J. Appl. Prob. **48,** 1049–1059 (2011)
Printed in England

DISTRIBUTION OF CLUMP STATISTICS FOR A COLLECTION OF WORDS

DONALD E. K. MARTIN * ** AND
DEIDRA A. COLEMAN,* *** *North Carolina State University*

Abstract

We give an efficient method based on minimal deterministic finite automata for computing the exact distribution of the number of occurrences and coverage of clumps (maximal sets of overlapping words) of a collection of words. In addition, we compute probabilities for the number of h-clumps, word groupings where gaps of a maximal length h between occurrences of words are allowed. The method facilitates the computation of p-values for testing procedures. A word is allowed to contain other words of the collection, making the computation more general, but also more difficult. The underlying sequence is assumed to be Markovian of an arbitrary order.

Keywords: Clumps of a pattern; coarsest partition; deterministic finite automaton

2010 Mathematics Subject Classification: Primary 60E05
Secondary 60J05

1. Introduction

In recent years, methods have been established to compute distributions associated with increasingly complex patterns, driven by statistical applications in many fields, such as reliability theory, national and computer security, and computational biology. Reviews of some of the theory on distributions of patterns and its applications are given in [3] and [13].

Depending on the problem at hand, one may be interested in studying distributional properties of pattern occurrences using various counting techniques. With *overlapping* counting (see, e.g. [2] and [11]), all occurrences of words of the pattern are counted. When counting *renewals* (see, e.g. [5]), a word occurrence restarts counting for all words in the system. Patterns may also be counted based on *clump counting*, the topic of this paper.

Distributions associated with clumping of patterns can be useful for analyzing DNA sequences. Examples include searching for exceptional patterns in DNA that could have functional roles, such as transcription binding sites [15], the study of homogeneity across a sequence [8], disentangling overlapping word occurrences [16], and seeded searches for similar DNA segments, where the number of successes in success runs of length at least k (coverage of clumps of $\overbrace{111\ldots11}^{k}$) was used as an initial screening criterion for determining candidate tandem repeats [4]. Detection of clumping effects can be useful in other fields as well, for example clumping of anomalous events is used in intrusion detection for information systems [7].

Received 12 January 2010; revision received 17 June 2011.
* Postal address: Department of Statistics, North Carolina State University, 4272 SAS Hall, Raleigh, NC 27695-8203, USA.
** Email address: martin@stat.ncsu.edu
*** Email address: dacolem2@ncsu.edu

We compute distributions of statistics associated with clumps of a *collection* of words in a Markovian sequence of an arbitrary order. Words of the collection are allowed to be completely contained in another word, a feature that greatly increases the complexity of the computations. We allow such generality because it is hard to anticipate all of the types of pattern systems for which a researcher may be interested in obtaining probabilistic results. The computation is carried out by associating a minimal deterministic finite automaton with the collection of words, and using a transition matrix (and its powers) to hold probabilities for automaton transitions, keeping track of statistic updates and computing coefficients of a probability generating function. This paper expands on the fine work of [17], where probability generating functions were used to compute distributions associated with clumps of a *single* word.

The paper is organized as follows. In the next section some notation and formal definitions are given. In Section 3 we give details and examples of using the computation algorithm. Section 4 contains an application to computing p-values for the observed number of clumps and clump coverage of the Chi motif in *Haemophilus influenzae*. The final section contains a summary.

2. Definitions and notation

Let $\boldsymbol{X} \equiv X_1, \ldots, X_n$ be a stationary mth order Markovian sequence, with observed values denoted by $x_1, \ldots, x_n$, and each X_j taking values in a finite alphabet Σ. Let π be the stationary distribution of the sequence over m-tuples $\tilde{x}_t \equiv (x_{t-m+1}, \ldots, x_t)$ that are embedded into a first-order Markov chain, so that π also serves as the initial distribution for $\tilde{x}_m$, and let T be the transition probability matrix for m-tuples.

A *word* is a finite string of symbols from Σ. A substring of a word is a *factor*. A factor that starts at the beginning of a word is its *prefix*, and a factor that ends at the end of the word is its *suffix* (the prefix or suffix is *proper* if it does not consist of the entire word). A *pattern* is a subset of Σ^+, the set of all nonempty words over Σ. Let the pattern $W \equiv \{w^{(1)}, \ldots, w^{(c)}\}$ consist of words $w^{(j)}$, $j = 1, \ldots, c$, satisfying $|w^{(j)}| \geq 2$, where the length of a string s is denoted by $|s|$. A word w of length $|w|$ *occurs* at x_t (at sequence location t) if $(x_{t-|w|+1}, \ldots, x_t) = w$.

Let (β_j, e_j) denote the beginning and ending sequence locations of word $w^{(j)}$. For words $w^{(j)}$ and $w^{(k)}$, if $\beta_j < \beta_k \leq e_j < e_k$, the words are said to *overlap,* whereas if $\beta_j \leq \beta_k < e_k < e_j$, $w^{(k)}$ is *enclosed in* $w^{(j)}$. The gap $g_{i,\kappa}$ between a word $w^{(\kappa)} \in W$ and the previously occurring word $w^{(i)} \in W$ is $g_{i,\kappa} = \beta_\kappa - e_i - 1$.

Definition 2.1. An *h-clump* of W (called an h-gap cluster in [17]) is a string that begins (and ends) with a word of W and has gaps $g \leq h$ for any subsequent word occurrences of the clump. When $h = -1$, we call the string a *clump*.

A clump may be composed of only the word $w^{(i)}$ that defines its existence. If other words of W overlap $w^{(i)}$ and each other on the right, then, by definition, all gaps satisfy $g \leq -1$, and the clump continues. If, on the other hand, $w^{(i)}$ is enclosed in a word $w^{(j)} \in W$, the occurrence of $w^{(i)}$ signals the occurrence of a clump, and since $g_{i,j} \leq -2$, with the occurrence of $w^{(j)}$, the clump continues. In general, a word of W that has one or more enclosed h-clumps is counted as a single h-clump. This opens up the possibility that the clump count will actually need to be *reduced* if a word that contains two or more enclosed clumps occurs, since the enclosed clumps have already been counted when the word occurs.

The sequence positions of a clump are said to be *covered*, so that clump coverage for the sequence is the total number of sequence positions that lie in a clump. If (η, γ) or $(\eta^{(h)}, \gamma^{(h)})$

https://doi.org/10.1239/jap/1324046018 Published online by Cambridge University Press

are respectively the number and coverage of clumps (or h-clumps) in $\boldsymbol{X}$, then we take $\gamma^{(h)} = \gamma$ (not counting gaps in coverage), whereas $\eta^{(h)} \leq \eta$.

A clump is also characterized by the following theorem.

Theorem 2.1. *A clump of W is a string such that all of its two-letter factors are a factor of at least one word of W that occurs in $\boldsymbol{X}$.*

Proof. Let the string s be a clump of W. If s consists of a word $w^{(i)} \in W$ then all of its two-letter factors are a factor of $w^{(i)}$. Let s consist of $w^{(i)}$ and one or more words of W that overlap $w^{(i)}$ (and themselves) on the right, and assume that s has a two-letter factor that is not a factor of at least one occurrence of a word of W. This implies that the gap between word occurrences is at least 0, a contradiction.

Now assume that, for string s, every two-letter factor lies in at least one occurrence of a word of W. Assume that there are two consecutive word occurrences that do not overlap and one is not enclosed in the other. Then, if e_j is the ending sequence position of the first word occurrence then the two-letter factor $(e_j, e_j + 1)$ is not in an occurrence of a single word of W, a contradiction.

Example 2.1. Let $\Sigma = \{a, b\}$ and $W = \{ab, aba, ba, aaa\}$. Then aba contains the enclosed word (clump) ab, but ba is not considered as being enclosed as it ends at the last symbol of aba and, thus, does not require any special attention. The words ab, aba, and aaa each overlap aaa, aba, and ba on the right. For the data sequence

$$x_1, \ldots, x_{20} = bb \underbrace{ab}_{\text{clump 1}} \underbrace{baaababa}_{\text{clump 2}} \underbrace{abab}_{\text{clump 3}} bb \underbrace{ba}_{\text{clump 4}},$$

the two-letter string bb always signals that a clump ends at the end of the last occurring word since bb is not a factor of any word of W. On the other hand, the two-letter factor aa at sequence positions 12 and 13, though a factor of $aaa \in W$, is not a factor of a word that occurs in the sequence, and, thus, when aaa does not occur at sequence position 14, we know that the clump ended at the position of the last occurring word (sequence position 12). The four clumps cover 16 of the 20 sequence positions, all other than the two sets of bb for which neither b is in a clump. The string $x_3, \ldots, x_{16}$ forms a single h-clump for any $h \geq 0$, and $x_3, \ldots, x_{20}$ is a single h-clump for any $h \geq 2$.

The set defined next is very helpful for determining the number and coverage of clumps.

Definition 2.2. The *extension set* W_{ext} is defined by $W_{\text{ext}} \equiv \{u \mid \text{there exist } w^{(i)}, w^{(j)} \in W \text{ with } u = \alpha_i v_{ij} \omega_j, \alpha_i v_{ij} = w^{(i)}, v_{ij}\omega_j = w^{(j)}, |\alpha_i|, |v_{ij}|, |\omega_j| > 0\}$.

Definition 2.2 implies that v_{ij} is both a proper suffix of $w^{(i)}$ and a proper prefix of $w^{(j)}$. The string ω_j extends $w^{(i)}$ to the overlapping word occurrence $w^{(j)}$. Since an h-clump with $h \geq 0$ may also be extended by concatenating a word of W to the right of $w^{(i)}$ with a gap of length no more than h, we define the extension set for this case as $W_{\text{ext}}^h \equiv W_{\text{ext}} \cup (\bigcup_{k=0}^{h} W\Sigma^k W)$, where $\Sigma^0 = \varepsilon$, the empty string, and $W_1 W_2$ denotes the concatenation of patterns W_1 and W_2. For $W = \{ab, aba, ba, aaa\}$,

$$W_{\text{ext}} = \{aba, abab, ababa, abaaa, bab, baba, baaa, aaab, aaaba, aaaa, aaaaa\},$$

and

$$W_{\text{ext}}^0 = W_{\text{ext}} \cup \{abab, ababa, abba, abaaa, abaab, abaaba, ababa, abaaaa, baab, baaba, baba, baaaa, aaaab, aaaaba, aaaba, aaaaaa\}.$$

https://doi.org/10.1239/jap/1324046018 Published online by Cambridge University Press

3. Computation of the distribution of statistics associated with clumps

Deterministic finite automata (DFAs) have the ability to 'recognize' words in longer strings, and have a natural connection to Markov models that make them useful for computing probabilities of various pattern types. This has recently been exploited to develop efficient computation algorithms using various data structures (see [9], [10], [12], and [14]). We compute distributions of clump statistics by associating probabilities with automaton transitions along with updating statistic values on transitions into counting states (words of W or extended words). Our approach more closely resembles that of [14], the main difference being that we set up an automaton for extended words. The algorithm is more straightforward when there are no enclosed clumps.

3.1. When there are no enclosed clumps

We begin by setting up an Aho–Corasick automaton [1] for the words of $W_{\text{ext}} \cup W \cup \Sigma^m$ (throughout the discussion, replace W_{ext} with W^h_{ext} for the case of h-clumps). The automaton can be considered as a directed graph, with automaton states represented as vertices, and edges connecting them. In what follows we use *path* between a starting and ending state in a graph-theoretic sense: the path is the shortest sequence of vertices between those states such that, from each vertex, there is a directed edge to the next vertex in the sequence.

The automaton states are partitioned into basic classes (with further restrictions imposed; see the next paragraph), and the number of states is minimized by finding the *coarsest partition* [18], using an algorithm analogous to the Hopcroft algorithm [6] for obtaining a minimal DFA. The basic classes are Q_{pre}, proper prefixes of words of W; Q_W, words of W themselves (and also all m-tuples ending with a $w^{(j)} \in W$ if $2 \le |w^{(j)}| < m$); Q_{path}, strings along the path between strings of Q_W and Q_{ext} that do not have a word of W as a suffix; $Q_{\text{path,ext}}$, strings along the path between strings of Q_W and Q_{ext} that have a word of W as a suffix; Q_{ext}, strings of W_{ext} that are not already classified as $Q_{\text{path,ext}}$; and Q_m, strings of Σ^m that do not fall into any of the classes above. (That a string in W_{ext} can also be along the path between a string of Q_W and Q_{ext} is borne out by the pattern $W = \{aaaa\}$. In that case $W_{\text{ext}} = \{aaaaa, aaaaaa, aaaaaaa\}$, and strings $aaaaa$ and $aaaaaa$ also lie on the path between $aaaa$ and $aaaaaaa$.)

The 'further restrictions' imposed on the partitioning of automaton states are: strings of the basic classes may not be combined if their suffix of length m is not the same, and counting states may not be combined if they have different updates to clump count or coverage when they are entered. These restrictions are necessary to be able to carry out the computation with the reduced state space after forming the coarsest partition.

The minimal automaton is then processed to determine statistic updates to associate with its edges. The updates are as follows: for $(s_{t-1}, s_t) \in (Q_{\text{pre}} \cup Q_m) \times Q_W$, the clump count increases by 1 and coverage increases by $|w^{(i)}|$ ($w^{(i)} \in W$ is the word that has just occurred); for $(s_{t-1}, s_t) \in [Q_{\text{path}} \times (Q_{\text{ext}} \cup Q_{\text{path,ext}})]$, the clump count remains the same but coverage increases by $|\omega^{(j)}|$ if $g < 0$, and by $|w^{(j)}|$ if $h \ge g \ge 0$, where g is the gap between the last two word occurrences; for $(s_{t-1}, s_t) \in [Q_W \times (Q_{\text{ext}} \cup Q_{\text{path,ext}})] \cup [(Q_{\text{path,ext}} \cup Q_{\text{ext}}) \times Q_{\text{ext}}]$, coverage increases by 1 but the clump count remains the same.

After establishing the updates, any state of Q_{ext} is deleted, with the incoming edge (along with the statistic update information) sent to $w^{(j)}$, the longest word of W that is its suffix. Also, states of length less than m are deleted since they are never entered for $t \ge m$.

The transition probability matrix T for m-tuples is used as input to the algorithm, and the stationary probability vector π for $\tilde{x}_m$ is computed using the equation $\pi T = \pi$, with the additional condition that the entries of π sum to 1. The initial probabilities from π are

https://doi.org/10.1239/jap/1324046018 Published online by Cambridge University Press

considered as being multiplied by $\zeta^{\eta_{\tilde{x}_m}}$ or $\xi^{\gamma_{\tilde{x}_m}}$, where $\eta_{\tilde{x}_m}$ and $\gamma_{\tilde{x}_m}$ denote the number and coverage of clumps in $\tilde{x}_m$. These initial probability polynomials are stored in a vector $\tilde{\pi}$ whose entries are themselves vectors, the coefficients of powers of ζ or ξ (in the end we will obtain coefficients of probability generating functions $\varphi_\eta(\zeta)$ and $\varphi_\gamma(\xi)$ for clump count and coverage). Using an easy modification to our algorithm, we can compute joint probabilities of the number and coverage of clumps.

A square transition matrix Ω (of dimension equal to the number of automaton states) is formed to hold both transition probabilities and statistic updates corresponding to automaton transitions. Let s_τ be the automaton state at time τ, with $\tilde{x}_\tau$ being the suffix of s_τ of length m. The transition probability associated with the transition $s_{t-1} \to s_t$ is exactly the probability from probability matrix T for the transition $\tilde{x}_{t-1} \to \tilde{x}_t$. Probabilities for transitions of automaton states are multiplied by powers of ζ or ξ (the exponent respectively indicating the change to the clump count or coverage). For example, for $(s_{t-1}, s_t) \in (Q_{\text{pre}} \cup Q_m) \times Q_W$, the edge probability is multiplied by ζ to indicate that the clump count increases by 1, or by $\xi^{|w^{(i)}|}$ if the distribution of coverage is sought, to indicate that coverage increases by $|w^{(i)}|$. However, instead of storing polynomials (probabilities multiplied by powers of ζ or ξ) as the entries of Ω, we store vectors of the polynomial coefficients.

If $\mathbf{1}$ denotes a column vector consisting of 1s that is used to sum over automaton states, then $\tilde{\pi}\Omega^{n-m}\mathbf{1}$ gives the probability generating function $\varphi_\eta(\zeta)$ or $\varphi_\gamma(\xi)$, so that the coefficient of ζ^η, for example, is the probability of clump count η. The computation is carried out by first determining Ω^{n-m} through successive doubling of Ω to obtain $\Omega^2, \Omega^4, \Omega^8, \Omega^{16}, \ldots$. These powers of Ω are multiplied to obtain Ω^{n-m}, reducing the number of multiplications needed—the reduction being more dramatic for higher powers. Multiplication of polynomials (the row/column entries of powers of Ω) is carried out through convolution of the vectors holding their coefficients. A MATLAB® program was written to implement the algorithm.

Example 3.1. ($W = \{aa, ab, ba\}$.) Consider first clumps of the pattern $W = \{aa, ab, ba\}$, with alphabet $\Sigma = \{a, b\}$ and $m = 2$. The matrix T is defined to have entries

$$\Pr(aa \mid aa) = \Pr(bb \mid bb) = \tfrac{3}{4} \quad \text{and} \quad \Pr(ba \mid ab) = \Pr(ab \mid ba) = \tfrac{1}{2},$$

with the other entries being implied (and with $\pi = (\pi_{aa}, \pi_{ab}, \pi_{ba}, \pi_{bb}) = (\frac{1}{3}, \frac{1}{6}, \frac{1}{6}, \frac{1}{3})$ being the resulting initial distribution for $\tilde{x}_2$ so that

$$\tilde{\pi} = \left(\tfrac{1}{3}\zeta, \tfrac{1}{6}\zeta, \tfrac{1}{6}\zeta, \tfrac{1}{3}\right)$$

for the number of clumps and

$$\tilde{\pi} = \left(\tfrac{1}{3}\xi^2, \tfrac{1}{6}\xi^2, \tfrac{1}{6}\xi^2, \tfrac{1}{3}\right)$$

for coverage).

The Aho–Corasick automaton for $W_{\text{ext}} = \{aaa, aba, baa, bab, aab\}$ is shown in Figure 1(a). The minimization procedure without applying the restriction for m-tuple suffixes has $Q_{\text{pre}} = \{\varepsilon, a, b\}$, $Q_W = W$, $Q_{\text{ext}} = W_{\text{ext}}$, and $Q_m = \{bb\}$, and gives a minimal automaton with eight states ($\{aaa, aba, baa\}$, $\{aab, bab\}$, and $\{aa, ba\}$ are single states). However, aba cannot be combined with aaa or baa since it has a different 2-tuple as its suffix, and aa and ba may not be combined as well. Actually, since all states of length less than two and all strings of W_{ext} are deleted, the computation can be run on the four 2-tuples (see Figure 1(b)). The automaton used in the computation for 0-clumps is shown in Figure 1(c).

https://doi.org/10.1239/jap/1324046018 Published online by Cambridge University Press

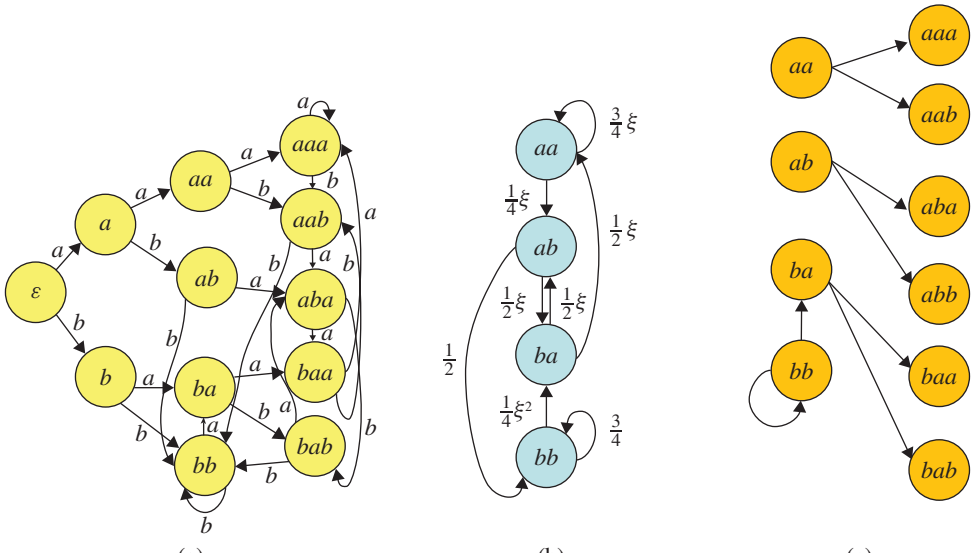

FIGURE 1: DFAs used to compute statistics of clumps/h-clumps of $W = \{aa, ab, ba\}$. (a) Aho–Corasick DFA for $W_{\text{ext}} \cup \Sigma^2$. (b) Final marked DFA for coverage of clumps. (c) Final DFA for 0-clumps, only showing edges from words of W.

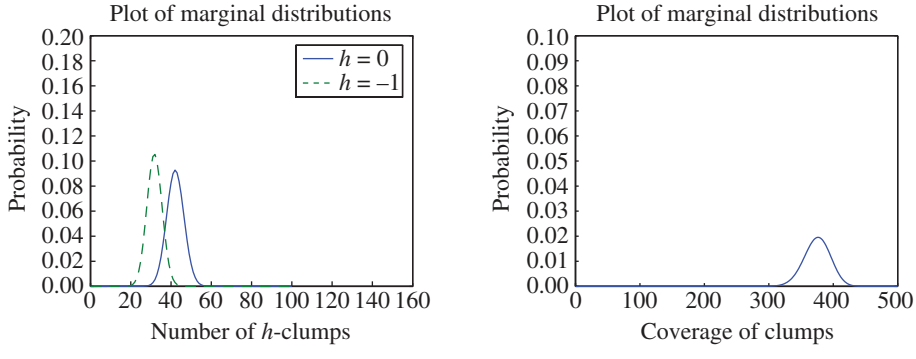

FIGURE 2: Distributions of (a) the h-clump count ($h = -1, 0$) and (b) coverage for the pattern $W = \{aa, ab, ba\}$ with $n = 500$, $m = 2$, and initial and transition probabilities as given in Example 3.1.

Using the MATLAB program, when $n = 5$, the resulting probability generating function for coverage of clumps is

$$\varphi(\xi) = \tfrac{9}{64} + \tfrac{3}{32}\xi^2 + \tfrac{7}{48}\xi^3 + \tfrac{9}{64}\xi^4 + \tfrac{23}{48}\xi^5.$$

The run time was 0.044 358 seconds. Figure 2 contains plots of probabilities for the distributions of η, $\eta^{(0)}$, and γ for $n = 500$. In this case $n - m = 498 = 111110010_2$, indicating that $\Omega^{498} = \Omega^{256}\Omega^{128}\Omega^{64}\Omega^{32}\Omega^{16}\Omega^2$. Thus, Ω^{498} may be obtained using 13 matrix multiplications (as opposed to 497 sequential multiplications by Ω), eight to obtain Ω^2, Ω^4, Ω^8, Ω^{16}, ..., Ω^{256}, another five to multiply the proper matrices. For this value of n, $\mathrm{E}(\eta) = 42.1667$, $\mathrm{var}(\eta) = 18.4772$, $\Pr(\eta \le 30) = 0.002\,716$, $\Pr(\eta \ge 60) = 1.605\,534 \times 10^{-5}$; $\mathrm{E}(\gamma) = 3.748\overline{3} \times 10^2$,

https://doi.org/10.1239/jap/1324046018 Published online by Cambridge University Press

$\Pr(\gamma \le 300) = 3.447\,027 \times 10^{-4}$, and $\Pr(\gamma \ge 430) = 0.001\,518$. The combined computer run time for obtaining probabilities for counts of h-clumps with $h = -1, 0$ was 0.578 520 seconds; the run time for coverage (when $h = -1$) was 0.076 945 seconds.

3.2. When there are enclosed h-clumps

In this case, words of W are still classified as Q_W, whether they are enclosed in another W word or not. Proper prefixes of words of W (that are not themselves a word of W) are subdivided into $Q_{\text{pre,new}}$, prefixes that end in a word of W that indicates a new clump; $Q_{\text{pre,ext}}$, prefixes that end in a word of W to extend a clump; and Q_{pre}, prefixes that do not end with a word of W. Strings along the path between a string of Q_W and W_{ext} that do not fall into any of the classes above, and that end with a word of W to indicate a new clump, are called $Q_{\text{path,new}}$; those that extend a clump are still called $Q_{\text{path,ext}}$. Strings of W_{ext} that do not fall into one of the classes above are called Q_{ext}, and strings of Σ^m that do not fall into any of the classes above are called Q_m. We impose the same restrictions as for the case when there were no enclosed words: we do not allow strings to be combined as a single state in the minimization process unless they have the same updates to clump count and coverage, and the same m-tuple as their suffix.

For transitions $s_{t-1} \to s_t$ with $(s_{t-1}, s_t) \in [(Q_m \cup Q_{\text{pre}}) \times (Q_{\text{pre,new}} \cup Q_W)] \cup (Q_{\text{path}} \times Q_{\text{path,new}}) \cup [(Q_{\text{pre,new}} \cup Q_{\text{pre,ext}}) \times Q_W]$, the edge probability is multiplied by $\zeta^{1-\eta_w}$ for the number of clumps, and by $\xi^{|w|-\gamma_w}$ for coverage, where $w \in W$ is the suffix of s_t that has just occurred, and η_w and γ_w respectively denote the number of clumps and coverage enclosed within w, i.e. occurring before the last symbol of w. For $(s_{t-1}, s_t) \in [(Q_{\text{pre,new}} \cup Q_{\text{pre,ext}}) \times Q_{\text{pre,ext}}] \cup [Q_W \times (Q_{\text{pre,ext}} \cup Q_{\text{ext}})] \cup [(Q_{\text{pre,ext}} \cup Q_{\text{path,ext}} \cup Q_{\text{ext}}) \times (Q_{\text{path,ext}} \cup Q_{\text{ext}})]$, a clump is either formed or extended at time $t-1$ and extended at time t, and the edge probability is multiplied by ξ. When no clump is extended at time $t-1$ but one is extended at time t with the occurrence of the word $w^{(j)}$, i.e. for $(s_{t-1}, s_t) \in (Q_{\text{pre}} \times Q_{\text{pre,ext}}) \cup (Q_{\text{path}} \times Q_{\text{path,ext}}) \cup [(Q_{\text{pre,new}} \cup Q_{\text{pre}} \cup Q_{\text{path,new}} \cup Q_{\text{path}}) \times Q_{\text{ext}}]$, the edge probability is multiplied by $\zeta^{-\eta_{w^j}}$ for the clump count and by $\xi^{|\omega_j|-\gamma_{\omega_j}}$ for coverage if $g < 0$, and by $\zeta^{-\eta_{w^{(j)}}}$ for the clump count and by $\xi^{|\omega^{(j)}|-\gamma_{\omega^{(j)}}}$ for coverage if $h \ge g \ge 0$.

Strings of Q_{ext} may then be deleted, with their incoming transitions (along with the statistic update information) mapped instead to the longest suffix of the string that remains in the automaton. The rest of the computation proceeds as in the case where there were no enclosed clumps.

Example 3.2. ($W = \{ab, bc, ba, abca, bcaba\}$.) Here $\Sigma = \{a, b, c\}$, and we take $m = 1$ ($\eta_{x_1} = \gamma_{x_1} = 0$ for all x_1). The string $abca$ contains the enclosed clump abc ($\eta_{abca} = 1$, $\gamma_{abca} = 3$). Also, $\eta_{bcaba} = 2$ and $\gamma_{bcaba} = 4$. Thus, when $abca$ occurs, the clump count remains the same (abc has already been counted) and the coverage is incremented by 1. On the other hand, when $bcaba$ occurs, the clump count is decreased by 1 (since $bcaba$ is considered as one clump, but two have already been counted) and coverage is incremented by 1. For this example, $W_{\text{ext}} = \{abc, aba, bab, babca, abcab, abcabca, abcaba, bcabab, bcababca\}$. Whereas the Aho–Corasick automaton for $W \cup W_{\text{ext}} \cup \Sigma$ has 23 states (see Figure 3(a)), its coarsest partition has 14, with $\{bab/abcab/bcabab\}$, $\{aba/babca/abcaba/abcabca/bcababca\}$, and $\{babc/abcabc/bcababc\}$ being combined as single states. Figure 3(b) depicts the final automaton that is used for computing the distributions (after deleting states of Q_{ext} and the empty string, which has length less than m).

https://doi.org/10.1239/jap/1324046018 Published online by Cambridge University Press

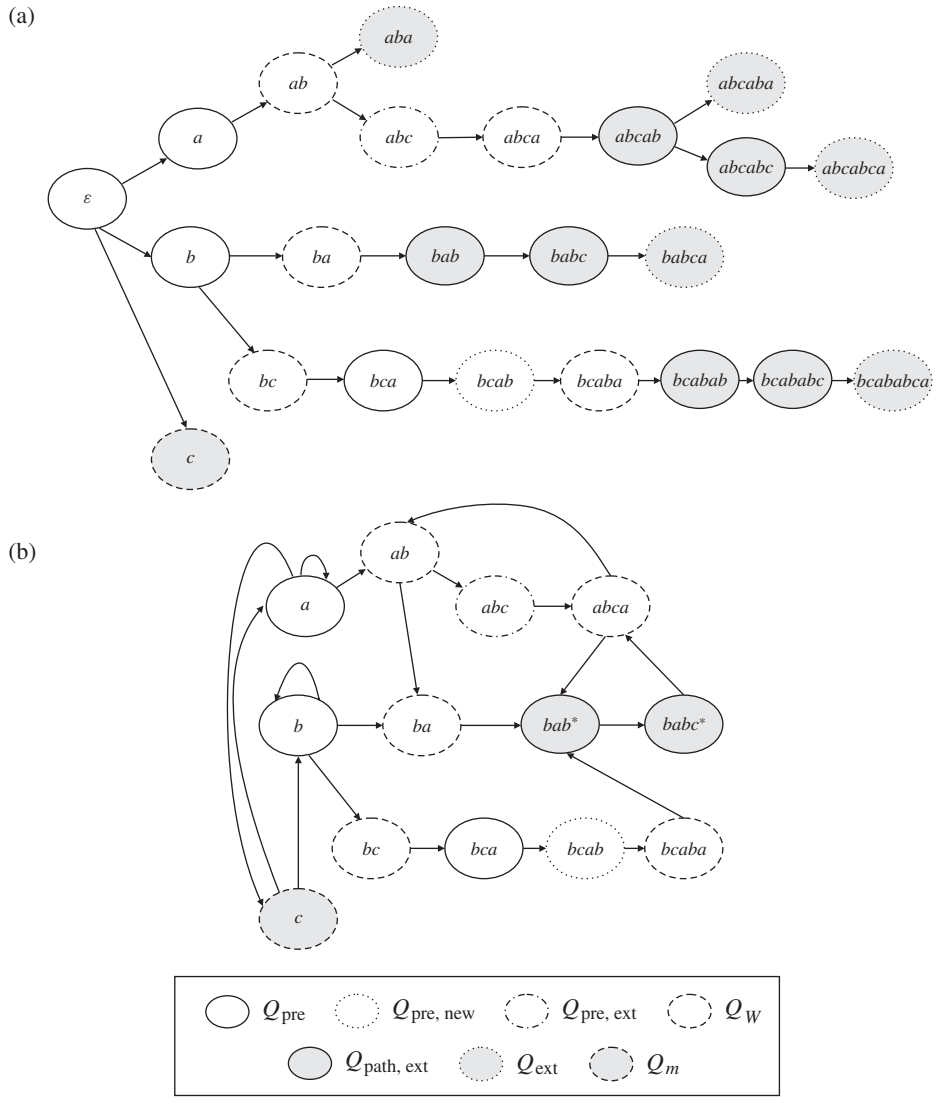

FIGURE 3: DFAs used to compute statistics of clumps of $W = \{ab, bc, ba, abca, bcaba\}$. (a) Aho–Corasick DFA for $W_{\text{ext}} \cup \Sigma^m$. (b) Final DFA. For the sake of clarity, not all automaton transitions are shown, and edge labels are omitted. The state bab^* denotes $\{bab/abcab/bcababab\}$, and $babc^*$ denotes $\{babc/abcabc/bcababc\}$.

The underlying sequence is assumed to be of length $n = 500$ with transition probability matrix

$$T = \begin{pmatrix} 0.5 & 0.25 & 0.25 \\ 0.3 & 0.4 & 0.3 \\ 0.3 & 0.4 & 0.3 \end{pmatrix}$$

(so that the initial distribution is $\pi = (\pi_a, \pi_b, \pi_c) = (\frac{3}{8}, \frac{11}{32}, \frac{9}{32})$). Figure 4 shows the distributions of the clump count and coverage. We report the tail probability $\Pr(\eta \geq 150) = 1.112 \times 10^{-12}$.

https://doi.org/10.1239/jap/1324046018 Published online by Cambridge University Press

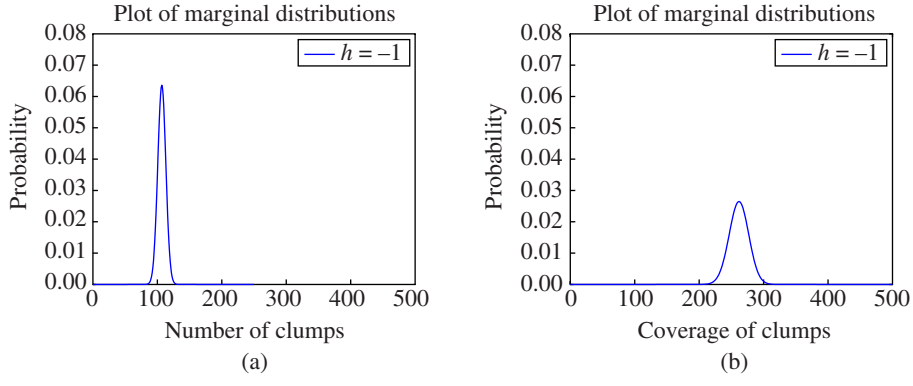

FIGURE 4: Distributions of (a) the clump count and (b) coverage for the pattern $W = \{ab, bc, ba, abca, bcaba\}$ with $n = 500$ and $m = 1$. Initial and transition probabilities are as given in the text.

4. Application to the Chi motif

We now compute distributions of clump statistics for the Chi motif of *H. influenzae*, $W = G\Sigma TGGTGG$, where $\Sigma = \{A, C, G, T\}$. This pattern is important because its presence is needed in processes that prevent a cell from destroying its own DNA each time that it is broken. The pattern is over-represented in *H. influenzae*, occurring 223 times in 215 clumps in the complete genome of 1 830 140 base pairs [8].

For this pattern, $W_{\text{ext}} = \{WTGG, WTGGTGG, W\Sigma TGGTGG\}$. Whereas the Aho–Corasick automaton has 165 states, many of them are combined in the minimal automaton (one reason being that any letter from $\Lambda = \{A, C, T\}$ used as the second symbol gives the same progress into W), leaving only 31 states in the minimal automaton:

$$\{A, C, T, G, GG, GA, GC, GT, GGT, G\Lambda T, G\Sigma TG, WTGGTGG, WTGGTG, WTGGT, G\Sigma TGG, G\Sigma TGGT, G\Sigma TGGTG, W, WG, WT, WA, WC, W\Lambda T, WGT, W\Sigma TG, W\Sigma TGG, W\Sigma TGGT, WTG, W\Sigma TGGTG, WTGG, W\Sigma TGGTGG\}.$$

We assume that the underlying DNA sequence is Markovian with $m = 1$. States $WTGGTGG$ and $W\Sigma TGGTGG$ are deleted since they are in Q_{ext}. For 0-clumps, $W^0_{\text{ext}} = W_{\text{ext}} \cup WW$ and the Aho–Corasick automaton has 273 states, whereas its coarsest partition has only 41, of which only 38 are needed to carry out the computation, an indication of the great savings that can take place by forming the coarsest partition.

We use the maximum likelihood estimates based on transitions of symbols $\{A, C, G, T\}$ in the data (see [15, p. 124]):

$$T = \begin{pmatrix} 0.383 & 0.155 & 0.164 & 0.299 \\ 0.343 & 0.187 & 0.216 & 0.254 \\ 0.269 & 0.264 & 0.197 & 0.269 \\ 0.230 & 0.160 & 0.220 & 0.390 \end{pmatrix}.$$

The initial distribution of x_1 is $(\pi_A, \pi_C, \pi_G, \pi_T) = (0.305, 0.184, 0.198, 0.313)$. The probability of observing 215 or more clumps in 1 830 140 base pairs using these parameters is $1.251\,521\,1 \times 10^{-11}$. Figure 5(a) shows the distribution of the number of both clumps

https://doi.org/10.1239/jap/1324046018 Published online by Cambridge University Press

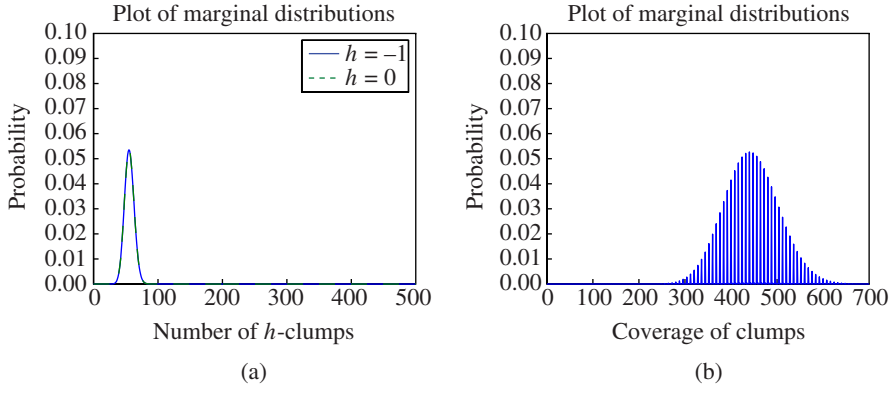

FIGURE 5: Distributions of (a) clmp and 0-clump counts and (b) coverage for the pattern $W = G\Sigma TGGTGG$ ($\Sigma = \{A, C, G, T\}$), with $n = 1.830\,14 \times 10^6$ and $m = 1$. Initial and transition probabilities are as given in the text.

and 0-clumps for the sequence, and Figure 5(b) has the distribution of clump coverage. The combined run time was approximately 26 minutes for the count of h-clumps, $h = -1, 0$.

5. Summary

In this paper an efficient method is given for computing distributions of the number and coverage of clumps (overlapping occurrences of a collection of words) and h-clumps (where gaps of length no more than h are allowed between word occurrences). The underlying sequence is assumed to be Markovian of a general order. We allow words of the collection to be contained in other words, thus facilitating very general applications. The method is applied to computing the statistical significance of observed clump statistics of the Chi motif in a DNA sequence of nearly two million nucleotides, showing that the algorithm is feasible for large data sets.

Acknowledgements

The authors would like to thank an anonymous referee for very helpful comments that greatly improved the paper. This work was supported by the National Science Foundation, under grants DMS-0805577 and DMS-1107084.

References

[1] AHO, A. V. AND CORASICK, M. J. (1975). Efficient string matching: an aid to bibliographic search. *Commun. ACM* **18,** 333–340.
[2] ASTON, J. A. D. AND MARTIN, D. E. K. (2005). Waiting time distributions of competing patterns in higher-order Markovian sequences. *J. Appl. Prob.* **42,** 977–988.
[3] BALAKRISHNAN, N. AND KOUTRAS, M. V. (2002). *Runs and Scans with Applications*. John Wiley, New York.
[4] BENSON, G. (1999). Tandem repeats finder: a program to analyze DNA sequences. *Nucleic Acids Res.* **27,** 573–580.
[5] BIGGINS, J. D. AND CANNINGS, C. (1987). Markov renewal processes, counters and repeated sequences in Markov chains. *Adv. Appl. Prob.* **19,** 521–545.
[6] HOPCROFT, J. (1971). An n log n algorithm for minimizing states in a finite automaton. In *Theory of Machines and Computations*, eds Z. Kohavi and A. Paz, Academic Press, New York, pp. 189–196.
[7] KOSORESOW, A. P. AND HOFMEYR, S. A. (1997). Intrusion detection via system call traces. *IEEE Software* **14,** 35–42.

[8] Ledent, S. and Robin, S. (2005). Checking homogeneity of motifs' distribution in heterogenous sequences. *J. Comput. Biol.* **12,** 672–685.
[9] Lladser, M. E., Betterton, M. D. and Knight, R. (2008). Multiple pattern matching: a Markov chain approach. *J. Math. Biol.* **56,** 51–92.
[10] Marshall, T. and Rahmann, S. (2008). Probabilistic arithmetic automata and their application to pattern matching statistics. In *Combinatorial Pattern Matching* (Lecture Notes Comput. Sci. **5029**), Springer, Berlin, pp. 95–106.
[11] Martin, D. E. K. and Aston, J. A. D. (2008). Waiting time distribution of generalized later patterns. *Comput. Statist. Data Anal.* **52,** 4879–4890.
[12] Nuel, G. (2007). Pattern Markov chains: optimal Markov chain embedding through deterministic finite automata. *J. Appl. Prob.* **45,** 226–243.
[13] Reinert, G., Schbath, S. and Waterman, M. S. (2005). Statistics on words with applications to biological sequences. In *Applied Combinatorics on Words*, eds J. Berstel and D. Perrin, Cambridge University Press, pp. 268–352.
[14] Ribeca, P. and Raineri, E. (2008). Faster exact Markovian probability functions for motif occurrences: a DFA-only approach. *Bioinformatics* **24,** 2839–2848.
[15] Robin, S., Rodolphe, F. and Schbath, S. (2005). *DNA, Words and Models*. Cambridge University Press.
[16] Schbath, S. (1995). Compound Poisson approximation of word counts in DNA sequences. *ESAIM Prob. Statist.* **1,** 1–16.
[17] Stefanov, V. T., Robin, S. and Schbath, S. (2007). Waiting times for clumps of patterns and for structured motifs in random sequences. *Discrete Appl. Math.* **155,** 868–880.
[18] Tewari, A., Srivastava, U. and Gupta, P. (2002). A parallel DFA minimization algorithm. In *High Performance Computing* (Lecture Notes Comput. Sci. **2552**), Springer, Berlin, pp. 34–40.