Опубликовано в материалах 2-й межвузовской научной конференции по проблемам информатики СПИСОК-2011, с. 326-329.

А. В. Александров, С. В. Казаков,
С. В. Мельников, Ф. Н. Царев,
А. А. Шалыто
Санкт-Петербургский государственный университет информационных технологий, механики и оптики
Е. Б. Прохорчук
Центр «Биоинженерия» РАН

Разработка метода удаления ошибок из набора чтений нуклеотидной последовательности

Многие современные задачи биологии и медицины требуют знания генома живых организмов, который состоит из нескольких нуклеотидных последовательностей ДНК. В связи с этим возникает необходимость в дешевом и быстром методе секвенирования, то есть определения последовательности нуклеотидов в образце ДНК.

Существующие технические средства (*секвенаторы*) не позволяют считать разом всю молекулу ДНК организма. 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 происхдит. В связи с этим сейчас получил распространениес ледующий дешевый и эффективный подход: сначала вычленяется случайно расположенный в геноме фрагмент длиной около 500 нуклеотидов, а затем считываются его префикс и

суффикс (длиной по 114 каждый). Эти префикс и суффикс называются *парными чтениями*. Этот процесс повторяется такое число раз, чтобы обеспечить достаточно большое покрытие генома чтениями.

Описанный выше технологический процесс реализуется, например, секвенаторами компании *Illumina* [1]. При этом важной особенностью работы этих секвенаторов является допуск ошибок. 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 спецефических случаях и по сравнению с ошибками замены очень редки.

Помимо самой последовательности нуклеотидов результатом работы секвенатора является также последовательность из величин качества для каждого нуклеотида. В ней содержится информация о качестве чтения каждого нуклеотида, по которой 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.

После исправления ошибок запускается алгоритм восстановления фрагментов нуклеотидной последовательности, результатом работы которого являются целые фрагменты, а не только их префиксы и суффиксы. Эти фрагменты называются *квазиконтиги* и используются для построения *контигов* — максимальных непрерывных последовательностей нуклеотидов, которые удалось восстановить. Затем контиги используются для построения *скэффолдов* — последовательностей контигов,

разделенных промежутками с длинами, для которых известны оценки. Таким образом, результатом работы сборщиков является набор последовательностей нуклеотидов, разделенных промежутками более-менее известной длины.

Существует большое число сборщиков [2-6], осуществляющих все или только некоторые из приведенных выше этапов. В данной работе описан метод, осуществляющий исправление ошибок и использующийся в качестве первого этапа сборки генома из набора чтений.

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 соображений вероятности попадания чтения на конкретную подстроку, но и из факта наличия в чтениях ошибок.

Подстроки, упомянутые выше, обычно называются *k*-мерами. К выбору величины *k* следует подходить достаточно серьезно, так как этот параметр сильно влияет на работу алгоритма. При выборе значения этого параметра следует учитывать следующие простые, но важные 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 чтениях, была ничтожно маленькой. В противном случае некоторые *k*-меры, содержащие ошибку, не будут исправлениы только потому, что в чтениях присутствует «похожий» на них *k*-мер, прочитанный из другого места генома.

Первая часть работы алгоритма состоит в собирании информации о *k*-мерах. Для каждого *k*-мера, присутствующего в чтениях хотя бы раз, подсчитывается, 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 (точное значение данного порога — еще один, но не такой существенный, как значение *k*, параметр алгоритма). С «надежными» *k*-мерами делать ничего не нужно, предполагается, что в них ошибок нет. «Подозрительные» же *k*-меры полагаются содержащими одну или, что менее вероятно, несколько ошибок, которые надлежит исправить.

После выделения «подозрительных» *k*-меров для каждого из них необходимо решить, в какой именно позиции была совершена ошибка. Для этого предлагается перебрать все позиции *k*-мера (их ровно *k* штук) 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 ошибочного прочтения. Если в течение перебора 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.

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.

Описанный подход был разработан и применен в рамках проекта *dnGASP* [7]. В этом проекте участникам предлагалось восстановить синтетический геном, содержащий около 1,8 миллиардов нуклеотидов. При реализации описанного алгоритма для хранения *k*-меров использовался хэш-мэп с открытой адресацией. 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. Эмпирически было установлено, что «надежные» *k*-меры — те, которые встречаются в чтениях не менее 4 раз.

Алгоритм исправления ошибок работал на 24-ядерном компьютере с 64 ГБ оперативной памяти. До запуска алгоритма в исходных данных было 6,5 миллиардов различных *k*-меров, из которых 3 миллиарда «надежных». Алгоритм работал около суток, после чего в данных стало 3,9 миллиардов (что на 40% меньше, чем в начале) различных *k*-меров, из которых 3,3 миллиарда «надежных».

В настоящий момент исследуются возможные изменения, которые можно осуществить для улучшения эффективности и быстродействия предложенного метода.

Литература

1. Illumina, Inc. http://www.illumina.com/
2. *Zerbino D. R., Birney E.* Velvet: Algorithms for de novo short read assembly using de Bruijn graphs. Genome Research 18. 2008. P. 821–829.
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 Research 19. 2009. P. 1117–1123.
   http://genome.cshlp.org/content/19/6/1117.full.pdf+html
4. *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 Research 20. 2010. P. 265–272.
5. *Pevzner P. A., Tang H., Waterman M. S.* An Eulerian path approach to DNA fragment assembly. Proceedings of the National Academy of Sciences 98. 2001. P. 9748–9753
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 whole-genome shotgun microreads. Genome Research 18. 2008. P. 810-820.

7. De novo Genome Assembly Project (dnGASP). http://cnag.bsc.es/