Joint work with: Péter Burcsi, Gabriele Fici, Frank Ruskey, Joe Sawada

A binary word is called prefix normal if no substring has more 1s than the prefix of the same length. For example, 1101010110 is prefix normal but 1100110110 is not, because the substring 1101 has too many 1s. These words are the sequences of the first differences of the function $F(w, k) = \max\{d(u) \mid u \text{ is a substring of } w \text{ of length } k\}$, where $d(u)$ denotes the number of 1s in the binary word $u$. For example, for the word $w = 1100110110$, we get:

<table>
<thead>
<tr>
<th>$k$</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>$F(w, k)$</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>6</td>
<td>6</td>
</tr>
</tbody>
</table>

The sequence of the first differences is $w' = 1101100110 =: \text{PNF}(w)$. We call this (necessarily binary) word the prefix normal form of $w$. It is prefix normal by construction, and it is the only prefix normal word in the equivalence class of $w$ w.r.t. the equivalence $w \equiv v$ iff $F(w, \cdot) = F(v, \cdot)$.

Prefix normal words are motivated by Binary Jumbled Pattern Matching (BJPM): Given a binary string $T$ and a query $(x, y)$, does $T$ contain a substring with $x$ zeros and $y$ ones? BJPM can be solved by a linear scan of $T$ in time $O(n)$. For the indexing variant (IBJPM), define $f(w, k)$ as the minimum number of 1s in a substring of $w$ of length $k$. Then the answer to query $(x, y)$ is YES if and only if $f(T, x + y) \leq y \leq F(T, x + y)$. Thus the problem can be solved by storing $F(w, \cdot)$ and $f(w, \cdot)$ in $O(n)$ space; this is a linear size index, with $O(1)$ query time. The current fastest computation of these functions $F$ and $f$ is given by Chan and Lewenstein, in $O(n^{1.89})$ time [4].

1. Expected critical prefix length of a random prefix normal word of length $n$.

Let the critical prefix of a binary word be defined as the sum of the lengths of the first run of 1s (possibly empty) plus the first run of 0s, and $cr(w)$ be the length of $w$’s critical prefix. E.g. $cr(1110001010) = 6$, $cr(0001100101) = 3$, $cr(1110000000) = 10$, $cr(1^n) = cr(0^n) = n$. 


Conjecture: The expected critical prefix length of a prefix normal word of length $n$ is $O(\log n)$ [2].

We can prove that $\text{Exp}(\text{cr}(w)) < 3$ if taken over all words of length $n$ (for infinite words $w$, it is exactly 3). We can also prove that $\text{Exp}(\text{PNF}(w)) = O(\log n)$, taken over all binary words of length $n$. The paper [5] contains a table with numbers of prefix normal words for $n = 32$ and each combination of $s, t$, for $s = 1,...,7$ and $1 \leq t \leq n$.

The current best result is that $\text{Exp}(\text{cr}(w)) = O(\sqrt{n \log n})$ for a randomly chosen prefix normal word of length $n$ [7].

2. Equivalence class sizes.

Some equivalence classes are singletons (e.g. $1^n, 0^n, 1001, . . .$; this implies that the word is a palindrome, since $F(w, \cdot) = F(w^{rev}, \cdot)$ always), some are much larger. The OEIS sequence number A238110 [6] lists the size of the largest equivalence class for $n$ up to 50. This question is the same as asking how many distinct words can have the same function $F$.

3. Enumeration of prefix normal words.

Let $p_{nw}(n)$ denote the number of prefix normal words of length $n$. It is easy to see that $p_{nw}$ grows exponentially. No closed form is known for $p_{nw}(n)$; OEIS sequence number A194850 [6] lists $p_{nw}(n)$ up to $n = 50$. We have generating functions for some (few) subsets, but not for $p_{nw}(n)$. Asymptotic bounds exist, which seem to imply $p_{nw}(n) = 2^n - \Theta(\log^2 n)$ [3].

The question can be rephrased as: How many different functions $F$ can exist, where a necessary and sufficient condition for a 0-1 step function $F$ to be the $F(w, \cdot)$ of some word $w$ is that for all $i < j$, $F(i+j) \leq F(i) + F(j)$.

4. Testing.

The best algorithm to decide whether a string $w$ is prefix normal is: Compute $\text{PNF}(w)$; $w$ is prefix normal iff $\text{PNF}(w) = w$. The fastest algorithm for doing this is given in [4]. However, it is not clear that recognition is as hard as computing the $F$-function.

5. Which prefix normal forms w.r.t. 1 can be combined with which prefix normal forms w.r.t. 0?

Define $F_0(w, \cdot)$ and $\text{PNF}_0(w)$ analogously to above, but w.r.t. 0 instead of 1. (For constructing $\text{PNF}_0(w)$, we put a 0 when $F_0$ increases, and a 1 otherwise.) Then the two prefix normal forms of $w$ encode the index for BJPM. These can be used to answer BJPM queries as follows:

$$(x, y) \text{ is a YES-query } \iff \text{rank}_1(\text{PNF}_0(w), x+y) \leq y \leq \text{rank}_1(\text{PNF}_1(w), x+y).$$

Prefix normal words w.r.t. 0 are defined analogously to prefix normal words w.r.t. 1. Given $w$, a prefix normal word w.r.t. 1, and $w'$, a prefix normal
word w.r.t. 0, we call w and w′ compatible if there exists a binary word v s.t. \(w = \text{PNF}_1(v)\) and \(w' = \text{PNF}_0(v)\).

The open problem is: Which prefix normal words w.r.t. 1 are compatible with which prefix normal words w.r.t. 0?

6. **How big are the Parikh-equivalence classes?**

Another equivalence relation is given by: \(w\) Parikh-equivalent to \(v\) iff \(\text{PNF}_1(w) = \text{PNF}_1(v)\) and \(\text{PNF}_0(w) = \text{PNF}_0(v)\). Note that this holds iff the Parikh sets of \(w\) and \(v\) are the same, where the Parikh set of a string is the set of Parikh vectors of its substrings. How big are these equivalence classes? I.e. how many different strings can have a same Parikh set?

Similar results about the multiset (not set) of Parikh vectors of substrings can be found in [1].

References


