Periodicity in Rectangular Arrays

UWaterloo Algorithms & Complexity Seminar

Taylor J. Smith

Joint work with Jeffrey Shallit

David R. Cheriton School of Computer Science
University of Waterloo
Waterloo, Ontario, Canada

April 20, 2016

Table of Contents

Table of Contents

Background

- The properties of primitivity and periodicity are well-studied in the field of combinatorics on words.
- From these properties, we get many useful applications (e.g. pattern matching).
- Most of the time, we consider primitivity and periodicity only in one dimension.
- What happens to these properties if we introduce a second dimension?

Table of Contents

Definitions

- A nonempty word z is **primitive** if it cannot be written in the form $z = w^i$ for some word w and some integer $i \geq 2$.
- If z is formed by repetitions of some smaller word w, then z is **periodic**.
- Given a nonempty word z, the shortest word w such that $z = w^j$ for some integer $j \geq 1$ is the **primitive root** of z.

Definitions

- A nonempty word z is **primitive** if it cannot be written in the form $z = w^i$ for some word w and some integer $i \geq 2$.
- If z is formed by repetitions of some smaller word w, then z is **periodic**.
- Given a nonempty word z, the shortest word w such that $z = w^j$ for some integer $j \geq 1$ is the **primitive root** of z.

Example

The word $z_1 = \texttt{door}$ is primitive. The primitive root of z_1 is $w_1 = \texttt{door}$ with $j = 1$.

Example

The word $z_2 = \texttt{dodo}$ is periodic. The primitive root of z_2 is $w_2 = \texttt{do}$ with $j = 2$.

Lyndon-Schützenberger Theorem

- The **Lyndon-Schützenberger theorem** defines a set of conditions for when the concatenation of two words x and y commutes; that is, when $xy = yx$.
- This theorem is one of the most well-known results in the field of combinatorics on words. (For a proof, see the paper by Lyndon and Schützenberger.)

Lyndon-Schützenberger Theorem

Theorem (1D Lyndon-Schützenberger Theorem)

Let $x, y \in \Sigma^+$. Then the following three conditions are equivalent:

1. $xy = yx$;
2. *There exist $z \in \Sigma^+$ and integers $k, l > 0$ such that $x = z^k$ and $y = z^l$;*
3. *There exist integers $i, j > 0$ such that $x^i = y^j$.*

UNIVERSITY OF WATERLOO

Theorem (1D Lyndon-Schützenberger Theorem)

*Let $x, y \in \Sigma^+$. Then the following **five** conditions are equivalent:*

1. $xy = yx$;
2. *There exist $z \in \Sigma^+$ and integers $k, l > 0$ such that $x = z^k$ and $y = z^l$;*
3. *There exist integers $i, j > 0$ such that $x^i = y^j$;*
4. ***There exist integers $r, s > 0$ such that $x^r y^s = y^s x^r$;***
5. $\boldsymbol{x\{x, y\}^* \cap y\{x, y\}^* \neq \emptyset}$.

UNIVERSITY OF WATERLOO

3. There exist integers $i, j > 0$ such that $x^i = y^j$.

$\Downarrow$

4. There exist integers $r, s > 0$ such that $x^r y^s = y^s x^r$.

Proof.
If $x^i = y^j$, then comparing prefixes and suffixes reveals that $x^i y^j = y^j x^i$.
Take $r = i$ and $s = j$ to get $x^r y^s = y^s x^r$. □

4. There exist integers $r, s > 0$ such that $x^r y^s = y^s x^r$.

$\Downarrow$

5. $x\{x, y\}^* \cap y\{x, y\}^* \neq \emptyset$.

Proof.
Let $z = x^r y^s$. Then $z \in x\{x, y\}^*$.
By condition 4, we know that $z = y^s x^r$, so $z \in y\{x, y\}^*$.
Therefore, $x\{x, y\}^* \cap y\{x, y\}^* \neq \emptyset$. □

5. $x\{x,y\}^* \cap y\{x,y\}^* \neq \emptyset$.

$\Downarrow$

1. $xy = yx$.

Proof.

By induction on $|xy|$.

- Both the base case ($|xy| = 2$) and the case where $|x| = |y|$ are trivial.
- Without loss of generality, assume $|x| < |y|$.
  Let z be as before. Since $z \in x\{x,y\}^*$ and $z \in y\{x,y\}^*$ by condition 5, we know x is a proper prefix of y.
  Let $y = xw$. Then z has the prefixes xx and xw, so $x^{-1}z \in x\{x,w\}^*$ and $x^{-1}z \in w\{x,w\}^*$. Thus, $x\{x,w\}^* \cap w\{x,w\}^* \neq \emptyset$.
  By induction, condition 1 holds for x and w, so $xw = wx$ and therefore $yx = (xw)x = x(wx) = xy$. □

Table of Contents

Definitions

- $\Sigma^{m \times n}$ is the set of all $m \times n$ rectangular arrays M of elements chosen from Σ.
- $M[0, 0]$ is the upper-left element of M, and $M[i..j, k..l]$ is the rectangular subarray consisting of rows i through j and columns k through l of M.
- If $M \in \Sigma^{m \times n}$, then $M^{p \times q}$ is the $pm \times qn$ rectangular array constructed by repeating M in p rows and q columns.

Definitions

- $\Sigma^{m\times n}$ is the set of all $m \times n$ rectangular arrays M of elements chosen from Σ.
- $M[0,0]$ is the upper-left element of M, and $M[i..j, k..l]$ is the rectangular subarray consisting of rows i through j and columns k through l of M.
- If $M \in \Sigma^{m\times n}$, then $M^{p\times q}$ is the $pm \times qn$ rectangular array constructed by repeating M in p rows and q columns.

Example

If $M = \begin{bmatrix} \mathtt{a} & \mathtt{b} \\ \mathtt{c} & \mathtt{d} \end{bmatrix}$, then $M^{2\times 3} = \begin{bmatrix} \mathtt{a} & \mathtt{b} & \mathtt{a} & \mathtt{b} & \mathtt{a} & \mathtt{b} \\ \mathtt{c} & \mathtt{d} & \mathtt{c} & \mathtt{d} & \mathtt{c} & \mathtt{d} \\ \mathtt{a} & \mathtt{b} & \mathtt{a} & \mathtt{b} & \mathtt{a} & \mathtt{b} \\ \mathtt{c} & \mathtt{d} & \mathtt{c} & \mathtt{d} & \mathtt{c} & \mathtt{d} \end{bmatrix}$.

Definitions

- An array M is **primitive** if the equation $M = A^{p \times q}$ for some array A and some integers $p, q \geq 1$ implies $p = 1$ and $q = 1$.
- Given an array M, we can write it in the form $M = A^{p \times q}$ for some **primitive root array** A and some integers $p, q \geq 1$.

- An array M is **primitive** if the equation $M = A^{p \times q}$ for some array A and some integers $p, q \geq 1$ implies $p = 1$ and $q = 1$.
- Given an array M, we can write it in the form $M = A^{p \times q}$ for some **primitive root array** A and some integers $p, q \geq 1$.

Example

The array $M_1 = \begin{bmatrix} 1 & 2 \\ 2 & 1 \end{bmatrix}$ is primitive.

Example

The array $M_2 = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$ is not primitive, since we can construct M_2 by taking $A = \begin{bmatrix} 1 \end{bmatrix}$, $p = 2$, and $q = 2$.

Definitions

UNIVERSITY OF WATERLOO

- Given two arrays A and B, we can concatenate these arrays, but we must insist on a matching of dimension.
- If A is $m \times n_1$ and B is $m \times n_2$, then $A \oplus B$ is the $m \times (n_1 + n_2)$ array obtained by placing B to the right of A.
- If A is $m_1 \times n$ and B is $m_2 \times n$, then $A \ominus B$ is the $(m_1 + m_2) \times n$ array obtained by placing B beneath A.

Definitions

- Given two arrays A and B, we can concatenate these arrays, but we must insist on a matching of dimension.
- If A is $m \times n_1$ and B is $m \times n_2$, then $A \oplus B$ is the $m \times (n_1 + n_2)$ array obtained by placing B to the right of A.
- If A is $m_1 \times n$ and B is $m_2 \times n$, then $A \ominus B$ is the $(m_1 + m_2) \times n$ array obtained by placing B beneath A.

Example

If $A_1 = \begin{bmatrix} \texttt{a} & \texttt{b} \end{bmatrix}$ and $B_1 = \begin{bmatrix} \texttt{c} & \texttt{d} \end{bmatrix}$, then $A_1 \ominus B_1 = \begin{bmatrix} \texttt{a} & \texttt{b} \\ \texttt{c} & \texttt{d} \end{bmatrix}$.

Example

If $A_2 = \begin{bmatrix} \texttt{a} & \texttt{b} \\ \texttt{d} & \texttt{e} \end{bmatrix}$ and $B_2 = \begin{bmatrix} \texttt{c} \\ \texttt{f} \end{bmatrix}$, then $A_2 \oplus B_2 = \begin{bmatrix} \texttt{a} & \texttt{b} & \texttt{c} \\ \texttt{d} & \texttt{e} & \texttt{f} \end{bmatrix}$.

Lyndon-Schützenberger Theorem (Redux)

UNIVERSITY OF WATERLOO

- Using our definitions, we can adapt the Lyndon-Schützenberger theorem for 1D words to produce an analogous theorem for 2D arrays.

Theorem (2D Lyndon-Schützenberger Theorem)

Let A and B be nonempty arrays. Then the following three conditions are equivalent:

1. *There exist positive integers p_1, p_2, q_1, q_2 such that $A^{p_1 \times q_1} = B^{p_2 \times q_2}$;*
2. *There exist a nonempty array C and positive integers r_1, r_2, s_1, s_2 such that $A = C^{r_1 \times s_1}$ and $B = C^{r_2 \times s_2}$;*
3. *There exist positive integers t_1, t_2, u_1, u_2 such that $A^{t_1,t_2} \circ B^{u_1,u_2} = B^{u_1,u_2} \circ A^{t_1,t_2}$ where $\circ$ can be either $\varobar$ or $\ominus$.*

Remark

- Conditions 1, 2, and 3 in the 2D version correspond to conditions 3, 2, and 4, respectively, in the 1D version.
- Here, we prove $2 \Rightarrow 1$ and $2 \Rightarrow 3$. (Other directions omitted.)

2. There exist a nonempty array C and positive integers r_1, r_2, s_1, s_2 such that $A = C^{r_1 \times s_1}$ and $B = C^{r_2 \times s_2}$.

$\Downarrow$

1. There exist positive integers p_1, p_2, q_1, q_2 such that $A^{p_1 \times q_1} = B^{p_2 \times q_2}$.

Proof.

Let $p_1 = r_2$, $p_2 = r_1$, $q_1 = s_2$, and $q_2 = s_1$. Then

$$\begin{aligned} A^{p_1 \times q_1} &= (C^{r_1 \times s_1})^{p_1 \times q_1} \\ &= C^{p_1 r_1 \times q_1 s_1} \\ &= C^{r_2 p_2 \times s_2 q_2} \\ &= (C^{r_2 \times s_2})^{p_2 \times q_2} \\ &= B^{p_2 \times q_2}. \end{aligned}$$

□

2. There exist a nonempty array C and positive integers r_1, r_2, s_1, s_2 such that $A = C^{r_1 \times s_1}$ and $B = C^{r_2 \times s_2}$.

$\Downarrow$

3. There exist positive integers t_1, t_2, u_1, u_2 such that $A^{t_1, t_2} \circ B^{u_1, u_2} = B^{u_1, u_2} \circ A^{t_1, t_2}$ where $\circ$ can be either $\varobar$ or $\ominus$.

Proof.

Assume the operation is $\varobar$. (The proof is similar for $\ominus$.)
Let $t_1 = r_2$, $t_2 = r_1$, $u_1 = s_2$, and $u_2 = s_1$. Then

$$
\begin{aligned}
A^{t_1 \times u_1} \varobar B^{t_2 \times u_2} &= (C^{r_1 \times s_1})^{t_1 \times u_1} \varobar (C^{r_2 \times s_2})^{t_2 \times u_2} \\
&= C^{r_1 t_1 \times s_1 u_1} \varobar C^{r_2 t_2 \times s_2 u_2} \\
&\vdots \\
&= C^{r_2 t_2 \times s_2 u_2} \varobar C^{r_1 t_1 \times s_1 u_1} \\
&= (C^{r_2 \times s_2})^{t_2 \times u_2} \varobar (C^{r_1 \times s_1})^{t_1 \times u_1} \\
&= B^{t_2 \times u_2} \varobar A^{t_1 \times u_1}.
\end{aligned}
$$

$\square$

Lyndon-Schützenberger Theorem (Redux)

- As a corollary to the 2D version of the Lyndon-Schützenberger theorem, we get the following result which will come in handy for the next topic.

Corollary

Given a nonempty array A, there exist a unique primitive array C and positive integers i and j such that $A = C^{i \times j}$.

Enumerating Primitive Arrays

- Over an alphabet of size k, there are

$$\psi_k(n) = \sum_{d|n} \mu(d) k^{n/d}$$

1D primitive words of length n, where $\mu(d)$ is the **Möbius function**, defined by

$$\mu(n) = \begin{cases} 1, & \text{if } n \text{ has an even number of prime factors;} \\ -1, & \text{if } n \text{ has an odd number of prime factors; and} \\ 0, & \text{if } n \text{ has a squared prime factor.} \end{cases}$$

Enumerating Primitive Arrays

- How do we arrive at this formula?

$$\psi_k(n) = \sum_{d \mid n} \mu(d) k^{n/d}$$

Enumerating Primitive Arrays

UNIVERSITY OF WATERLOO

- How do we arrive at this formula?

$$\psi_k(n) = \sum_{d|n} \mu(d) k^{n/d}$$

- The sum $\sum_{d|n}$ sums over all positive divisors d of n.

- How do we arrive at this formula?

$$\psi_k(n) = \sum_{d|n} \mu(d) k^{n/d}$$

- The sum $\sum_{d|n}$ sums over all positive divisors d of n.
- The Möbius function $\mu(d)$ is obtained by the earlier definition.

Enumerating Primitive Arrays

- How do we arrive at this formula?

$$\psi_k(n) = \sum_{d|n} \mu(d) k^{n/d}$$

- The sum $\sum_{d|n}$ sums over all positive divisors d of n.
- The Möbius function $\mu(d)$ is obtained by the earlier definition.
- The expression $k^{n/d}$ counts the number of k-ary words of length n/d.

Enumerating Primitive Arrays

- How do we arrive at this formula?

$$\psi_k(n) = \sum_{d|n} \mu(d) k^{n/d}$$

- Since there are k^n possible k-ary words of length n, and each word of length n is concatenated from copies of some primitive word of length k^d, where $d|n$, then the sum counts all k-ary words and the Möbius function removes the non-primitive words.
- To be precise, this formula is obtained via a process called **Möbius inversion**. (For more details, see Hardy and Wright, *An Introduction to the Theory of Numbers*.)

Enumerating Primitive Arrays

UNIVERSITY OF WATERLOO

Example

Enumerating all primitive words of length 4 over a binary alphabet:

$$\begin{aligned}\psi_2(4) &= \sum_{d|4} \mu(d) 2^{4/d} \\ &= \mu(1)2^{4/1} + \mu(2)2^{4/2} + \mu(4)2^{4/4} \\ &= (1)(2^4) + (-1)(2^2) + (0)(2^1) \\ &= 16 \text{ total words } - \underbrace{4 \text{ non-primitive words}}_{\text{copies of } 00,01,10,11}\end{aligned}$$

Indeed, the 12 primitive words of length 4 over the alphabet $\{0, 1\}$ are 0001, 0010, 0011, 0100, 0110, 0111, 1000, 1001, 1011, 1100, 1101, and 1110.

Enumerating Primitive Arrays

- Again, we can adapt the 1D version of this formula to produce an analogous 2D version that enumerates all primitive arrays of size $m \times n$.
- The 2D version of the formula is surprisingly straightforward.

Enumerating Primitive Arrays

Theorem

Over an alphabet of size k, there are

$$\psi_k(m,n) = \sum_{d_1|m} \sum_{d_2|n} \mu(d_1)\mu(d_2) k^{mn/(d_1 d_2)}$$

primitive arrays of size $m \times n$.

UNIVERSITY OF WATERLOO

Proof.

Define $g(m,n) = k^{mn}$. By our corollary, each of these $m \times n$ arrays has a unique primitive root of size $d_1 \times d_2$, where $d_1|m$ and $d_2|n$. Thus, $g(m,n) = \sum_{\substack{d_1|m \\ d_2|n}} \psi_k(d_1,d_2)$.

By Möbius inversion,

$$\sum_{\substack{d_1|m \\ d_2|n}} \mu(d_1)\mu(d_2)\, g\left(\frac{m}{d_1}, \frac{n}{d_2}\right) = \sum_{d_1|m} \mu(d_1) \sum_{d_2|n} \mu(d_2) \sum_{\substack{c_1|m/d_1 \\ c_2|n/d_2}} \psi_k(c_1,c_2)$$

$$= \sum_{c_1 d_1|m} \mu(d_1) \sum_{c_2 d_2|n} \mu(d_2)\, \psi_k(c_1,c_2)$$

$$= \sum_{\substack{d_1|m/c_1 \\ d_2|n/c_2}} \mu(d_1)\mu(d_2) \sum_{c_1|m} \sum_{c_2|n} \psi_k(c_1,c_2).$$

UNIVERSITY OF WATERLOO

Proof (Cont.)

$$\sum_{\substack{d_1|m \\ d_2|n}} \mu(d_1)\mu(d_2)\, g\left(\frac{m}{d_1}, \frac{n}{d_2}\right) = \underbrace{\sum_{\substack{d_1|m/c_1 \\ d_2|n/c_2}} \mu(d_1)\mu(d_2)} \sum_{c_1|m} \sum_{c_2|n} \psi_k(c_1, c_2)$$

Let $r = m/c_1$ and $s = n/c_2$. By a property of the sum of the Möbius function, the bracketed expression evaluates to 1 if $r = 1$ and $s = 1$; that is, if $c_1 = m$ and $c_2 = n$. Therefore, in this case the sum reduces to $\psi_k(m, n)$, and we get

$$\sum_{\substack{d_1|m \\ d_2|n}} \mu(d_1)\mu(d_2) k^{(m/d_1)(n/d_2)} = \psi_k(m, n).$$

□

Checking Primitivity of an Array

UNIVERSITY OF WATERLOO

- The literature features a good deal of previous work on pattern matching in two-dimensional arrays.
- However, none of this work is directly related to the matters of primitivity or periodicity.
- It would be desirable to have an (efficient) algorithm to check the primitivity of an array.

Checking Primitivity of an Array

- Could we take the elements of the array in row-major/column-major order, then check if this resulting word is primitive?
- No, since this method does not work in some cases.

Checking Primitivity of an Array

- Could we take the elements of the array in row-major/column-major order, then check if this resulting word is primitive?
- No, since this method does not work in some cases.

Example

The matrix $\begin{bmatrix} \mathtt{a} & \mathtt{a} \\ \mathtt{b} & \mathtt{b} \end{bmatrix}$ is not 2D primitive.

Its row-majorized word `aabb` is 1D primitive.

Example

The matrix $\begin{bmatrix} \mathtt{a} & \mathtt{b} & \mathtt{a} \\ \mathtt{b} & \mathtt{a} & \mathtt{b} \end{bmatrix}$ is 2D primitive.

Its row-majorized word `ababab` is not 1D primitive.

Checking Primitivity of an Array

Theorem

It is possible to check the primitivity of an $m \times n$ array and to compute the primitive root in $O(mn)$ time, for fixed alphabet size.

Proof.

The algorithm on the following slide computes the primitive root of an $m \times n$ array in linear time. If the primitive root is equal to the original array, then the primitivity of the array is also verified in linear time. □

Checking Primitivity of an Array

Algorithm: Computing the primitive root of A

procedure $\textsc{2DPrimitiveRoot}(A)$
    for $0 \leq i < m$ **do**
        $r_i \leftarrow \textsc{1DPrimitiveRoot}(A[i, 0..n-1])$
    $q \leftarrow \text{lcm}(|r_0|, |r_1|, \ldots, |r_{m-1}|)$
    for $0 \leq j < n$ **do**
        $c_j \leftarrow \textsc{1DPrimitiveRoot}(A[0..m-1, j])$
    $p \leftarrow \text{lcm}(|c_0|, |c_1|, \ldots, |c_{n-1}|)$
    for $0 \leq i < p$ **do**
        for $0 \leq j < q$ **do**
            $C[i, j] \leftarrow A[i, j]$
    return (C, p, q)

Checking Primitivity of an Array

- We make the following observations.

Remark

- We assume there exists an algorithm $\text{1DPRIMITIVEROOT}(w)$ to obtain the primitive root of some word w.
- A word w is primitive if and only if w is not a factor of the word $w_F w_L$, where w_F is w with the first symbol removed and w_L is w with the last symbol removed.
- Checking the above property can be done in linear time by using, for example, the Knuth-Morris-Pratt string-matching algorithm.

Checking Primitivity of an Array

- We also require the following lemma.

Lemma

Let A be an $m \times n$ array. Let the primitive root of row i of A be r_i and the primitive root of column j of A be c_j. Then the primitive root of A has dimension $p \times q$, where

$$q = \operatorname{lcm}(|r_0|, |r_1|, \ldots, |r_{m-1}|)$$

and

$$p = \operatorname{lcm}(|c_0|, |c_1|, \ldots, |c_{n-1}|).$$

Checking Primitivity of an Array

Algorithm: Computing the primitive root of A

for $0 \leq i < m$ **do**
  $r_i \leftarrow \textsc{1DPrimitiveRoot}(A[i, 0..n-1])$
$q \leftarrow \mathrm{lcm}(|r_0|, |r_1|, \ldots, |r_{m-1}|)$

- This loop computes the primitive root of each row in A.
- Each primitive root is stored in r_i and the least common multiple of the primitive roots of rows is stored in q.

Checking Primitivity of an Array

UNIVERSITY OF WATERLOO

Algorithm: Computing the primitive root of A

for $0 \le j < n$ **do**
    $c_j \leftarrow \textsc{1DPrimitiveRoot}(A[0..m-1, j])$
$p \leftarrow \mathrm{lcm}(|c_0|, |c_1|, \ldots, |c_{n-1}|)$

- This loop computes the primitive root of each column in A.
- Each primitive root is stored in c_i and the least common multiple of the primitive roots of columns is stored in p.

Checking Primitivity of an Array

Algorithm: Computing the primitive root of A

```
for 0 ≤ i < p do
    for 0 ≤ j < q do
        C[i, j] ← A[i, j]
```

- This loop iterates through the array A and keeps only those elements in A that comprise the primitive array C.
- By our lemma, this primitive array C is of dimension $p \times q$.

Table of Contents

Conclusions

UNIVERSITY OF WATERLOO

- The one-dimensional version of the Lyndon-Schützenberger Theorem admits two new equivalent conditions.
- There exists an analogous two-dimensional version of the Lyndon-Schützenberger Theorem.
- There exists a rather simple formula to count the number of primitive arrays of size $m \times n$ over a k-letter alphabet.
- We can check the primitivity of an $m \times n$ array and compute its primitive root in linear time.

Future Work

UNIVERSITY OF WATERLOO

- Is there a two-dimensional analogue to conditions 1 and 5 of the 1D Lyndon-Schützenberger Theorem?
- Can we investigate primitivity and periodicity in dimensions higher than 2?
- Define a **pedal triangle** as the triangle obtained by dropping perpendiculars from a point P within a triangle $\angle ABC$ to each side of $\angle ABC$. If the nth pedal triangle is similar to the original triangle, then the **period** of this triangle is equal to n. Interestingly, the sequence $\psi_2(2, n)$ counts the number of pedal triangles with period exactly n. How are these concepts related?

References

[1] G. H. Hardy and E. M. Wright. *An Introduction to the Theory of Numbers*. Oxford University Press, Oxford, 6th edition, 2008.

[2] R. C. Lyndon and M.-P. Schützenberger. The equation $a^M = b^N c^P$ in a free group. *Mich. Math. J.*, 9(4):289–298, 1962.

[3] J. Shallit and T. J. Smith. Periodicity in rectangular arrays. arXiv:1602.06915.