Analytic Pattern Matching: 
From DNA to Twitter

Wojciech Szpankowski*

Purdue University
W. Lafayette, IN 47907

March 10, 2015

Information Theory, Learning, and Big Data, Berkeley, 2015

*Joint work with Philippe Jacquet
1. Motivations
   - Finding Biological Signals
   - Searching Google
   - Classifying Twitter

2. Pattern Matching Problems
   - Exact String Matching
   - Constrained String Matching
   - Generalized String Matching
   - Subsequence String Matching
   - String Complexity

3. Example of an Analysis:
   Exact String Matching.
Biological world is highly stochastic and inhomogeneous (S. Salzberg).
Motivation – Google & Subsequence Matching

Scholarly articles for tree estimation probability theory
Detection, estimation, and modulation theory - Van Trees - Cited by 9276
... discrete probability distributions with dependence trees - Chow - Cited by 1975
Estimation of failure probability of oil and gas ... - Yuhua - Cited by 163

Estimation theory - Wikipedia, the free encyclopedia
It is also possible for the parameters themselves to have a probability distribution (e.g., Bayesian ...). Numerous fields require the use of estimation theory.

Bayes' theorem - Wikipedia, the free encyclopedia
In probability theory and statistics, Bayes’ theorem (alternatively Bayes’ law ...). The role of Bayes’s theorem is best visualized with tree diagrams, as shown to the right...... "Bayes Estimate/Risk/Solution", "Empirical Bayes", and "Bayes Factor".
"allow users to download an entire movie in one second." I need this [http://t.co/3fbNfKEkah](http://t.co/3fbNfKEkah)

Green energy boss accuses Govt of obstructing renewable energy development [http://t.co/v5Lq2Jx1GQ](http://t.co/v5Lq2Jx1GQ)

Figure 1: Two similar twitter texts have many common words

Figure 2: Twitter's Classification
Outline Update

1. Motivations
2. Pattern Matching Problems
   - Exact String Matching
   - Constrained String Matching
   - Generalized String Matching
   - Subsequence String Matching
3. Example of an Analysis: Exact String Matching.
Pattern Matching

Let $W$ and $T$ be (set of) strings generated over a finite alphabet $\mathcal{A}$.

We call $W$ the pattern and $T$ the text. The text $T$ is of length $n$ and is generated by a probabilistic source.

We shall write

$$T^n_m = T_m \ldots T_n.$$

The pattern $W$ can be a single string

$$W = w_1 \ldots w_m, \ w_i \in \mathcal{A}$$

or a set of strings

$$W = \{W_1, \ldots, W_d\}$$

with $W_i \in \mathcal{A}^{m_i}$ being a set of strings of length $m_i$.

**Basic question:**

how many times $W$ occurs in $T$ (or how long to wait until $W$ occurs in $T$).

Define

$$O_n(W) = \# \{ i : T^i_{i-m+1} = W, \ m \leq i \leq n \}.$$
(Exact) String Matching

In the exact string matching the pattern \( \mathcal{W} = w_1 \ldots w_m \) is a given string (i.e., consecutive sequence of symbols).

Generalized String Matching

In the generalized pattern matching a set of patterns (rather than a single pattern) is given, that is,

\[
\mathcal{W} = (\mathcal{W}_0, \mathcal{W}_1, \ldots, \mathcal{W}_d), \quad \mathcal{W}_i \in \mathcal{A}^{m_i}
\]

where \( \mathcal{W}_i \) itself for \( i \geq 1 \) is a subset of \( \mathcal{A}^{m_i} \) (i.e., a set of words of a given length \( m_i \)).

The set \( \mathcal{W}_0 \) is called the forbidden set.

Three cases to be considered:

\( \mathcal{W}_0 = \emptyset \) — one is interested in the number of patterns from \( \mathcal{W} \) occurring in the text.

\( \mathcal{W}_0 \neq \emptyset \) — we study the number of \( \mathcal{W}_i, i \geq 1 \) pattern occurrences under the condition that no pattern from \( \mathcal{W}_0 \) occurs in the text.

\( \mathcal{W}_i = \emptyset, i \geq 1, \mathcal{W}_0 \neq \emptyset \) — restricted pattern matching.
Pattern Matching Problems

Hidden Words or Subsequence Pattern Matching

In this case we search in text for a subsequence \( \mathcal{W} = w_1 \ldots w_m \) rather than a string, that is, we look for indices \( 1 \leq i_1 < i_2 < \cdots < i_m \leq n \) such that

\[
T_{i_1} = w_1, \quad T_{i_2} = w_2, \ldots, \quad T_{i_m} = w_m.
\]

We also say that the word \( \mathcal{W} \) is “hidden” in the text.

For example:

\[
\mathcal{W} = \text{date} \\
T = \text{hidden pattern}
\]

occurs four times as a subsequence in the text as hidden pattern but not even once as a string.

Self-Repetitive Pattern Matching

In this case the pattern \( \mathcal{W} \) is part of the text:

\[
\mathcal{W} = T_1^m.
\]

We may ask when the first \( m \) symbols of the text will occur again. This is important in Lempel-Ziv like compression algorithms.
How do you distinguish a cat from a dog by their DNA? Did Shakespeare really write all of his plays?

Pattern matching techniques can offer answers to these questions and to many others, from molecular biology, to telecommunications, to classifying Twitter content. This book for researchers and graduate students demonstrates the probabilistic approach to pattern matching, which predicts the performance of pattern matching algorithms with very high precision using analytic combinatorics and analytic information theory. Part I compiles known results of pattern matching problems via analytic methods. Part II focuses on applications to various data structures on words, such as digital trees, suffix trees, string complexity and string-based data compression. The authors use results and techniques from Part I and also introduce new methodology such as the Mellin transform and analytic depoissonization. More than 100 end-of-chapter problems help the reader to make the link between theory and practice.

Philippe Jacquet is a research director at INRIA, a major public research lab in Computer Science in France. He has been a major contributor to the Internet OLSR protocol for mobile networks. His research interests involve information theory, probability theory, quantum telecommunication, protocol design, performance evaluation and optimization, and the analysis of algorithms. Since 2012 he has been with Alcatel-Lucent Bell Labs as head of the department of Mathematics of Dynamic Networks and Information. Jacquet is a member of the prestigious French Corps des Mines, known for excellence in French industry, with the rank of “Ingénieur Général”. He is also a member of ACM and IEEE.

Wojciech Szpankowski is Saul Rosen Professor of Computer Science and (by courtesy) Electrical and Computer Engineering at Purdue University, where he teaches and conducts research in analysis of Algorithms, information theory, bioinformatics, analytic combinatorics, random structures, and stability problems of distributed systems. In 2008 he launched the interdisciplinary Institute for Science of Information. Szpankowski is a Fellow of IEEE and an Honorary Fellow. He received the Humboldt Research Award in 2002.
Chapter 1: Probabilistic Models

Chapter 2: Exact String Matching (DNA Applications)
2.1 Formulation of the problem
2.2 Language representation
2.3 Generating functions
2.4 Moments
2.5 Limit laws

Chapter 3: Constrained Exact String Matching (Constrained Coding)
3.1 Enumeration of (d, k) sequences
3.7 Application: Significant signals in neural data

Chapter 4: Generalized String Matching (Biological Applications)
4.1 String matching over a reduced set
4.2 Generalized string matching via automata
4.3 Generalized string matching via a language approach

Chapter 5: Subsequence String Matching (Google Applications)
5.1 Problem formulation
5.2 Mean and variance analysis
5.3 Autocorrelation polynomial revisited
5.4 Central limit laws
5.5 Limit laws for fully constrained pattern
5.6 Generalized subsequence problem
Chapter 6: Algorithms and Data Structures
6.1 Tries
6.2 Suffix trees
6.3 Lempel-Ziv’77 scheme
6.4 Digital search tree
6.5 Parsing trees and Lempel-Ziv’78 algorithm

Chapter 7: Digital Trees

Chapter 8: Suffix Trees & Lempel-Ziv’77
8.1 Random tries resemble suffix trees
8.2 Size of suffix tree
8.3 Lempel-Ziv’77

Chapter 9: Lempel-Ziv’78 Compression Algorithm

Chapter 10: String Complexity (Twitter Applications)
10.1 Introduction to string complexity
10.2 Analysis of string self-complexity
10.3 Analysis of the joint complexity
10.4 Average joint complexity for identical sources
10.5 Average joint complexity for non identical sources
10.6 Joint complexity via suffix trees
10.7 Conclusion and applications
1. Motivations
2. Pattern Matching Problems
3. Example of an Analysis:
   Exact String Matching
Analysis: Exact String Matching

In the **exact string matching** the pattern \( \mathcal{W} = w_1 \ldots w_m \) is a given string and one searches for its occurrences in a random text \( T_1^n \).

**Memoryless Source**: The text is a realization of an independently, identically distributed sequence of random variables such that a symbol \( s \in \mathcal{A} \) occurs with probability \( P(s) \).

Extensions to **Markovian Source** are relatively easy.

**Objective**: probabilistic laws for

\[
O_n(\mathcal{W}) = \# \{ i : T_{i-m+1}^i = \mathcal{W}, \ m \leq i \leq n \}.
\]

**Tools**: Symbolic calculus and analytic tools of languages:

Language \( \mathcal{L} \) is a collection of words satisfying some properties. Generating function \( L(z) \) of language \( \mathcal{L} \) is defined as

\[
L(z) = \sum_{u \in \mathcal{L}} P(u) z^{|u|}
\]

where \( P(w) \) is the stationary probability \( u \) occurrence, \(|u|\) is the length of \( w \).
Given a pattern $\mathcal{W}$, we define the autocorrelation set $\mathcal{S}$ as:

$$\mathcal{S} = \{w_{k+1}^m : w_1^k = w_{m-k+1}^m\}, \; w_1^k = w_{m-k+1}^m$$

and $\mathcal{WW}$ is the set of positions $k$ satisfying $w_1^k = w_{m-k+1}^m$.

The generating function of $\mathcal{S}$ is $S(z)$ known also as the autocorrelation polynomial.

$$S(z) = \sum_{k \in \mathcal{WW}} P(w_{k+1}^m)z^{m-k}.$$ 

**Example:** Let $\mathcal{W} = bab$ over the alphabet $\mathcal{A} = \{a, b\}$.

$$\mathcal{WW} = \{1, 3\} \quad \text{and} \quad \mathcal{S} = \{\epsilon, ab\},$$

where $\epsilon$ is the empty word, since

$$b \quad a \quad b$$

$$b \quad a \quad b$$

For the unbiased memoryless source: $S(z) = 1 + P(ab)z^2 = 1 + \frac{z^2}{4}$. 
Define $\mathcal{T}_r$ as set of words containing exactly $r \geq 1$ occurrences of $\mathcal{W}$:

$$\mathcal{T}_r = \mathcal{R} \cdot \mathcal{M}^{r-1} \cdot \mathcal{U}.$$ 

which can be illustrated as

(i) We define $\mathcal{R}$ as the set of words containing only one occurrence of $\mathcal{W}$, located at the right end. For example, for $\mathcal{W} = aba$, we have $ccaba \in \mathcal{R}$.

(ii) We also define $\mathcal{U}$ as

$$\mathcal{U} = \{u : \mathcal{W} \cdot u \in \mathcal{T}_1\}$$

that is, a word $u \in \mathcal{U}$ if $\mathcal{W} \cdot u$ has exactly one occurrence of $\mathcal{W}$ at the left end of $\mathcal{W} \cdot u$,

$$bba \in \mathcal{U}, \quad ba \notin \mathcal{U}.$$ 

(iii) Let $\mathcal{M}$ be the language:

$$\mathcal{M} = \{u : \mathcal{W} \cdot u \in \mathcal{T}_2 \text{ and } \mathcal{W} \text{ occurs at the right of } \mathcal{W} \cdot u\},$$

that is, $\mathcal{M}$ is a language such that $\mathcal{W} \mathcal{M}$ has exactly two occurrences of $\mathcal{W}$ at the left and right end of a word from $\mathcal{M}$ (e.g., $ba \in \mathcal{M}$ since $ababa$).
Lemma 1. (i) The languages $M$, $U$ and $R$ satisfy:

\[ \bigcup_{k \geq 1} M^k = A^* \cdot W + S - \{\epsilon\}, \]

\[ U \cdot A = M + U - \{\epsilon\}, \quad W \cdot M = A \cdot R - (R - W), \]

where $A^*$ is the set of all words.

(ii) The generating functions associated with languages $M$, $U$ and $R$ satisfy

\[ \frac{1}{1 - M(z)} = S_W(z) + P(W) \frac{z^m}{1 - z}, \]

\[ U_W(z) = \frac{M(z) - 1}{z - 1}, \quad R(z) = P(W) z^m \cdot U_W(z). \]

Theorem 1. The generating functions $T_r(z) = \sum_{n \geq 0} \Pr\{O_n(W) = r\} z^n$ and $T(z, u) = \sum_{r=1}^{\infty} T_r(z) u^r$ satisfy

\[ T_r(z) = R(z) M_W^{r-1}(z) U_W(z), \quad r \geq 1 \]

\[ T(z, u) = R(z) \frac{u}{1 - u M(z)} U_W(z). \]
**Main Results: Asymptotics**

**Theorem 2.** (i) **Moments.** The expectation satisfies, for \( n \geq m \):

\[
\mathbb{E}[O_n(W)] = P(W)(n - m + 1),
\]

while the variance is

\[
\text{Var}[O_n(W)] = nc_1 + c_2
\]

with

\[
c_1 = P(W)(2S(1) - 1 - (2m - 1)P(W),
\]
\[
c_2 = P(W)((m - 1)(3m - 1)P(W) - (m - 1)(2S(1) - 1) - 2S'(1)).
\]

(ii) **Case** \( r = O(1) \). Let \( \rho_W \) be the smallest root of

\[
D_W(z) = (1 - z)S_W(z) + z^m P(W) = 0.
\]

Then

\[
\Pr\{O_n(W) = r\} \sim \sum_{j=1}^{r+1} (-1)^j a_j \binom{n}{j} \rho_W^{-(n+j)}
\]

where

\[
a_{r+1} = \frac{\rho_W^m P(W)(\rho_W - 1)^{r-1}}{(D'_W(\rho_W))^{r+1}},
\]

and the remaining coefficients can be easily computed, too.
(iii) **CLT**: Case $r = EO_n + x\sqrt{\text{Var}O_n}$ for $x = O(1)$. Then:

$$
\Pr\{O_n(\mathcal{W}) = r\} = \frac{1}{\sqrt{2\pi c_1 n}} e^{-\frac{1}{2}x^2} \left(1 + O\left(\frac{1}{\sqrt{n}}\right)\right).
$$

(iv) **Large Deviations**: Case $r = (1 + \delta)EO_n$. Let $a = (1 + \delta)P(\mathcal{W})$ with $\delta \neq 0$. For complex $t$, define $\rho(t)$ to be the root of

$$
1 - e^t MW(e^\rho) = 0,
$$

while $\omega_a$ and $\sigma_a$ are defined as

$$
-\rho'(\omega_a) = a,
\quad
-\rho''(\omega_a) = \sigma_a^2.
$$

Then

$$
\Pr\{O_n(\mathcal{W}) \sim (1 + \delta)EO_n\} = \frac{e^{-(n-m+1)I(a) + \delta_a}}{\sigma_a \sqrt{2\pi(n - m + 1)}}
$$

where $I(a) = a\omega_a + \rho(\omega_a)$ and $\delta_a$ is a constant.
Denise and Regnier (2002) observed that in biological sequence whenever a word is overrepresented, then its subwords are also overrepresented. For example, if $W_1 = \text{AATAAA}$, then

$$W_2 = \text{ATAAN}$$

is also overrepresented. Overrepresented subwords are called artifact, and it is important to disregard automatically noise created by artifacts.

**New Approach:**

Once a dominating signal has been detected, we look for a weaker signal by comparing the number of observed occurrences of patterns to the conditional expectations not the regular expectations.

To solve this harder question one needs a new approach thru Generalized Pattern Matching discussed in Chapter 4. Thea, as in Denise and Regnier (2002) we find

$$E[O_n(W_2)|O_n(W_1) = k] \sim \alpha n.$$ 

When $W_1$ is overrepresented the constant $\alpha$ differs significantly from $E[O_n(W_2)]$. 
Beaudoin et al. (2000) studied several variants of the well-known AAUAAA polyadenylation signal in mRNA of human genes. To avoid artifacts, Beaudoin et al. cancelled all sequences where the overrepresented hexamer was found.

Using our approach, Denise and Regnier (2002) discovered/eliminated all artifacts and found new signals in a much simpler and reliable way.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>AAUAAA</td>
<td>3456</td>
<td>1</td>
<td>363.16</td>
<td>167.03</td>
<td>1</td>
<td>1678.53</td>
<td>1.04</td>
<td>1300</td>
</tr>
<tr>
<td>AAAUAA</td>
<td>1721</td>
<td>2</td>
<td>363.16</td>
<td>71.25</td>
<td>2</td>
<td>1311.03</td>
<td>6.05</td>
<td>404</td>
</tr>
<tr>
<td>AUAAAA</td>
<td>1530</td>
<td>3</td>
<td>363.16</td>
<td>61.23</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UUUUUU</td>
<td>1105</td>
<td>4</td>
<td>416.36</td>
<td>33.75</td>
<td>8</td>
<td>373.30</td>
<td>37.87</td>
<td>2</td>
</tr>
<tr>
<td>AUAAAAU</td>
<td>1043</td>
<td>5</td>
<td>373.23</td>
<td>34.67</td>
<td>6</td>
<td>1529.15</td>
<td>12.43</td>
<td>4078</td>
</tr>
<tr>
<td>AAAAUA</td>
<td>1019</td>
<td>6</td>
<td>363.16</td>
<td>34.41</td>
<td>7</td>
<td>848.76</td>
<td>5.84</td>
<td>420</td>
</tr>
<tr>
<td>UAAAAU</td>
<td>1017</td>
<td>7</td>
<td>373.23</td>
<td>33.32</td>
<td>9</td>
<td>780.18</td>
<td>8.48</td>
<td>211</td>
</tr>
<tr>
<td>AUUAAA</td>
<td>1013</td>
<td>1</td>
<td>373.23</td>
<td>33.12</td>
<td>10</td>
<td>385.85</td>
<td>31.93</td>
<td>3</td>
</tr>
<tr>
<td>AUAAAG</td>
<td>972</td>
<td>9</td>
<td>184.27</td>
<td>58.03</td>
<td>4</td>
<td>593.90</td>
<td>15.51</td>
<td>34</td>
</tr>
<tr>
<td>UAAUAA</td>
<td>922</td>
<td>10</td>
<td>373.23</td>
<td>28.41</td>
<td>13</td>
<td>1233.24</td>
<td>–8.86</td>
<td>4034</td>
</tr>
<tr>
<td>UAAAAA</td>
<td>922</td>
<td>11</td>
<td>363.16</td>
<td>29.32</td>
<td>12</td>
<td>922.67</td>
<td>9.79</td>
<td>155</td>
</tr>
<tr>
<td>UUAAAA</td>
<td>863</td>
<td>12</td>
<td>373.23</td>
<td>25.35</td>
<td>15</td>
<td>374.81</td>
<td>25.21</td>
<td>4</td>
</tr>
<tr>
<td>CAUAAA</td>
<td>847</td>
<td>13</td>
<td>185.59</td>
<td>48.55</td>
<td>5</td>
<td>613.24</td>
<td>9.44</td>
<td>167</td>
</tr>
<tr>
<td>AAAAUA</td>
<td>841</td>
<td>14</td>
<td>353.37</td>
<td>25.94</td>
<td>14</td>
<td>496.38</td>
<td>15.47</td>
<td>36</td>
</tr>
<tr>
<td>UAAUA</td>
<td>805</td>
<td>15</td>
<td>373.23</td>
<td>22.35</td>
<td>21</td>
<td>1143.73</td>
<td>–10.02</td>
<td>4068</td>
</tr>
</tbody>
</table>
THANK YOU