным, в некоторых эффективность параллельной версии ниже, чем последовательной из-за накладных расходов.

Таким образом, достаточно эффективное выделение независимых подзадач в алгоритме не представляется возможным. Но это вовсе не означает невозможность разработки параллельной версии алгоритма – для этого можно создать подзадачи, которых раньше не было. Для этого нужно на каждом шаге из множества возможных позиций выбирать не одну, а несколько – порождая таким образом ещё одну «фазовую траекторию» в пространстве позиций для сдвига.

Выбирать дополнительные позиции можно различными способами, например, выбирать несколько позиций с наибольшими оценками или использовать для этого дополнительные наборы эвристик.

Подзадачи, порождаемые таким образом, могут иметь фиксированный размер (т.е. количество итераций основного цикла выбора-сдвига). После их завершения можно выбирать некоторое количество лучших результатов и на их основе генерировать новые подзадачи.

Описанная параллельная версия реализации мультиэвристического подхода является, по сути, новым алгоритмом, а предложенный способ порождения новых подзадач можно назвать новой эвристикой. Результат работы описанного параллельного алгоритма отличается от результата работы последовательной версии – он является более точным. В то время как исходный алгоритм получает решение путём «жадного» выбора очередной позиции для сдвига, параллельная версия сравнительно глубоко анализирует последствия сделанного выбора.

РЕЗУЛЬТАТ СРАВНЕНИЯ ЦЕПОЧЕК ДНК

Для тестирования алгоритма были использованы митохондриальные ДНК различных организмов. Эти молекулы ДНК, содержащиеся в митохондриях клетки,

не подвержены рекомбинации и наследуются по материнской линии у большинства многоклеточных организмов, поэтому их изменение может происходить только за счёт мутации. Путём сравнения последовательности митохондриальной ДНК и возникших в ней со временем мутаций можно не только определить степень родства различных видов, но и приблизительно вычислить время, необходимое для накопления мутаций в той или иной популяции [15]. Таким образом, можно вычислить и эпоху, когда мутаций ещё не было, и популяция предков была генетически однородной.

Была использована генетическая информация следующих организмов: Homo sapiens (человек), Pan troglodytes (шимпанзе), Bison bison (бизон), Bos taurus (дикий бык), Sus scrofa taiwanensis (свинья), Canis lupus (волк), Felis catus (домашняя кошка), Gallus gallus (курица), Mus musculus (мышь), Rattus norvegicus (крыса), Orcinus orca (касатка), Orcaella brevirostris (иравадийский дельфин), Peponocephala electra (бесклювый дельфин), Gadus morhua (треска), Drosophila simulans (дрозофила) [16].

По результатам тестирования последовательной версии алгоритма можно сделать следующие выводы. Первые две эвристики дают слабо различающиеся результаты для всех пар ДНК, отобранных для сравнения. Однако, предположения, на которых они основаны, не являются неверными, так как третья эвристика, построенная на их основе, показала достаточно «адекватный» результат, который был близок к результату сравнения с помощью алгоритма поиска наибольшей общей подпоследовательности. Например, степень сходства бесклювого дельфина с иравадийским составила 0.91, с касаткой – 0.87, с остальными хордовыми – от 0.59 до 0.63, с дрозофилой – 0.45. Результат, полученный с использованием четвёртой эвристики, мало отличается от результата, полученного с помощью третьей. Пятая эвристика дала результат, в целом похожий на результат четвёртой эвристики, но менее выраженный в области малых значений. Шестая эвристика дала результат, не позволяющий делать какие-либо выводы о степени сходства ДНК. Седьмая эвристика, полученная путём объединения шестой и третьей эвристики, показала результат, близкий к результату сравнения с помощью алгоритма Нидлмана-Вунша. Причём в области больших значений (больше 0.9) результаты отличались не более, чем на 1%.

Результаты, полученные с помощью параллельной версии алгоритма, в целом повторяют результаты, полученные с помощью последовательной, но смещены в область бóльших значений. При этом результат применения пятой эвристики стал лучше, т.е. ближе к результату применения алгоритма поиска наибольшей общей подпоследовательности. Результат применения седьмой эвристики, наоборот, ухудшился – степень сходства некоторых видов стала некорректной.

В целом результат сравнения для большинства эвристик соответствует таксономии органического мира [11]. При этом алгоритмы работают достаточно быстро, что делает их полезными при анализе генетических данных (время работы алгоритмов является линейным, но зависит от параметров эвристик).

СПИСОК ЛИТЕРАТУРЫ

1. Д. Гасфилд. Строки, деревья и последовательности в алгоритмах. Информатика и вычислительная биология. – СПб: Невский Диалект, БХВ-Петербург. – 2003.
2. Б. Мельников. Мультиэвристический подход к задачам дискретной оптимизации // Кибернетика и системный анализ(НАН Украины), 2006, №3. – С.32–42.
3. Б.Ф. Мельников. Эвристики в программировании недетерминированных игр. // Известия РАН. Программирование, 2001, № 5. – С. 63–80.
4. А. Панин. Применение мультиэвристического подхода к задаче определения схожести ДНК. // Вектор науки ТГУ №3(17). – Тольятти, 2011. – С. 27-29.
5. S. Henikoff, J.G. Henikoff. Amino Acid Substitution Matrices from Protein Blocks. // PNAS 89(22), 1992. – p. 10915–10919.
6. D.S. Hirschberg. A linear space algorithm for computing maximal common subsequences. // Communications of the ACM 18(6), 1975. – p. 341–343.
7. J.W. Hunt, T.G. Szymanski. A fast algorithm for computing longest common subsequences. // Communications of the ACM, Vol. 20, No. 5, 1977. – p. 350–353.
8. B. Melnikov, A. Radionov, V. Gumayunov. Some special heuristics for discrete optimization problems // 8th Int. Conf. on Enterprise of Information Systems(ICEIS-2006), Pathos(Cyprus) – pp. 91–95;
9. E.W. Myers. An Overview of Sequence Comparison Algorithms in Molecular Biology. – 1991.
10. S. Needleman, C. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins // Journal of Molecular Biology, 1970, 48(3). – p. 443–453.
11. A. Shipunov. Systema Naturae or the outline of living world classification // Protistology. 2009. 6(1). – p. 3–13
12. I. Torshin. Bioinformatics in the Post–Genomic Era: The Role of Bio-physics, – Novapublishers, 2006
13. R.A. Wagner, M.J. Fischer. The string-to-string correction problem. // Journal of the ACM, Vol. 21, No. 1, 1974. – p. 168–173.
14. G. Wieds. Bioinformatics explained: BLAST versus Smith-Waterman. –CLCBio, 2007.
15. A. C. Wilson, R. L. Cann, S. M. Carr, M. George Jr., U. B. Gyllensten, K. Helm-Bychowski, R. G. Higuchi, S. R. Palumbi, E. M. Prager, R. D. Sage, and M. Stoneking. Mitochondrial DNA and two perspectives on evolutionary genetics. // Biological Journal of the Linnean Society(1985) 26. – p. 375–400.
16. NCBI Nucleotide database – [URL] http://www.ncbi.nlm.nih.gov/nuccore (дата обращения 29.09.2012)

Работа частично поддержана программой Министерства образования и науки РФ в рамках Госзадания Тольяттинского государственного университета на 2012 год (шифр 6.3072.2011).

Работа частично поддержана региональным грантом РФФИ № 13-01-97003.

THE PARALLEL IMPLEMENTATION OF THE MULTIHEURISTICAL APPROACH IN THE NUCLEOTIDE SEQUENCE COMPARICON PROBLEM

B.F. Melnikov, doctor of physical and mathematical sciences, professor,
professor of department «Applied mathematics and computer science»
A.G. Panin, candidate of physical and mathematical sciences,
lecturer of department «Applied mathematics and computer science»
Togliatti state university, Togliatti (Russia)

Keywords: multiheuristical approach; nucleotide sequences comparison; Levenshtein distance; strings alignment; parallel programming.

Annotation: We consider the parallel implementation of the multiheuristical approach to determine the nucleotide sequences comparison problem. This approach allows getting a class of metrics on the set of strings, some of which can produce interesting results when applied to the DNA comparison. Several heuristics and the results of their application comparison, as well as the parallel and the sequential algorithm comparison are considered in this article.

УДК 519.68

ПРИМЕНЕНИЕ АЛГОРИТМОВ КЛАСТЕРИЗАЦИИ ПОДЗАДАЧ ДЛЯ ВЕРШИННОЙ МИНИМИЗАЦИИ НЕДЕТЕРМИНИРОВАННЫХ КОНЕЧНЫХ АВТОМАТОВ

Е.А. Мельникова, кандидат физико-математических наук,
доцент кафедры «Прикладная математика и информатика»
Тольяттинский государственный университет, Тольятти (Россия)

Ключевые слова: дискретная оптимизация, эвристические алгоритмы, кластеризация ситуаций, минимизация недетерминированных конечных автоматов.

Аннотация: В статье рассматривается применение алгоритмов кластеризации подзадач в специальном мультиэвристическом подходе к задачам дискретной оптимизации. А именно, рассматриваются конкретные варианты применения кластеризации ситуаций в задаче минимизации недетерминированных конечных автоматов.

ВВЕДЕНИЕ

Данную статью можно считать продолжением нескольких предыдущих работ, посвящённых применению кластеризации ситуаций в специальном мультиэвристическом подходе к задачам дискретной оптимизации [1–4]. А именно, рассматриваются конкретные варианты кластеризации ситуаций при применении описанного мультиэвристического подхода в задаче минимизации недетерминированных конечных автоматов (НКА).

Задачу вершинной минимизации недетерминированных конечных автоматов можно свести к одной (но важнейшей) из возникающих в ней вспомогательных подзадач. Для любого НКА можно построить *таблицу соответствия вершин* автоматов, являющихся каноническими для исходного и зеркального автоматов [3]. Тогда задачу минимизации можно сформулировать следующим образом. Задана прямоугольная матрица, заполненная элементами 0 или 1. Некоторую пару подмножеств строк и столбцов назовём блоком, если

- на их пересечениях стоят только 1;
- множество нельзя пополнить ни строкой, ни столбцом, не нарушив первого свойства.

Допустимым решением является множество блоков, покрывающих все элементы 1 заданной матрицы. Стоимость решения – количество содержащихся в нем блоков. Цель – выбрать допустимое решение, содержащее минимальное число блоков.

Структура данной статьи следующая. В разделе 2 описывается возможный подход к сравнению эффективности эвристических алгоритмов. В разделе 3 приведены примеры метрик, которые применяются для кластеризации подзадач. В разделе 4 представлены результаты вычислительных экспериментов по применению эвристик, связанных с кластеризацией подзадач. А в заключении приводятся некоторые направления для дальнейшей работы по данной тематике.