УДК 577.218

Использование графических ускорителей для выявления функциональных сигналов в регуляторных районах дифференциально экспрессирующихся генов AGRP нейронов гипоталамуса мыши в ответ на голодание

А. В. Бочарников, Е. В. Игнатьева, О. В. Вишневский[1]

Выявление *de novo* контекстных сигналов в регуляторных районах генов эукариот существенно затрудняется как огромными объемами анализируемых выборок последовательностей, так и гигантским разнообразием контекстных сигналов. Нами предложен новый алгоритм оценки представленности вырожденных олигонуклеотидных мотивов, записанных в 15-буквенном IUPAC коде, в выборке нуклеотидных последовательностей и показана его высокая производительность по сравнению с ранее предложенным подходом. Данный метод основан, во-первых, на использовании деревьев префиксов, во-вторых, на соответствии префиксов мотивов диапазонам хешей в хешированных нуклеотидных последовательностях анализируемой выборки и, в-третьих, на технологии CUDA, позволяющей использовать для массового параллельного счета графические ускорители, широкодоступные для исследователей.

Предложенный подход был использован для проведения контекстного анализа промоторных областей генов мыши с достоверно изменившейся после лишения пищи экспрессией в AGRP (Agouti Related Peptide) нейронах гипоталамуса. Когда животное лишено пищи, так называемые AGRP нейроны гипоталамуса вырабатывают молекулы, которые повышают аппетит и облегчают набор веса. Понимание клеточных механизмов, лежащих в основе функционирования нейронов AGRP в ответ на потерю веса, необходимо для разработки методов борьбы с ожирением, которое является наследственным заболеванием, имеющим лишь несколько безопасных и долгосрочных эффективных методов лечения и стратегий вмешательства. Проведенный нами анализ выявил значимые олигонуклеотидные мотивы, ассоциированные с голоданием.

Ключевые слова: олигонуклеотидный мотив, GPGPU, ожирение.

1. Введение

Выявление функциональных сигналов в регуляторных районах генов является важной задачей современной биоинформатики и необходимо для понимания базовых механизмов регуляции транскрипции и трансляции.

Выявление олигонуклеотидных мотивов *de novo* является одним из наиболее ранних и широко используемых биоинформатических подходов к обнаружению контекстных сигналов в регуляторных районах генов. Такие мотивы могут соответствовать как сайтам связывания транскрипционных факторов, так и определенным физико-химическим особенностям регуляторных районов. В то же время их использование затруднено гигантским разнообразием возможных вариантов. Так, мотив длиной в 8 нуклеотидов, записанный в 15-буквенном IUPAC

[1] Благодарности: Работа поддержана бюджетным проектом № 0324-2019-0040.

коде (табл. 1), составляет $15^8 \sim 2.5 \times 10^9$ различных вариантов записи в 4-буквенном коде. Это заставляет исследователей использовать различные эвристические подходы, основанные на анализе частот *l*-плетов [1], деревьев суффиксов [2], локальном множественном выравнивании [3], EM методе (Expectation–Maximization) [4, 5] и т.д. В то же время такие подходы не гарантируют нахождения наиболее достоверного мотива. Решением может являться использование полнопереборного строкового подхода, что требует применения высокопроизводительных массивно-параллельных вычислительных систем, таких как GPU-ускорители [6].

Ранее нами был предложен такой полнопереборный подход, реализованный в виде программной системы Argo_CUDA [7] как на CPU, так и на GPU-устройствах и продемонстрирована его высокая эффективность в сравнении с существующими подходами. В данной работе мы предлагаем новый подход, основанный на использовании дерева префиксов мотивов и позволяющий существенно увеличить производительность.

Предложенный подход был использован для анализа промоторов генов, экспрессия которых достоверно отличалась в AGRP нейронах гипоталамуса голодающих и сытых мышей. В промоторах генов, повышающих экспрессию в ответ на голод и понижающих ее, были найдены достоверные олигонуклеотидные мотивы. Анализ полученных мотивов выявил достоверное сходство с рядом известных сайтов связывания транскрипционных факторов. Классификация генов, промоторы которых содержали сигналы, найденные с использованием системы DAVID [8, 9], выявила их достоверную ассоциацию с голоданием.

2. Материалы и методы

2.1. Описание алгоритма поиска мотивов, основанного на дереве префиксов мотивов

Для полнопереборной оценки представленности каждого из 15^k вырожденных мотивов длины k в выборке из N нуклеотидных последовательностей длины L необходимо сравнить каждый мотив длины k с каждой k-символьной подстрокой этой выборки, что потребует $N \times (L - k + 1)$ сравнений. Для ускорения этого процесса удобно записывать и нуклеотидные последовательности, и мотивы в виде бинарных хэшей (табл. 1).

Таблица 1. Бинарное представление вырожденных олигонуклеотидов записанных в 15-буквенном IUPAC коде

IUPAC код	Нукл.	Хэш
A	A	0001
T	T	0010
W	A/T	0011
G	G	0100
R	A/G	0101

IUPAC код	Нукл.	Хэш
K	T/G	0110
D	A/T/G	0111
C	C	1000
M	A/C	1001
Y	T/C	1010

IUPAC код	Нукл.	Хэш
H	A/T/C	1011
S	G/C	1100
V	A/G/C	1101
B	T/G/C	1110
N	A/T/G/C	1111

В этом случае для записи одного символа требуется 4 бита, а сравнение мотива с 8-нуклеотидной подстрокой возможно с использованием одной операции AND (табл. 2).

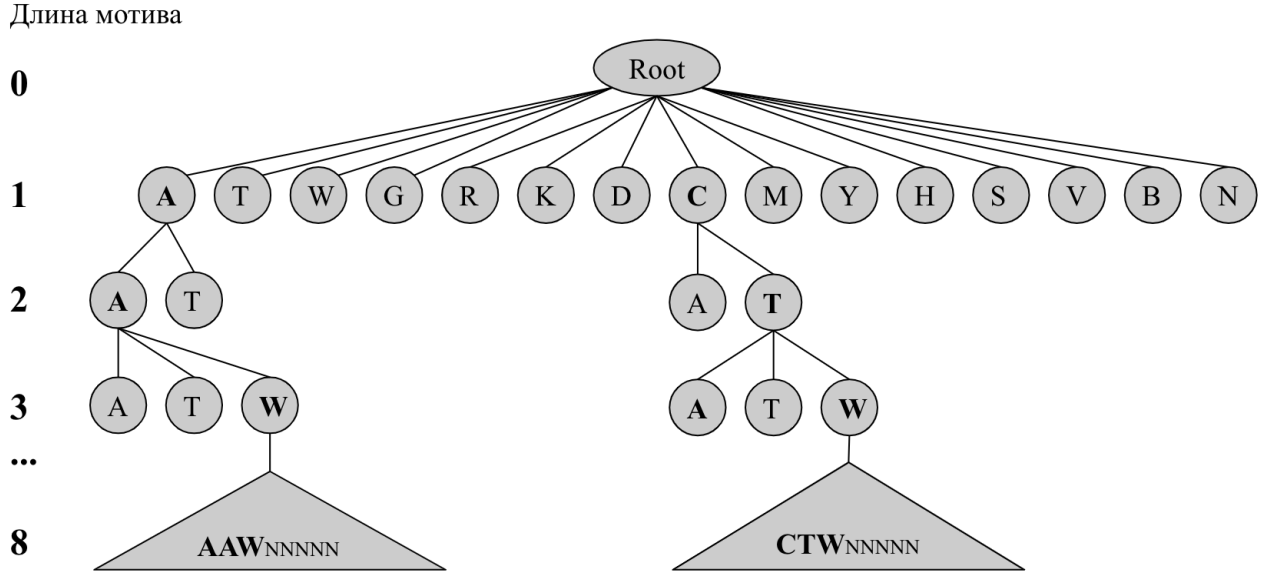
Таблица 2. Сравнение мотива WWWWAAAA и олигонуклеотида ATATAAAA

```
          boolean match = (hash & motif_hash) == hash

motif_hash   WWWWAAAA   0011 0011 0011 0011 0001 0001 0001 0001
hash         ATATAAAA   0001 0010 0001 0010 0001 0001 0001 0001
match        true       0001 0010 0001 0010 0001 0001 0001 0001
```

Если хотя бы в одной позиции мотив не совпадает с олигонуклеотидом, то результат побитового AND будет равен нулю. Таким образом, сложность сравнения мотива с подстрокой снижается с 8 операций посимвольного сравнения при стандартной записи до одной операции побитового AND при записи в виде хэша.

Полный перебор всех 15^k вариантов мотивов можно представить в виде обхода дерева (рис. 1). Корневому узлу соответствует пустая строка. У каждого узла, кроме листовых, имеется 15 потомков, в каждом из которых хранится символ 15-буквенного IUPAC алфавита. В каждом узле текущий префикс мотива определяется набором символов из всех узлов на пути от корня к текущему узлу. Для каждой вершины поддерево со всеми узлами-потомками содержит все возможные варианты мотивов с заданным префиксом. В листовых вершинах префиксом является полный мотив.



Поддерево мотивов с префиксом **AAT** Поддерево мотивов с префиксом **CTW**

Рис. 1. Представление множества мотивов в виде дерева степени 15 глубины 8

Мотив совпадает с подстрокой входной выборки только в том случае, если их префиксы совпадают. Таким образом, каждому возможному префиксу мотива соответствует множество подстрок входной выборки нуклеотидных последовательностей (рис. 2).

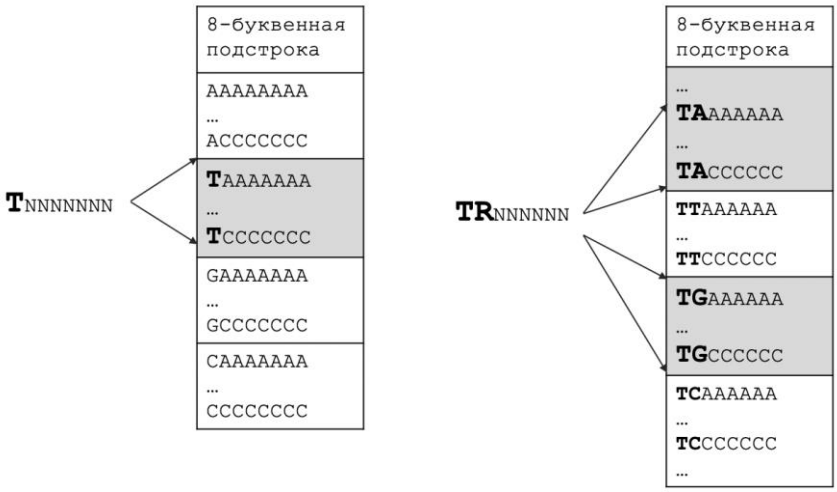
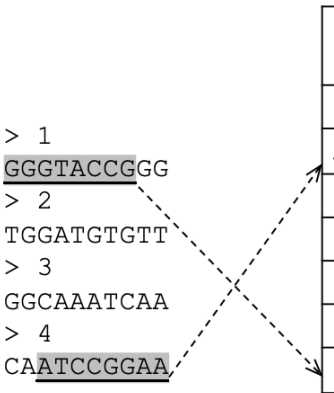


Рис. 2. Пример сопоставления вырожденных мотивов (с префиксами T и TR) и олигонуклеотидных диапазонов во входной выборке последовательностей с сохранением номера последовательности, в которой эти мотивы предоставлены

Для сокращения числа сравнений каждого мотива со всеми подстроками входной выборки был предложен метод построения диапазонов подстрок по префиксу мотива длины l_p, состоящий из трех шагов.

Шаг 1. Представление входной выборки в виде набора отсортированных хэшей.

Входные последовательности представляются в виде множества всех входящих в них k-буквенных подстрок. После чего подстроки преобразуются в единый список хэшей и сортируются в лексикографическом порядке (рис 3).



8-буквенная подстрока	Номер последовательности	Бинарное представление (хэш)
AATCCGGA	4	0001 0001 0010 1000 1000 0100 0100 0001
ATCCGGAA	4	0001 0010 1000 1000 0100 0100 0001 0001
TGGATGTG	2	0010 0100 0100 0001 0010 0100 0010 0100
GATGTGTT	2	0100 0001 0010 0100 0010 0100 0010 0010
GTACCGGG	1	0100 0010 0001 1000 1000 0100 0100 0100
GGATGTGT	2	0100 0100 0001 0010 0100 0010 0100 0010
GGTACCGG	1	0100 0100 0010 0001 1000 1000 0100 0100
GGGTACCG	1	0100 0100 0100 0010 0001 1000 1000 0100
GGCAAATC	3	0100 0100 1000 0001 0001 0001 0010 1000
GCAAATCA	3	0100 1000 0001 0001 0001 0010 1000 0001
CAAATCAA	3	1000 0001 0001 0001 0010 1000 0001 0001
CAATCCGG	4	1000 0001 0001 0010 1000 1000 0100 0100

Рис. 3. Пример построения отсортированного в лексикографическом порядке списка хэшей всех 8-буквенных подстрок по входной выборке нуклеотидных последовательностей длины 10

Шаг 2. Построение диапазонов для нуклеотидных префиксов (в 4-букв. коде).

На втором шаге перебираются все возможные олигонуклеотидные префиксы заданной длины (строка 3). Для каждого префикса с помощью бинарного поиска определяется диапазон во входной выборке, где он представлен (строки 5–6).

```
1 total_prefixes = pow(4, prefix_size)
2 Ranges ranges_by_olig_prefix
3 for (i = 0; i < total_prefixes; i++)
4    olig_prefix_hash = olig_idx_to_hash(i)
5    low_bound= lower_bound(sequences, olig_prefix_hash)
6    up_bound = upper_bound(sequences, olig_prefix_hash | 0xFFFFFFFF)
7    if (low_bound != up_bound)
8        ranges_by_olig_prefix[olig_prefix_hash].add({ low_bound, up_bound })
```

Здесь *prefix_size* – длина префикса мотива l_p, *olig_idx_to_hash* – функция, которая по индексу олигонуклеотида возвращает его представление в виде хэша (табл. 2), *lower_bound* – позиция первого элемента, который не меньше заданного, *upper_bound* – позиция первого элемента, который больше заданного, *sequences* – отсортированный список хэшей всех подстрок длины *k* входной выборки.

Шаг 3. Построение диапазонов для префиксов мотивов (в 15-букв. IUPAC коде).

На третьем шаге перебираются все возможные варианты префиксов мотивов заданной длины (строка 3). Для каждого из префиксов мотива рассматриваются все варианты его записи в виде олигонуклеотида (строка 5) и для них объединяются диапазоны, посчитанные в шаге 2 (строка 8).

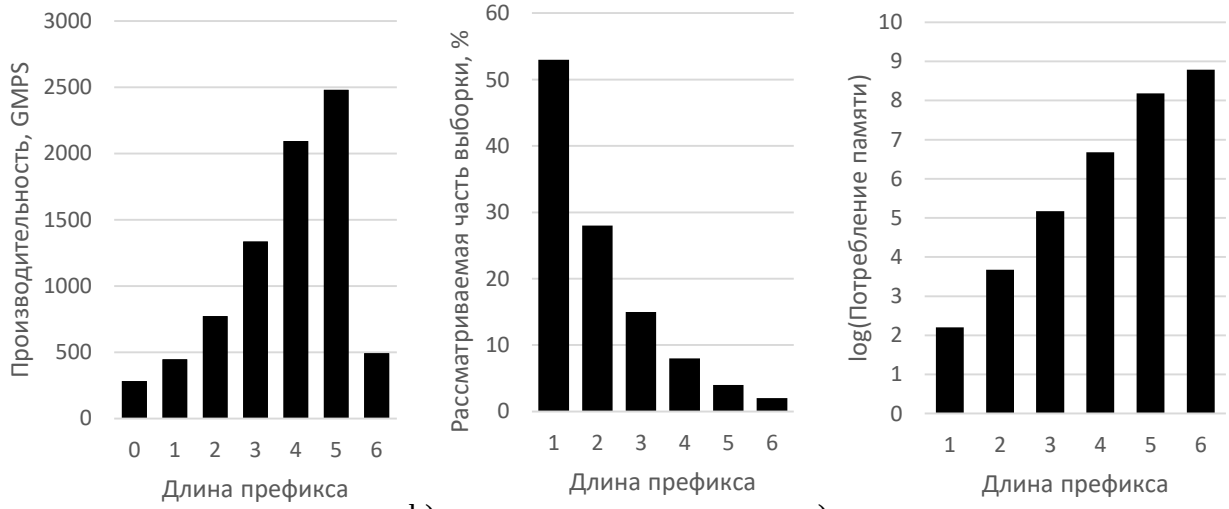
```
1 total_prefixes = pow(16, prefix_size)
2 Ranges ranges_by_motif_prefix
3 for (i = 1; i < total_prefixes; i++)
4     motif_prefix_hash = motif_idx_to_hash(i, prefix_size)
5     for (j = 0; j < motif_variants(i); j++)
6         olig_hash_mask = motif_variant(motif_prefix_hash)
7         range = ranges_by_olig_prefix[olig_hash_mask]
8         ranges_by_motif_prefix[motif_prefix_hash].add(range)
```

Здесь *motif_idx_to_hash* – функция, которая по индексу мотива возвращает его представление в виде хэша.

После того, как получены диапазоны для всех возможных префиксов мотивов, они могут быть использованы для оценки представленности мотивов в выборке:

```
1 for (i = 0; i < pow(15,8); i++)
2     motif_hash = motif_idx_to_hash(i)
3     motif_prefix = motif_hash >> (32 - 4*prefix_size)
4     motif_ranges = ranges.get(motif_prefix)
5     for (j = 0; j < ranges.size; j++)
6         for (k = motif_ranges[j].start; k < motif_ranges[j].end; k++)
7             match = (sequence_hashes[k] & motif_hash) == sequence_hashes[k])
```

Таким образом, число сравнений мотива с подстроками входной выборки существенно снижается. Проведенный нами анализ показал, что количество перебираемых подстрок уменьшается (рис. 4б), а производительность (рис. 4а) и размер оперативной памяти (рис. 4с) для хранения диапазонов растут в зависимости от длины префикса. Оптимальным размером префикса является 5 или 6 в зависимости от размеров входной выборки.



a) b) c)

Рис. 4. Оценка (a) производительности GMPS (Giga matches per second, ось Y), (b) рассматриваемый процент подпоследовательностей в выборке и (c) размер оперативной памяти для хранения диапазонов в зависимости от длины префикса (ось X) для выборки из 1000 последовательностей длины 128

2.2. Реализация алгоритма поиска мотивов на GPU

Современные GPU-ускорители содержат тысячи вычислительных ядер, позволяющих одновременно обсчитывать десятки тысяч потоков. При этом наиболее эффективно с помощью GPU могут быть распараллелены задачи, где отсутствуют зависимости по данным. К таким задачам может быть отнесен и расчет представленности мотива в выборке нуклеотидных последовательностей.

```
1  motif_index = blockIdx.x * blockDim.x + threadIdx.x
2  motif_hash = idx_to_hash(motif_index)
3  extern __shared__ unsigned char matched_sequences[]
4  motif_prefix = motif_hash >> (8 - prefix_size)*4
5  Range range = ranges[motif_prefix]
6  offset = threadIdx.x * sequences_count

8  for (i = range.start_pos; i < range.end_pos; i++)
9      hash = sequence_hashes[i]
10     match = (hash & motif_hash) == hash
11     if (match)
12         seq_id = hash_to_sequence_id[i]
13         matched_sequences[offset + seq_id] = 1
14
15 for (i = offset; i < (offset + sequences_count); i++)
16     if (matched_sequences[i]) presense += 1
17         motif_presense[motif_index] = presense
```

При расчетах на GPU каждый поток в блоке формирует очередной индекс мотива (строка 2) и с помощью функции *idx_to_hash* преобразует его в хэш. Префикс вычисляется с помощью битового сдвига хэша мотива на *(8-prefix_size)*4* бит вправо (строка 4). Полученный префикс используется для получения соответствующего ему диапазона подстрок входной выборки (строка 5). По индексу каждой подстроки, совпадающей с мотивом (строка 10), в массив результатов *matched_sequence* записывается единица (строка 13). Каждому потоку из блока выделяется свой участок памяти для результатов размером *sequence_count* (строки 6, 13, 15). Встречаемость мотива во входной выборке оценивается количеством единиц в массиве результатов (строки 16–17). Для ускорения сохранения результатов сравнений массив *matched_sequences* расположен в разделяемой памяти GPU (строка 3). Размер S разделяемой памяти ограничен (48–96 Кб на блок), поэтому максимальное число последовательностей, обрабатываемых за 1 запуск вычислительного ядра, составляет $S \,/\, THREAD\_PER\_BLOCK$

3. Результаты и обсуждение

3.1. Реализация алгоритма поиска мотивов на GPU

Для того чтобы получить оценки производительности работы программы, не зависящие ни от длины и количества анализируемых последовательностей, ни от длины мотивов, будем измерять производительность в количестве сравнений позиций мотивов с позициями выборки последовательностей за единицу времени. Для набора мотивов M и выборки последовательностей D производительность GMPS вычисляется следующим образом:

$$\boldsymbol{GMPS = \frac{|M| \times |D|}{t \times 10^9} = \frac{k \times N_{mot} \times N_{seq} \times (L - k + 1)}{t \times 10^9}},$$

где $|M|$ – суммарное количество всех букв в наборе вырожденных олигонуклеотидных мотивов, $|D|$ – суммарное количество всех позиций в выборке последовательностей, с которыми возможно сравнение с мотивами, t – время работы в секундах.

С использованием предложенной меры мы провели оценку эффективности использования разработанного метода на двух графических ускорителях: NVidia GTX 1070 и NVidia Tesla v100. На рис. 5 приведены сравнительные оценки производительности на различных вычислительных платформах на выборке в зависимости a) от длины последовательностей и b) от их количества. Длина префикса – 5.

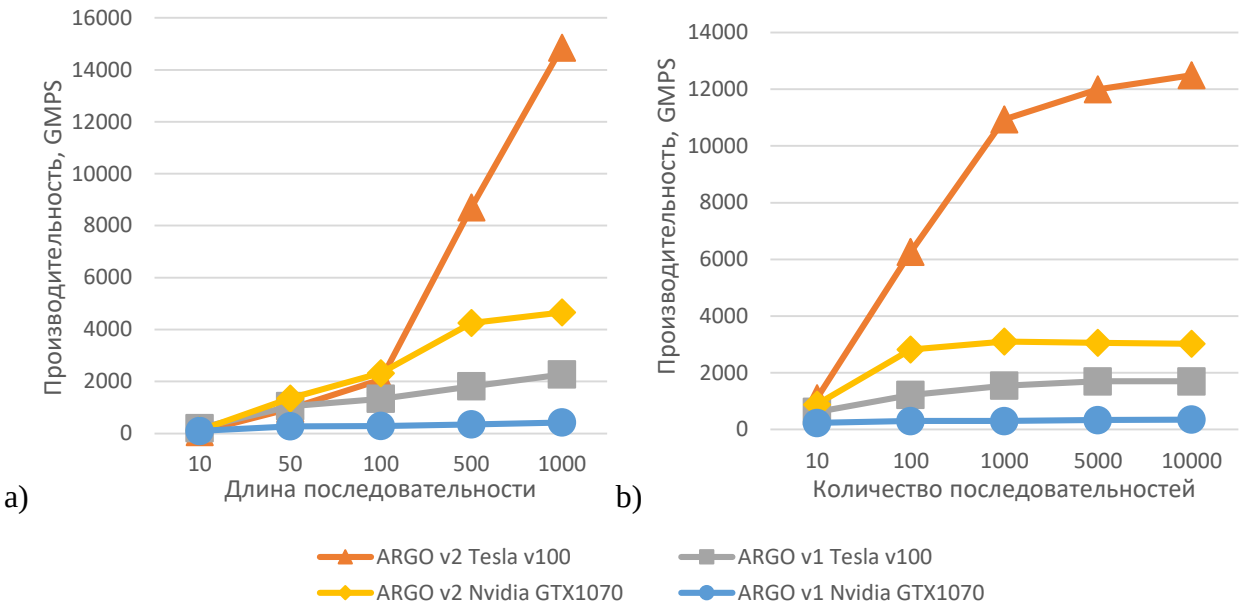


Рис. 5. Оценка производительности GMPS (Giga matches per second, ось Y)
на различных устройствах в зависимости
a) от длины последовательностей (ось X), количество последовательностей – 100;
b) от количества последовательностей (ось X), длина последовательностей – 128

Из рис. 5a и 5b хорошо видно, что NVidia Tesla v100, обладающая 5120 процессорами и более современной архитектурой, имеет в среднем в 4 раза более высокую производительность по сравнению с NVidia GTX 1070 (2048 ядер). Отметим, что оба графические ускорителя имеют худшую производительность при малых длинах анализируемых последовательностей и их малом количестве. Это можно объяснить тем, что внутренний цикл становится достаточно коротким и запуск вычислительного ядра происходит чаще. Можно отметить, что при количестве последовательностей от 1000 производительность обоих GPU стабилизируется. При этом производительность системы растет с увеличением размера выборки.

Рис. 5 ясно демонстрирует, что на всех рассмотренных GPU-устройствах при всех используемых параметрах анализируемых выборок предложенный нами новый алгоритм оценки представленности мотивов обеспечивает в среднем десятикратный прирост производительности.

3.2. Выявление олигонуклеотидных мотивов в промоторах генов голодающих мышей

Список дифференциально экспрессирующихся генов мышей был получен из [10]. Он содержал 968 генов, повышающих экспрессию в ответ на голодание, и 610 генов, понижающих экспрессию. Из базы данных Ensembl [11] с использованием системы BioMart [12] были получены последовательности промоторов в районе [-200; +1] относительно старта транскрипции генов, повышающих (выборка Neg) экспрессию в ответ на голодание и понижающих экспрессию (выборка Pos). Обе выборки были проанализированы с использованием предложенного нами нового метода.

Анализ полученных олигонуклеотидных мотивов с известными регуляторными мотивами проводился с помощью системы Tomtom [13]. Функциональная аннотация генов, промоторы которых содержали найденные сигналы, проводилась с использованием системы DAVID [8, 9].

Проведенный анализ показал, что мотивы выборки Neg имеют достоверное сходство с такими сайтами связывания транскрипционных факторов (ССТФ), как NFYA, FOXI1, CEBPZ, NFYB, NFYC, PBX3, SP4, ZN335, SP2, а мотивы выборки Pos имеют достоверное сходство с такими ССТФ, как SP1, E2F4, EGR1, SALL4, GLI2, KLF8, SRBP2, KLF5, ARNT2, ZIC1, HTF4, E2F7, PLAG1.

Функциональная аннотация генов, промоторы которых содержали мотивы, найденные в выборке Neg, показала их достоверную ассоциацию с GO терминами ответа на эндоплазматический ретикулярный стресс (ERS). В их число входили такие хорошо известные гены – регуляторы или эффекторы ответа на ERS, как синоволин 1 (Syvn1), hypoxia up-regulated 1 (Hyou1) и белки теплового шока (Hspa5, Hsp90b1).

Мотивы из выборки Pos, найденные в генах, достоверно ассоциированы с широким кругом биологических процессов, связанных с нормальным развитием и функционированием нейронов.

4. Заключение

Нами предложен новый высокопроизводительный полнопереборный алгоритм для выявления вырожденных олигонуклеотидных мотивов в больших выборках нуклеотидных последовательностей, основанный на дереве префиксов мотивов. С его помощью в промоторах дифференциально экспрессирующихся генов мышей были выявлены контекстные сигналы, достоверно ассоциированные с голоданием.

Литература

1. *Pesole G, Liuni S, Dsouza M.* PatSearch: A pattern matcher sontware that finds functional elements in nucleotide and protein sequences and assesses their statistical significance // Bioinformatics. 2000. V. 16, № 5. P. 439–450.
2. *Marsan L, Sagot M. F.* Algorithms for extracting structured motifs using a suffix tree with an application to promoter and regulatory site consensus identification // J Comput Biol. 2000. V. 7 (3–4). P. 345–362.
3. *Hertz G, Stormo G.* Identifying DNA and protein patterns with statistically significant alignments of multiple sequences // Bioinformatics. 1999. V. 15 (7–8). P. 563–577.
4. *Grundy W. N., Bailey T. L., Elkan C. P.* ParaMEME: A parallel implementation and a web interface for a DNA and protein motif discovery tool // CABIOS. 1996. V. 12. P. 303–310.
5. *Lawrence C. E., Altschul S. F., Boguski M. S., Liu J. S. et al.* Detecting subtle sequence signals: A Gibbs sampling strategy for multiple alignment // Science. 1993. V. 262, № 5131. P. 208–214.
6. *Nickolls J., Buck I., Garland M., Skadron K.* Scalable Parallel Programming with CUDA // Queue. 2008. V. 6, № 2. P.40–53.
7. *Vishnevsky O. V., Bocharnikov A. V., Kolchanov N. A.* ARGO_CUDA: Exhaustive GPU based approach for motif discovery in large DNA datasets // Journal of Bioinformatics and Computation Biology. 2017. V. 16, № 1.
8. *Huang D. W., Sherman B. T., Lempicki R. A.* Systematic and integrative analysis of large gene lists using DAVID Bioinformatics Resources // Nature Protoc. 2009. V. 4, № 1. P. 44-57.
9. *Huang D. W., Sherman B. T., Lempicki R. A.* Bioinformatics enrichment tools: paths toward the comprehensive functional analysis of large gene lists // Nucleic Acids Res. 2009. V. 37, № 1. P. 1–13.
10. *Henry F. et al.* Cell type-specific transcriptomics of hypothalamic energy-sensing neuron responses to weight-loss // Elife. 2015. Sep 2–4.
11. *Zerbino D. R., Flicek P. et al.* Ensembl 2018 // PubMed. 2018. PMID: 29155950.
12. *Durinck S., Spellman P., Birney E., Huber W.* Mapping identifiers for the integration of genomic datasets with the R/Bioconductor package biomaRt // Nature Protocols. 2009. V. 4. P. 1184–1191.
13. *Gupta S., Stamatoyannopoulos J. A., Bailey T. L., and Noble W. S.* Quantifying similarity between motifs // Genome Biology. 2007. V. 8, № 2. R24.

Статья поступила в редакцию 26.07.2019;
переработанный вариант – 30.09.2019.

Бочарников Андрей Васильевич
аспирант, Институт систем информатики им. А. П. Ершова СО РАН (630090, Новосибирск, пр. Академика Лаврентьева, 6), e-mail: andrey.bocharnikov@gmail.com.

Игнатьева Елена Васильевна
к.б.н, старший научный сотрудник, Институт цитологии и генетики СО РАН (630090, Новосибирск, пр. Академика Лаврентьева, 10); доцент, Новосибирский государственный университет, e-mail: eignat@bionet.nsc.ru.

Вишневский Олег Владимирович
к.б.н, научный сотрудник, Институт цитологии и генетики СО РАН; ст. преподаватель, Новосибирский государственный университет, e-mail: oleg@bionet.nsc.ru

GPU-based algorithm for context analysis of the core promoter region of mouse genes differently expressed in hypothalamic energy-sensing neurons in response to weight-loss

A. Bocharnikov, E. Ignatieva, O. Vishenvskiy

De novo motif discovery in the regulatory regions of eukaryotic genes poses a complex computational problem due to the large size of datasets and huge diversity of motifs. This article suggests a new algorithm for measuring the presence of degenerate oligonucleotide motifs written as a 15-letter IUPAC code in a DNA dataset. Its performance has increased 10 times compared with the previous one. There are three key ingredients of this method. The first one is the prefix trees. The second is the relation between motif prefixes and hash ranges in the analyzed nucleotide sequences. The third consists of applying CUDA framework to the massive parallelization allowing to use affordable graphic accelerators.
The context analysis of promoter regions of mouse genes differently expressed (DEG) in hypothalamic AGRP neurons after food deprivation was performed with the proposed method. When an animal is deprived of food, AGRP neurons produce molecules that increase appetite and stimulate weight gain. The understanding of how AGRP neurons respond to weight loss is important to confront the obesity. Nowadays, this hereditary disease lacks methods of treatment and intervention strategies which would be both safe and efficient in the long term. The performed analysis revealed relevant oligonucleotide motifs that were associated with starvation.

Keywords: oligonucleotide motif, GPGPU, obesity.