СИСТЕМНЫЙ АНАЛИЗ

И.В. СЕРГИЕНКО, А.М. ГУПАЛ, А.В. ОСТРОВСКИЙ

УДК 519.217.2

ПРЕДСКАЗАНИЕ СТРУКТУРЫ ГЕНОВ С ИСПОЛЬЗОВАНИЕМ СМЕСЕЙ ВЕРОЯТНОСТНЫХ РАСПРЕДЕЛЕНИЙ

Аннотация. Рассмотрена задача восстановления последовательности скрытых состояний для смесей распределений, описываемых обобщениями цепей Маркова произвольного порядка и скрытых марковских моделей. Предложен алгоритм динамического программирования для решения этой задачи, а также его модификации, направленные на устранение рекурсии и сокращение перебора. Полученные результаты применены к задаче распознавания фрагментов генов в геномах растений.

Ключевые слова: цепь Маркова, скрытые переменные, ген, биоинформатика, нуклеотид, экзон, интрон, правдоподобие.

ВВЕДЕНИЕ

Задача определения местоположения генов в ДНК и предсказания их внутренней структуры является одним из наиболее востребованных приложений биоинформатики. Для ее решения используются методы на основе обобщенных моделей Маркова со скрытыми переменными [1], обладающие определенными недостатками — чрезмерной специфичностью и сложностью интерпретации. В [2] предложен альтернативный подход к распознаванию внутренней структуры генов, основанный на сочетании скрытых марковских моделей и цепей Маркова высших порядков; его применение для геномов растений и насекомых по эффективности сравнимо с актуальными на сегодняшний день алгоритмами.

Для повышения качества распознавания геномов высших организмов (например, млекопитающих) возможно использование композиций моделей с эксклюзивной компетентностью, в которых каждый ген распознается строго одной из составляющих моделей, подбирающейся исходя из последовательности его нуклеотидов. В [3] в качестве признаков, по которым происходит разбиение на области компетентности, рассматриваются концентрации нуклеотидов; в [4] такое разбиение строится на основе EM-алгоритма с последующей аппроксимацией с помощью метода опорных векторов (SVM). Эксклюзивность областей компетентности позволяет решать задачу восстановления структуры гена таким же образом, как и в случае использования одной модели; однако это требование приводит к сужению класса рассматриваемых распределений и, потенциально, к снижению качества распознавания. В связи с этим в данной статье рассматривается алгоритм, который работает непосредственно со смесями вероятностных распределений.

1. ПОСТАНОВКА ЗАДАЧИ

Пусть задана строка $S \in R_s^*$, отдельные символы которой принадлежат множеству наблюдаемых символов R_s. Известно, что каждый символ строки порожден

строго одним символом из множества скрытых символов R_h, т.е. S соответствует последовательность $H \in R_h^*$ той же длины; в ее нахождении и заключается задача, которая согласно принципу максимума правдоподобия имеет вид

$$\log P(H|S) \to \max_H . \tag{1}$$

Для рассматриваемой проблемы определения фрагментов генов множество наблюдаемых символов состоит из четырех элементов; эти символы соответствуют нуклеотидам (аденину, цитозину, гуанину и тимину), из последовательности которых состоят гены. Основными функциональными фрагментами любого гена являются чередующиеся между собой экзоны (участки, кодирующие определенный белок организма с помощью универсального генетического кода) и интроны (расположенные между экзонами участки, не принимающие участия в синтезе белка) [4]. Соответственно множество скрытых символов R_h состоит из двух элементов.

Упорядоченные пары из наблюдаемых и соответствующих им скрытых символов назовем полными состояниями. Для строки полных состояний $Q \in R_q^*$, где $R_q \equiv R_s \times R_h$, определены функции, возвращающие ее наблюдаемую и скрытую части:

$$\mathrm{pr}_s(Q) = S;\ \ \mathrm{pr}_h(Q) = H.$$

С учетом введенных обозначений задача (1) эквивалентна следующему соотношению:

$$\log P(Q) \to \max_Q,\ \ \mathrm{pr}_s(Q) = S. \tag{2}$$

В соответствии с общей формулировкой задачи обучения с прецедентами для определения оптимальных параметров θ^* вероятностного распределения $P(Q)$ используется конечный набор строк (обучающая выборка) $T \subset R_q^*$, для которых известны как наблюдаемая, так и скрытая части. Из принципа максимума правдоподобия следует

$$\theta^* = \arg\max_\theta \sum_{Q \in T} \log P(Q|\theta) . \tag{3}$$

В работах [2, 5] для функции правдоподобия $P(Q)$ используется формула для цепи Маркова произвольного l-го порядка

$$P(Q) = \varphi(|Q|)\pi(Q_1 \dots Q_l) \prod_{i=l+1}^{|Q|} p(Q_i | Q_{i-l} \dots Q_{i-1}) , \tag{4}$$

где φ — функция распределения строк по длине, π — распределение начальных вероятностей, p — распределение переходных вероятностей; таким образом, параметрами модели является тройка функций $\theta = (\varphi, \pi, p)$. Решение задачи (2) с учетом (4) можно найти с помощью динамического программирования; при этом из (3) следует, что оптимальные начальные и переходные вероятности модели находятся как частоты встречи соответствующих последовательностей в строках обучающей выборки.

В работах [3, 4] рассмотрена функция правдоподобия в виде смеси распределений

$$P_w(Q) = \sum_{j=1}^{k} w_j P(Q|\theta_j),\ \ w_j \ge 0,\ \ \sum_{j=1}^{k} w_j = 1 , \tag{5}$$

где w_j — априорный вес j-й модели, $P(Q|\theta_j)$ — функция распределения, определяемая по формуле (4); параметрам составной модели соответствует вектор

$$\Theta = (w_1, \theta_1; \dots ; w_k, \theta_k) = (w_1, \varphi_1, \pi_1, p_1; \dots ; w_k, \varphi_k, \pi_k, p_k) .$$

Условие эксклюзивной компетентности составляющих композиции

$$\forall Q \in R_q^* \ \exists j: 1 \le j \le k \ \ P(\theta_j | Q) = 1 \tag{6}$$

приводит к «схлопыванию» функции (5) к виду

$$P(Q) = w_j P(Q|\theta_j), \ \ j = j(\mathrm{pr}_s(Q)), \tag{7}$$

т.е. полагается, что индекс модели можно найти с помощью наблюдаемой части рассматриваемой строки. Задача максимизации функции правдоподобия (2) при выполнении равенства (7) аналогична такой задаче при использовании единственной модели (4).

В работе [3] условие (6) достигается за счет модификации функции распределения (4) таким образом, чтобы множество генерируемых им строк ограничивалось некоторой областью; это при определенных условиях не влияет на способ решения задачи нахождения параметров модели (3) по сравнению с применением одного распределения. В [4], напротив, в композиции используются исходные модели (4), а для нахождения их оптимальных параметров применяется итеративный EM-алгоритм, по результатам которого условие (6) выполняется с достаточно высокой точностью. При этом для определения области компетентности по наблюдаемой части строки Q используется SVM.

Несмотря на эффективность описанного выше подхода, условие изолированности областей компетентности (7) даже при точном приближенном выполнении (6) является несколько искусственным и не следует из каких-либо априорных соображений. Рассмотрим далее исходную задачу восстановления последовательности полных состояний для смеси распределений

$$\log P_w(Q) = \log\left(\sum_{j=1}^{k} w_j P(Q|\theta_j) \right) \to \max_{\theta}, \ \ \mathrm{pr}_s(Q) = S \ . \tag{8}$$

При этом веса составляющих моделей $\mathbf{w} \equiv (w_1, \ldots, w_k)$ и их параметры $\{(\varphi_j, \pi_j, p_j)\}_{j=1}^{k}$ полагаем заданными.

2. АЛГОРИТМ РЕШЕНИЯ

Для произвольной строки $x \in R_q^*$, символа $y \in R_q$ и неотрицательных весов $\mathbf{u} \equiv (u_1, \ldots, u_k)$ справедливо равенство

$$\log\left(\sum_{j=1}^{k} u_j P(xy|\theta_j) \right) = \log\left(\sum_{j=1}^{k} u_j P(x|\theta_j) p_j(y|x) \right) =$$

$$= \log\left(\sum_{j=1}^{k} \widetilde{u}_j P(x|\theta_j) \right) + \log C \ , \tag{9}$$

где

$$C = C(x, y, \mathbf{u}) = \sum_{j=1}^{k} u_j p_j(y|x) \ \ \forall j = 1, \ldots, k, \ \ \widetilde{u}_j = \widetilde{u}_j(x, y, \mathbf{u}) = \frac{u_j p_j(y|x)}{C} \ . \tag{10}$$

Пусть $F(i, x, \mathbf{u})$ — максимум для задачи (8) для префикса строки Q длины $i > l$ и весов составляющих моделей $\mathbf{u}$ при дополнительном условии, что последние l ее полных состояний фиксированы:

$$F(i, x, \mathbf{u}) = \max \log\left(\sum_{j=1}^{k} u_j P(Q_1 \ldots Q_i | \theta_j) \right), \ \ \mathrm{pr}_s(Q) = S, \ \ Q_{i-l+1} \ldots Q_i = x. \tag{11}$$

Тогда согласно (9)

$$F(i,x,\mathbf{u}) = \max_y \left(F(i-1,\, yx_1 \ldots x_{l-1}, \tilde{\mathbf{u}}(x,\, y,\, \mathbf{u})) + \right.$$

$$\left. + \log\left(\sum_{j=1}^{k} u_j\, p_j(x_l \,|\, yx_1 \ldots x_{l-1}) \right) \right), \qquad (12)$$

где максимум берется по полным состояниям $y \in R_q$, соответствующим наблюдаемой строке: $\mathrm{pr}_s(y) = S_{i-l}$; веса $\tilde{\mathbf{u}} = (\tilde{u}_1, \ldots, \tilde{u}_k)$ вычисляются по формуле (10). Таким образом, решение задачи (11) определяется решением нескольких подобных задач меньшего размера, что позволяет решать ее методом динамического программирования, так же как при использовании функции правдоподобия (4). Граничными условиями при рекурсивном вычислении функции F по формуле (12) являются равенства вида

$$F(l,x,\mathbf{u}) = \log\left(\sum_{j=1}^{k} u_j \pi_j(x) \right). \qquad (13)$$

При описанном способе вычисления функции (11) определение решения исходной задачи (8) тривиально:

$$\max \log P_w(Q) = \max_x F(|Q|, x, \mathbf{w}), \qquad (14)$$

где перебор выполняется по строкам полных состояний x длины l, которые соответствуют суффиксу наблюдаемой строки S: $\mathrm{pr}_s(x) = S_{|S|-l+1} \ldots S_{|S|}$. Оптимальное решение Q восстанавливается путем запоминания оптимальных полных состояний при вычислении максимумов (12) и (14).

В отличие от задачи, рассмотренной в [5], формулы (12)–(14) не позволяют вычислить рекуррентную функцию $F(i,x,\mathbf{u})$, итеративно наращивая размерность задачи, т.е. переменную i. Два первых параметра функции принимают конечное количество значений ($l \le i \le |S|$, $x \in R_q^l$), а множество весов $\mathbf{u}$ имеет мощность континуума и не может быть предсказано априорно (за исключением тривиального случая, когда композиция состоит из одной модели). Таким образом, возможным способом вычисления функции F является непосредственная имплементация рекуррентной формулы (12) с запоминанием промежуточных значений. При этом начальному этапу работы алгоритма согласно формуле (14) соответствует следующий псевдокод.

Алгоритм 1. Определение строки Q с рекурсией.

Дано: наблюдаемая строка S, веса моделей $\mathbf{w}$, параметры моделей $\{(\varphi_j, \pi_j, p_j)\}_{j=1}^k$.

1. $M := \varnothing$; — кэш-память для хранения значений функции F;
2. **для всех** $x \in R_q^l$, $\mathrm{pr}_s(x) = S_{|S|-l+1} \ldots S_{|S|}$:
3. $(F^-[x], Q[x]) := FR(|Q|, x, \mathbf{w})$;
4. $x^* := \arg\max_x F^-[x]$;
5. **вернуть** $Q[x^*]\, x^*$.

Здесь полагается, что рекурсивная процедура $FR(i,x,\mathbf{u})$ возвращает, помимо значения функции F для соответствующего набора аргументов, оптимальную последовательность полных состояний $Q_1 \ldots Q_{i-l}$, восстановленную за счет запоминания значений переменной y, которые максимизируют (12).

Алгоритм 2. Процедура $FR(i, x, \mathbf{u})$.

1. **если** $i = l$:
2. вычислить значение функции F по формуле (13);
3. **вернуть** (F, ε) (ε — пустая строка);
4. **иначе: если** $(i, x, \mathbf{u})$ нет в кэш-памяти M:
5. **для всех** $y \in R_q$, $\mathrm{pr}_s(y) = S_{i-l}$:
6. $(F^-[y], Q[y]) := FR(i-1, yx_1 \ldots x_{l-1}, \tilde{\mathbf{u}}(x, y, \mathbf{u}))$;
7. найти оптимальное состояние y^* и значение функции F согласно формуле (12), исходя из вычисленных значений F^-;
8. занести оптимум в кэш-память: $M[i, x, \mathbf{u}] := (F, Q[y^*]y^*)$;
9. **вернуть** $M[i, x, \mathbf{u}]$ — значение из кэш-памяти.

3. МОДИФИКАЦИИ И ЭВРИСТИКИ

Поскольку множество различных рассматриваемых весов для каждого набора величин (i, x) быстро растет с уменьшением i, непосредственное использование приведенного алгоритма для длинных строк S не представляется возможным. Это ограничение можно обойти, если предположить, что для близких векторов весов $\mathbf{u}$ значения функции F также мало отличаются:

$$\rho(\mathbf{u}, \mathbf{u}') \le \rho_{\min} \Rightarrow F(i, x, \mathbf{u}) \approx F(i, x, \mathbf{u}'), \tag{15}$$

где метрика ρ соответствует пространству ограниченных последовательностей ℓ^∞:

$$\rho(\mathbf{u}, \mathbf{u}') = \max_{1 \le j \le k} |u_j - u'_j|.$$

Исходя из эвристики, если при вызове процедуры $FR(i, x, \mathbf{u})$ в кэш-памяти M содержится результат работы процедуры для тех же параметров (i, x) и близкого к $\mathbf{u}$ вектора весов, то дальнейших вызовов процедуры не происходит, а вместо этого немедленно возвращается сохраненное значение. Чем меньше пороговое расстояние $\rho_{\min}$, тем точнее аппроксимация (15) и соответственно ниже скорость работы алгоритма и выше объем потребляемой им памяти. Значение $\rho_{\min} = 0$ отвечает исходному алгоритму.

При реализации кэш-памяти M в виде хэш-таблицы простейшим способом реализации эвристики является использование в качестве ключей таблицы объектов, для которых хэш-функция не зависит от весов $\mathbf{u}$, а операция равенства учитывает формулу (15):

$$hash(i, x, \mathbf{u}) = \mathrm{const}(\mathbf{u});$$

$$(i, x, \mathbf{u}) \sim (i', x', \mathbf{u}') \Leftrightarrow (i = i') \wedge (x = x') \wedge (\rho(\mathbf{u}, \mathbf{u}') \le \rho_{\min}).$$

Недостаток такого подхода — высокое количество коллизий по хэш-функции; тем не менее для рассматриваемой задачи он оказался достаточно эффективным.

Глубина стека вызовов процедуры FR составляет $\Theta(|S|)$, что может привести к его переполнению. Более того, поскольку результат каждого вызова содержит строку длины $\Theta(|S|)$, требуемый алгоритмом объем памяти составляет порядка $|S|^2$, что недопустимо много даже для современных операционных сред. Для решения этих проблем необходим переход к нерекурсивному вычислению функции F за счет использования очереди вызовов, реализованной в виде линейного массива. При этом каждый элемент очереди соответствует отдельному вызову процедуры FR с определенным набором параметров и является объектом со

следующим набором полей:

- $i, x, \mathbf{u}$ — аргументы вызова;
- F — значение функции;
- y^* — оптимальное полное состояние при вычислении функции по формуле (12);
- ch — массив из индексов элементов очереди, отвечающих дочерним вызовам, необходимым для вычисления F.

Размер объекта, таким образом, составляет $O(k+|R_h|)$. Кэш-память в рамках модификации применяется для запоминания и последующего использования номеров элементов очереди, соответствующих различным наборам аргументов $(i, x, \mathbf{u})$.

Определение оптимальной строки Q состоит из двух этапов, первым из которых является формирование очереди вызовов и установление связей между ее элементами.

Алгоритм 3. Определение строки Q без рекурсии (прямой ход).

1. инициализировать очередь: $q := \varnothing$;
2. **для всех** $x \in R_q^l$, $\mathrm{pr}_s(x) = S_{|S|-l+1} \dots S_{|S|}$:
3. добавить в q вызов с параметрами $(|S|, x, \mathbf{w})$;
4. **пока** в очереди есть нерассмотренные элементы:
5. взять из q первый нерассмотренный элемент e;
6. **если** $e.i = l$:
7. вычислить значение $e.F$ по формуле (13);
8. **иначе**:
9. **для всех** $y \in R_q$, $\mathrm{pr}_s(y) = S_{i-l}$:
10. определить параметры нового вызова $(e.i-1, \tilde{x}, \tilde{\mathbf{u}})$;
11. **если** $(e.i-1, \tilde{x}, \tilde{\mathbf{u}})$ нет в кэш-памяти M:
12. добавить в q элемент с аргументами $(e.i-1, \tilde{x}, \tilde{\mathbf{u}})$;
13. $M[e.i-1, \tilde{x}, \tilde{\mathbf{u}}] := |q|$;
14. запомнить ссылку на дочерний вызов: $e.ch[y] := M[e.i-1, \tilde{x}, \tilde{\mathbf{u}}]$.

Вторым этапом работы алгоритма является вычисление по составленному дереву вызовов значений рекуррентной функции и определение на их основе оптимальной скрытой строки.

Алгоритм 4. Определение строки Q без рекурсии (обратный ход).

1. **для** $j = |q|, |q|-1, \dots, 1$:
2. **если** $e.i > 1$:
3. вычислить $e.F$ по формуле (12) исходя из значений $\{q[e.ch[y]].F\}$ и определить $e.y^*$;
4. найти элемент очереди, соответствующий вызову вида $FR(|S|, x, \mathbf{w})$ с оптимальным значением $x \in R_q^l$:

$$ptr := \arg\max_j q[j].F,\ 1 \le j \le |R_q^l|;$$

5. восстановить суффикс строки Q: $Q_{|S|-l+1} \dots Q_{|S|} := q[ptr].x$;
6. **пока** $ptr > 0$:
7. $e := q[ptr]$; $ptr := e.ch[e.y^*]$;
8. $Q_{e.i} := e.y^*$.

Методом математической индукции можно показать, что элементы очереди расположены по невозрастанию длины i:

$$\forall j, j' : 1 \le j < j' \le |q| \Rightarrow q[j].i \ge q[j'].i.$$

Следовательно, индексы дочерних вызовов любого элемента больше индекса самого элемента:

$$\forall j : 1 \le j \le |q| \quad q[j].ch[y] > j ,$$

поэтому на шаге 3 алгоритма 4 значения $\{q[e.ch[y]].F\}$ вычислены ранее, что свидетельствует о корректности определения значений функции F для всех объектов в очереди q за один проход.

Сложность алгоритма 3 определяется двумя вложенными циклами на шагах 4 и 9. Сложность определения параметров нового вызова функции на шаге 10 составляет $O(k)$. Следовательно, при условии, что доступ к элементам кэш-памяти осуществляется в среднем за $O(1)$, временные затраты на алгоритм 3 равны $O(k|q||R_h|)$. Сложность алгоритма 4 составляет $O(k|q|)$ за счет цикла на шаге 1; таким образом, общая вычислительная сложность восстановления цепочки полных состояний равна $O(k|q||R_h|)$. Память, используемая алгоритмом, определяется объемом хэш-таблицы и составляет $O((k+|R_h|)|q|)$.

В обеих оценках сложности учитывается размер очереди после завершения прямого хода вычислений — $|q|$; выразить его через другие переменные не представляется возможным из-за сильной зависимости от граничного расстояния $\rho_{\min}$. Нижней границей для $|q|$ в общем случае является количество различных наборов (i, x), для которых вычисляется функция F: $|q| \ge |S| \cdot |R_h^l|$. Эту оценку можно улучшить, если после определения параметров дочернего вызова на шаге 10 алгоритма 3 добавлять элемент в очередь только в том случае, когда значение величины C, вычисленной по формуле (10), является строго положительным; в противном случае значение функции гарантированно равно $-\infty$. Для рассматриваемой задачи распознавания фрагментов генов эта модификация является особенно эффективной, так как большинство переходных вероятностей в составляющих композицию моделях равно нулю.

4. ВЫЧИСЛИТЕЛЬНЫЙ ЭКСПЕРИМЕНТ

Для определения качества распознавания предложенных алгоритмов были рассмотрены геномы растений *Glycine max* (соя культурная), *Oryza sativa* (рис посевной), *Populus trichocarpa* (тополь бальзамический), *Sorghum bicolor* (двухцветное сорго) и *Vitis vinifera* (виноград культурный), взятые из репозитория NCBI [6]. Во внимание принимались гены, для которых полностью известна нуклеотидная запись (около 90% от всех генов для всех рассматриваемых видов). Для реализации алгоритмов использовался язык программирования Java.

Оптимальные смеси распределений (5) строились с помощью описанного в [4] итеративного EM-алгоритма с последовательным добавлением компонентов смеси; для каждого числа компонентов выполнялось десять итераций. При этом порядок цепей Маркова был принят равным $l = 6$ как обеспечивающий максимальное качество распознавания для большинства растений при использовании простых вероятностных моделей вида (4).

Первый этап эксперимента — проверка обоснованности описанной в разд. 3 эвристики по сокращению размера очереди вызовов для вычисления рекуррентной функции. Для каждого из ста случайным образом отобранных генов каждого вида сравнивались три значения логарифмического правдоподобия:

- правдоподобие для действительной последовательности полных состояний — $\log P^{(a)}$;
- правдоподобие, вычисленное в процессе работы алгоритма 4, — $\log P^{(q)}$;
- правдоподобие, вычисленное путем подстановки восстановленной алгоритмом последовательности полных состояний в формулу (5), — $\log P^{(e)}$.

Таблица 1. Характеристики выполнения алгоритмов 3, 4 при использовании смеси из двух распределений

Геном	$\rho_{\min}$ (10^{-3})	Длина гена, S	Размер очереди, q	$\log P^{(e)}$	Δ_{qe}	Δ_{ae}
Glycine max	5	3568	567765	–4662	1,163	6,955
Oryza sativa	1	2379	220667	–3124	0,546	5,946
Populus trichocarpa	2,5	2268	897660	–2991	0,271	5,337
Sorghum bicolor	1	2408	241832	–3196	0,863	8,634
Vitis vinifera	2,5	3467	951629	–4555	0,580	5,104

Таблица 2. Характеристики выполнения алгоритмов 3, 4 для десяти генов *Glycine max* при использовании смеси из двух распределений с $\rho_{\min} = 5 \cdot 10^{-3}$

Номер гена	Длина гена, S	Размер очереди, q	$\log P^{(q)}$	$\log P^{(e)}$	$\log P^{(a)}$
1	1387	137054	–1787,60	–1787,63	–1787,63
2	1587	300279	–2100,12	–2100,06	–2100,06
3	2049	418185	–2675,63	–2675,63	–2675,63
4	3632	603760	–4798,59	–4797,61	–4792,29
5	3865	784925	–5042,09	–5042,13	–5043,73
6	4093	388386	–5195,06	–5195,04	–5201,65
7	5681	949672	–7461,74	–7461,23	–7473,25
8	6075	1200067	–8072,33	–8072,38	–8082,89
9	6347	1093695	–8233,01	–8233,02	–8235,90
10	11876	2348274	–15773,02	–15773,06	–15795,49

На основании приведенных правдоподобий вычислялись разности:

$$\Delta_{qe} = |\log P^{(q)} - \log P^{(e)}|; \ \Delta_{ae} = |\log P^{(a)} - \log P^{(e)}|.$$

Результаты вычислений средних значений (табл. 1) свидетельствуют о том, что гипотеза (15) выполняется с достаточной точностью: $\log P^{(e)} \approx \log P^{(q)}$; помимо этого, из приближенного равенства $\log P^{(e)} \approx \log P^{(a)}$ следует применимость для рассматриваемой задачи принципа максимума правдоподобия. Точность выполнения равенств уменьшается с увеличением длины рассматриваемой строки S (табл. 2).

На втором этапе эксперимента исследована зависимость размера очереди q, используемой в алгоритме 3, от длины строки наблюдаемых состояний S, а также от граничного расстояния $\rho_{\min}$. Как оказалось, зависимость достаточно точно приближается степенной функцией

$$|q| = K|S|^{\alpha}, \ \alpha > 1,$$

причем значение показателя α увеличивается с уменьшением граничного расстояния (рис. 1). Для каждого из рассматриваемых геномов определено значение $\rho_{\min}$, приведенное в табл. 1, при котором по меньшей мере 90% генов образуют очередь длиной не более $5 \cdot 10^6$ элементов. Ограничение на длину очереди связано с доступным объемом памяти, а также временем выполнения алгоритма, которые, как выяснено в разд. 3, прямо зависят от $|q|$. Определенные в результате эксперимента значения использовались в дальнейших вычислениях.

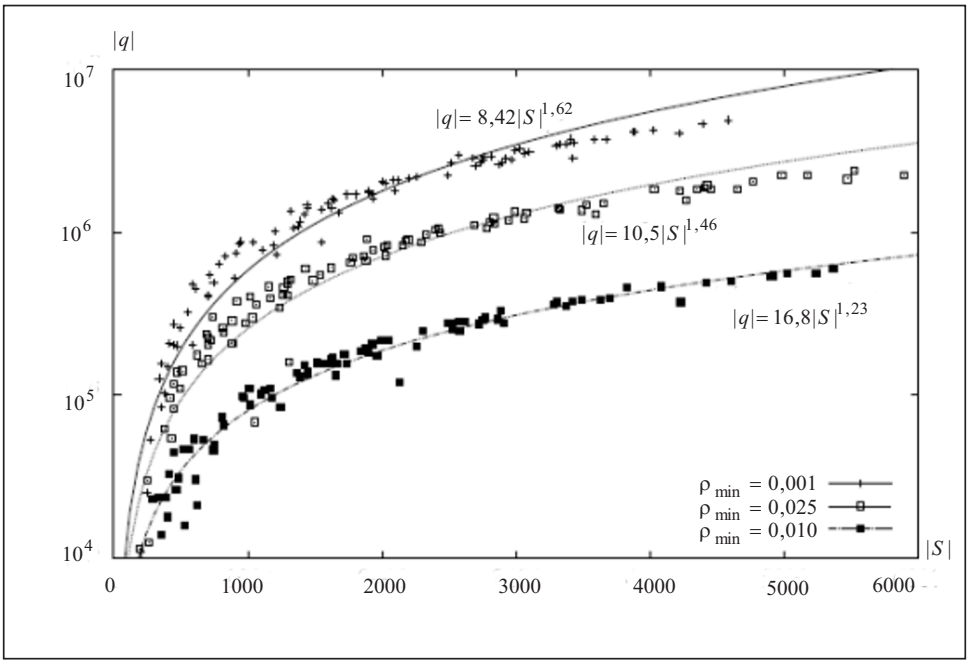

Рис. 1. Зависимость размера очереди $|q|$ от длины гена $|S|$ для генома *Populus trichocarpa* и их аппроксимация степенными функциями для трех различных значений граничного расстояния ρ_{min}

Заключительный этап эксперимента — сравнение эффективности моделей (5) с простыми вероятностными моделями (4), а также моделями с эксклюзивной компетентностью составляющих, рассмотренных в [5]. Для оценки качества моделей использовались описанные в [1] метрики:

• меры, определяющие качество предсказания отдельных скрытых состояний, — нуклеотидная специфичность *NSp*, нуклеотидная чувствительность *NSn*, коэффициент корреляции *CC* и средняя условная вероятность *ACP*;

• меры, определяющие качество распознавания границ между экзонами и интронами, — экзонная специфичность *ESp* и экзонная чувствительность *ESn.*

Таблица 3. Меры качества распознавания для подходов из [2, 4] и алгоритмов 3, 4

Геном	Число алгоритмов	Значения меры качества, %					
		NSp	*NSn*	*CC*	*ACP*	*ESp*	*ESn*
Glycine max	1	80,96	89,86	77,65	88,84	69,19	64,01
	2 эксклюзивных	86,62	90,90	83,08	91,54	70,29	66,32
	2	87,72	92,62	84,41	92,21	70,68	68,43
Oryza sativa	1	90,61	78,65	72,58	86,29	61,60	48,16
	2 эксклюзивных	89,96	88,99	80,28	90,14	62,42	54,46
	2	89,93	88,55	79,86	89,93	64,45	56,86
Populus trichocarpa	1	87,94	93,26	82,67	91,33	69,59	61,36
	2 эксклюзивных	92,09	93,55	86,97	93,49	69,77	64,00
	2	93,14	94,20	87,43	93,72	69,74	64,33
Sorghum bicolor	1	84,44	76,06	64,66	82,33	58,01	43,36
	2 эксклюзивных	86,62	90,90	83,08	91,54	70,29	66,32
	2	87,72	92,62	84,41	92,21	70,68	68,43
Vitis vinifera	1	80,28	87,38	77,42	88,72	64,42	55,37
	2 эксклюзивных	83,50	88,67	80,69	90,35	64,06	56,58
	2	89,54	91,90	85,12	92,56	67,00	59,98

Значения всех метрик находятся на отрезке [0,1]; бóльшим значениям соответствует лучшее качество алгоритма.

Для контроля эффекта переобучения (чрезмерной специализации параметров модели) применялась кросс-валидация [7]: выборка случайным образом разбивалась на пять приблизительно равных частей, каждая из которых по очереди применялась в качестве контрольной, а оставшиеся четыре части формировали обучающую выборку; полученные значения метрик качества после этого усреднялись. Как и в [4], в целях ускорения вычислений при обучении использовались веса и параметры составляющих композицию моделей, полученные при работе EM-алгоритма на всей доступной выборке.

Согласно результатам работы (табл. 3) применение смесей моделей позволяет повысить качество распознавания для всех исследуемых видов геномов. При этом для всех видов, кроме *Oryza sativa*, качество, показываемое алгоритмами 3, 4, выше, чем при использовании метода с эксклюзивными зонами компетентности, описанного в [4].

ЗАКЛЮЧЕНИЕ

Рассмотрена в общем виде задача восстановления последовательности скрытых состояний для смесей вероятностных распределений, имеющих вид цепей Маркова произвольного порядка. Для ее решения предложен алгоритм динамического программирования с эвристикой, позволяющей значительно сократить перебор. Полученный алгоритм применен к задаче распознавания фрагментов генов для пяти геномов растений, что ощутимо повысило качество распознавания этих геномов.

Несмотря на достаточную эффективность описанной в разд. 3 эвристики для рассмотренных организмов, в случае более сложных геномов (например, млекопитающих) ее использование фактически невозможно: при больших граничных расстояниях $\rho_{\min}$ формула (15) выполняется с неудовлетворительной точностью; при меньших значениях $\rho_{\min}$ недопустимо большим становится размер очереди q в алгоритме 3. Та же проблема возникает при увеличении числа составляющих в смеси распределений (5). Это обусловливает поиск других методов вычисления рекуррентной формулы (12). Среди возможных направлений для дальнейших исследований отметим тестирование работы алгоритма на сходных задачах (например, для определения пространственной структуры белков) [5, 8].

СПИСОК ЛИТЕРАТУРЫ

1. Knapp K., Chen Y.-P.P. An evaluation of contemporary hidden Markov model genefinders with a predicted exon taxonomy // Nucleic Acids Research. — 2007. — **35.** — P. 317–324.
2. Сергиенко И.В., Гупал А.М., Островский А.В. Распознавание фрагментов генов в ДНК с применением моделей Маркова со скрытыми переменными // Кибернетика и системный анализ. — 2012. — № 3. — С. 58–67.
3. Гупал А.М., Островский А.В. Использование композиций моделей Маркова для определения функциональных участков генов // Кибернетика и системный анализ. — 2013. — № 5. — С. 61–68.
4. Сергиенко И.В., Гупал А.М., Островский А.В. Использование EM-алгоритма для классификации генов // Кибернетика и системный анализ. — 2015. — № 1. — С. 48–58.
5. Островский А.В. Определение вторичной структуры белков с помощью моделей Маркова // Международный научно-технический журнал «Проблемы управления и информатики». — 2013. — № 2. — С. 140–147.
6. Национальный центр биотехнологической информации США. — http://ncbi.nlm.nih.gov/.
7. Ng A.Y. Preventing overfitting of cross-validation data // Proc. 14th Intern. Conf. on Machine Learning. — Waltham: Morgan Kaufmann, 1997. — P. 245–253.
8. Предсказание вторичной структуры белков на основе байесовских процедур распознавания на цепях Маркова / И.В. Сергиенко, Б.А. Белецкий, С.В. Васильев, А.М. Гупал // Кибернетика и системный анализ. — 2007. — № 2. — С. 59–64.

Поступила 03.07.2014