В показанном примере используются два порта в общее адресное пространство. Для обеспечения бесконфликтной работы памяти достаточно применить двухпортовую память или регистровый файл. В случае, когда доступна только однопортовая память, акселератор будет проводить половину времени работы в ожидании данных от памяти. В более сложных случаях для обеспечения высокой производительности алгоритм должен быть переработан для использования нескольких независимых банков памяти.

В показанном примере нет межитерационных зависимостей, поэтому он может быть легко распараллелен путем дублирования аппаратуры. Единственная обратная связь проходит через функциональный блок (+1) в счетчике итераций. Если задержка через этот функциональный блок составляет 5 нс, то акселератор сможет работать на частоте до 200 МГц при соответствующей конвейеризации остальной части тракта данных. Чтобы поднять частоту акселератора, потребуется разбить операцию на обратной связи на несколько стадий, однако это не приведет к увеличению производительности, так как счетчик итерации будет производить действительные данные только 1 раз за N тактов, где N – число конвейерных стадий на обратной связи.

Заключение

В работе представлены основные типы потоковых вычислительных моделей, пригодных для аппаратной реализации. Показан пример создания акселератора на основе расширенной BDF модели, являющейся одной из наиболее простых и эффективных при прямой реализации в аппаратуре. Выделены основные факторы, влияющие на производительность потоковых машин.

Потоковые модели обладают значительной выразительной мощью, что позволяет с их применением описывать широкий класс алгоритмов из различных предметных областей. При этом полученные модели пригодны как для прямой аппаратной реализации, так и для отображения на различные реконфигурируемые машины [5]. Наиболее перспективной областью применения потоковых моделей является их использование в виде промежуточного представления в системах высокоуровневого синтеза. С их помощью можно выразить все вычислительные и управляющие конструкции, используемые в современных языках высокого уровня, при этом они позволяют максимально использовать возможности распараллеливания вычислений в специализированной аппаратуре.

Литература

1. Budiu M., Artigas P.V., Goldstein S.C. Dataflow: A Complement to Superscalar // Proc. of the IEEE International Symposium on Performance Analysis of Systems and Software. – Washington, DC, USA, 2005. – P. 177–186.
2. Kahn G. The semantics of a simple language for parallel programming // Proc. of IFIP Congress. – Stockholm, Sweden, 1974. – P. 471–475.
3. Edward A. Lee, David G. Messerschmitt. Synchronous Data Flow // Proc. of the IEEE. – 1997. – V. 75. – № 9. – P. 1235–1245.
4. Buck J.T. A dynamic dataflow model suitable for efficient mixed hardware and software implementations of DSP applications // Proc. of the Third International Workshop on Hardware/Software Codesign. – France, 1994. – P. 165–172.
5. Swanson S., Michelson, K. WaveScalar // Proc. of the 36th annual IEEE/ACM International Symposium on Microarchitecture. – USA, 2003. – P. 291–302.

Попов Роман Игоревич – Санкт-Петербургский государственный университет информационных технологий, механики и оптики, аспирант, ripopov@gmail.com

УДК 004.021

МЕТОД ИСПРАВЛЕНИЯ ОШИБОК В НАБОРЕ ЧТЕНИЙ НУКЛЕОТИДНОЙ ПОСЛЕДОВАТЕЛЬНОСТИ

А.В. Александров, С.В. Казаков, С.В. Мельников, А.А. Сергушичев, Ф.Н. Царев, А.А. Шалыто

Задача секвенирования и сборки больших геномов является одной из актуальных задач современной биоинформатики. Исходными данными для сборки генома являются так называемые чтения, получаемые с помощью машин-секвенаторов. Эти чтения могут содержать ошибки, вызванные тем, что секвенирование основано на выполнении ряда химических реакций. Часто одним из этапов сборки генома является исправление ошибок в наборе чтений. Большинство методов исправления ошибок в чтениях строят граф де Брюина. Предлагаемый метод не использует граф де Брюина, что позволяет существенно сократить использование памяти. Метод реализован и опробован при сборке генома в рамках проекта «de novo Genome Assembly Project».

Ключевые слова: геном, секвенирование, сборка генома, исправление ошибок.

Введение

Многие современные задачи биологии и медицины требуют знания генома живых организмов, который состоит из нескольких нуклеотидных последовательностей молекул дезоксирибонуклеиновой ки-

слоты (ДНК). В связи с этим возникает необходимость в дешевом и быстром методе секвенирования, т.е. определения последовательности нуклеотидов в образце ДНК.

Существующие технические средства (секвенаторы) не позволяют считать разом всю молекулу ДНК. Вместо этого они позволяют читать фрагменты генома небольшой длины. Длина фрагмента может варьироваться и является важным параметром секвенирования, так как от нее напрямую зависит стоимость секвенирования и время, затрачиваемое на чтение одного фрагмента: чем больше длина считываемого фрагмента, тем выше стоимость чтения и тем дольше это чтение происходит. В связи с этим сейчас получил распространение следующий дешевый и эффективный подход: сначала вычленяется случайно расположенный в геноме фрагмент длиной около 500 нуклеотидов, а затем считываются его префикс и суффикс (длиной около 80–120 нуклеотидов каждый). Эти префикс и суффикс называются *парными чтениями*. Этот процесс повторяется такое число раз, чтобы обеспечить достаточно большое покрытие генома чтениями.

Отметим, что описанные выше префикс и суффикс читаются с разных нитей ДНК: один – с прямой, другой – с обратно-комплементарной, причем неизвестно, который откуда. По этой причине удобно рассматривать геном и чтения, дополненные своими обратно-комплементарными копиями.

Описанный выше технологический процесс реализуется, например, секвенаторами компании Illumina [1]. При этом важной особенностью работы этих секвенаторов является возможность совершения ошибок при чтении. Это означает, что некоторые нуклеотиды секвенируются неверно (например, вместо нуклеотида A читается нуклеотид G). Ошибки замены – не единственный тип ошибок, допускаемый секвенаторами. Так, возможны также ошибки вставки и удаления. Однако эти ошибки встречаются в довольно специфических случаях и по сравнению с ошибками замены очень редки.

Кроме самой последовательности нуклеотидов, результатом работы секвенатора является информация о качестве прочтения каждого из нуклеотидов. По ней может быть вычислена вероятность того, что данный нуклеотид был прочитан неверно.

Для эффективной работы последующих стадий алгоритма очень важно исправить как можно больше ошибок в чтениях.

Существует большое число сборщиков, осуществляющих все или только некоторые из приведенных выше этапов. Все их можно разделить на две группы. Одни используют для исправления ошибок так называемый граф де Брюина [2], например ABySS [3], Velvet [4] или SOAPdenovo [5]. Другие проводят частотный анализ строк длины k (их обычно называют k-мерами) – для каждого k-мера считают, сколько раз он встретился в чтениях, и на основе этих данных исправляют ошибки. Так работают, например, ALLPATHS [6] и EULER [7]. Все эти методы довольно требовательны к вычислительным ресурсам и не очень хорошо масштабируются.

Целью настоящей работы является разработка метода, лишенного указанных недостатков, который будет использоваться в качестве первого этапа сборки генома из набора парных чтений.

Идея метода

Для эффективного исправления ошибок необходимо, чтобы каждая позиция генома была прочитана несколько раз, что, ввиду небольшой вероятности ошибки, дает право считать, что наибольшее число раз нуклеотид на каждой позиции был прочитан верно. На практике используются наборы чтений, покрывающие геном несколько десятков раз. Важно отметить, что не только отдельные позиции всего генома были прочитаны несколько десятков раз, но и небольшие его подстроки (не длиннее самих чтений) встречаются в чтениях несколько раз, причем чем длиннее подстрока, тем меньше шансов, что несколько различных чтений ее содержат. Последнее соображение вытекает не только из увеличения вероятности попадания чтения на конкретную подстроку при увеличении ее длины, но и из факта наличия в чтениях ошибок.

В работе рассматриваются строки одинаковой длины k (так называемые k-меры). Для каждого k-мера, присутствующего в чтениях (прямых или обратно-комплементарных) хотя бы раз, вычислим, сколько раз он встречается в чтениях. На основании этой статистики все k-меры можно разделить на 2 группы – «надежные» k-меры и «подозрительные». «Надежные» k-меры – это те, которые встречаются в чтениях достаточно большое число раз – не меньше значения некоторого порога t, которое является параметром алгоритма и выбирается вручную. С «надежными» k-мерами делать ничего не нужно, предполагается, что в них ошибок нет. «Подозрительные» же k-меры вызваны или плохим покрытием тех частей генома, откуда они были прочитаны, либо, что гораздо более вероятно, наличием в них одной или нескольких ошибок, которые надлежит исправить.

После выделения «подозрительных» k-меров для каждого из них необходимо определить, в какой именно позиции могла быть совершена ошибка. Для этого предлагается перебрать все позиции k-мера (их ровно k) и все возможные нуклеотиды, попробовать заменить имеющийся нуклеотид на перебираемый и проанализировать получившийся k-мер. Если новый k-мер попадает в группу «надежных», значит, возможно, рассматриваемый k-мер является результатом его ошибочного прочтения. Если в течение пе-

ребора был найден только один «надежный» *k*-мер, получающийся из «подозрительного» путем замены одного нуклеотида на другой, полагается, что данный «подозрительный» *k*-мер исправлен, а соответствующее исправление запоминается. Если таких *k*-меров несколько, неясно, какое из исправлений запоминать, поэтому в таких случаях «подозрительные» *k*-меры не исправляются. И, наконец, если не было найдено ни одного способа, исправить «подозрительный» *k*-мер, полагается, что в нем было совершено больше одной ошибки, после чего запускается аналогичная процедура, но с исправлением не одного, а сразу двух нуклеотидов. Аналогичным образом можно пытаться изменить не только пары, но и тройки, а также кортежи из большего числа нуклеотидов, однако данное обобщение ощутимо сказывается на быстродействии алгоритма.

При выборе значения параметра *k* следует учитывать следующие соображения.

- Величина *k* должна быть значительно меньше длины чтений. Если длина *k*-мера будет сравнима с длиной чтения, то большинство *k*-меров будет встречаться в чтениях один раз, что не даст никакой информации для исправления ошибок.
- Величина *k* должна быть достаточно большой, чтобы вероятность того, что случайный *k*-мер заданной длины встречается в геноме, была ничтожно малой. В противном случае некоторые *k*-меры, содержащие ошибку, не будут исправлены только потому, что есть достаточно большая вероятность того, что эти *k*-меры на самом деле встречаются в геноме, а тогда они могут много раз встречаться в чтениях.

Алгоритм исправления ошибок в чтениях

Для начала отметим, что качество нуклеотидов сильно ухудшается у конца чтения, поэтому для данного алгоритма имеет смысл обрезать куски чтений плохого качества, так как, с одной стороны, благодаря им образуется много «подозрительных» *k*-меров, с другой стороны, их невозможно исправить из-за большого числа ошибок.

Также важно заметить, что все *k*-меры можно разбить на группы на основании префикса небольшой длины, с которого они начинаются. Если при этом не исправлять ошибки в префиксе, то исправление не выводит *k*-меры из группы. Это не повлечет за собой неисправленные ошибки в префиксах, так как при обращении *k*-мера его префикс становится суффиксом.

Таким образом, алгоритм состоит из следующих шагов:

- обрезание *k*-меров на основании качества;
- разбиение *k*-меров на группы на основании префиксов;
- выполнение для каждой группы:
  - сбор статистики по *k*-мерам;
  - исправление ошибок.

Важно отметить, что алгоритм поиска ошибок в *k*-мерах легко распараллеливается, так как для обработки одного *k*-мера ему требуется только доступ на чтение к общей структуре данных, хранящей статистику по содержанию *k*-меров в чтениях, а также кратковременный доступ на запись для сохранения результата.

Экспериментальные результаты

Описанный подход был разработан и применен в рамках проекта dnGASP [8]. В этом проекте участникам предлагалось восстановить синтетический геном, содержащий около 1,8 миллиардов нуклеотидов, который был покрыт чтениями 44 раза. Этот геном являлся смесью геномов различных живых организмов. При реализации алгоритма для хранения *k*-меров использовалась хэш-таблица с открытой адресацией [9]. Это позволило осуществлять добавление нового *k*-мера и проверку «надежности» нового за константное время (последнее особенно важно для распараллеливания алгоритма). Ввиду того, что в рамках этого конкурса чтения имели длину 114, было выбрано значение *k*, равное 30.

По частотному распределению *k*-меров (рисунок) было выбрано значение порога *t*, равное четырем. Алгоритм исправления ошибок работал на компьютере с четырьмя шетиядерными процессорами Intel® Xeon® E7450 с тактовой частотой 2,4 ГГц и с 24 ГБ оперативной памяти. До запуска алгоритма в исходных данных было 6,5 миллиардов различных *k*-меров, из которых 3 миллиарда «надежных». Алгоритм работал около 24 часов, после чего в данных стало 3,9 миллиардов (что на 40% меньше, чем в начале) различных *k*-меров, из которых 3,3 миллиарда «надежных».

Заключение

Разработан метод исправления ошибок, основанный на частотном анализе *k*-меров. Проведено экспериментальное исследование метода, показавшее его работоспособность на данных, близких к реальным. Разработанный метод может эффективно использовать как достаточно малый, так и большой объем ресурсов. В настоящий момент исследуются возможные изменения, которые можно осуществить для улучшения эффективности и быстродействия предложенного метода. Например, планируется в

большей степени учитывать информацию о качестве прочитанных нуклеотидов, предоставляемую секвенаторами. Кроме того, рассматривается возможность учета взаимосвязи *k*-меров в чтениях.

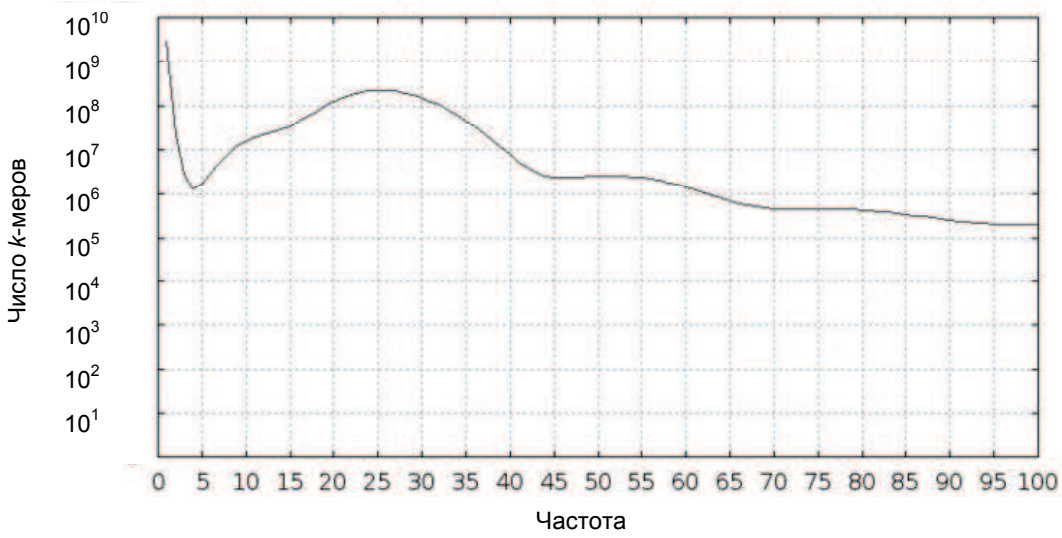

Рисунок. Зависимость числа *k*-меров от того, сколько раз они встречаются в чтениях

Литература

1. Illumina, Inc. [Электронный ресурс]. – Режим доступа: http://www.illumina.com/, свободный. Яз. англ. (дата обращения 28.05.2011).
2. Pevzner P. A 1-Tuple DNA sequencing: computer analysis // J. Biomol. Struct. Dyn. – 1989. – V. 7. – P. 63–73.
3. Simpson J.T., Wong K., Jackman S.D., Schein J.E., Jones S.J., Birol I. ABySS: A parallel assembler for short read sequence data // Genome Res. – 2009. – V. 19 – P. 1117–1123.
4. Zerbino D.R., Birney E. Velvet: Algorithms for de novo short read assembly using de Bruijn graphs // Genome Res. – 2008. – V. 18 – P. 821–829.
5. Li R., Zhu H., Ruan J., Qian W., Fang X., Shi Z., Li Y., Li S., Shan G., Kristiansen K. et al. De novo assembly of human genomes with massively parallel short read sequencing // Genome Res. – 2010. – V. 20. – P. 265–272.
6. Butler J., MacCallum I., Kleber M., Shlyakhter I.A., Belmonte M.K., Lander E.S., Nusbaum C., Jaffe D.B. AllPaths: De novo assembly of wholegenome shotgun microreads // Genome Res. – 2008. – V. 18. – P. 810–820.
7. Pevzner P.A., Tang H., Waterman M.S. EULER: An Eulerian path approach to DNA fragment assembly // Proc. Natl. Acad. Sci. – 2001. – V. 98. – P. 9748–9753.
8. de novo Genome Assembly Project (dnGASP) [Электронный ресурс]. – Режим доступа: http://cnag.bsc.es/, свободный. Яз. англ. (дата обращения 28.05.2011).
9. Кормен Т., Лейзерсон Ч., Ривест Р., Штайн К. Алгоритмы: построение и анализ: Пер. с англ. – 2-е изд. – М.: Вильямс, 2007. – 1296 с.

Александров Антон Вячеславович – Санкт-Петербургский государственный университет информационных технологий, механики и оптики, студент, alexandrov@rain.ifmo.ru

Казаков Сергей Владимирович – Санкт-Петербургский государственный университет информационных технологий, механики и оптики, студент, svkazakov@rain.ifmo.ru

Мельников Сергей Вячеславович – Санкт-Петербургский государственный университет информационных технологий, механики и оптики, студент, melnikov@rain.ifmo.ru

Сергушичев Алексей Александрович – Санкт-Петербургский государственный университет информационных технологий, механики и оптики, студент, alserg@rain.ifmo.ru

Царев Федор Николаевич – Санкт-Петербургский государственный университет информационных технологий, механики и оптики, аспирант, fedor.tsarev@gmail.com

Шалыто Анатолий Абрамович – Санкт-Петербургский государственный университет информационных технологий, механики и оптики, доктор технических наук, профессор, зав. кафедрой, shalyto@mail.ifmo.ru