

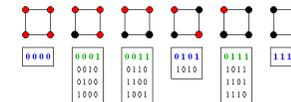
<http://www.theory.cs.uvic.ca/~cos/inf/neck/NecklaceInfo.html>

 JUN 13 2018
 23 searches
 29 Aug 2001 - 11 Jun 2017

Information on necklaces, unlabelled necklaces, Lyndon words, De Bruijn sequences

A *k*-ary necklace is an equivalence class of *k*-ary strings under rotation. We take the lexicographically smallest such string as the representative of each equivalence class and use this in the output of the program. A *Lyndon word* is an aperiodic necklace representative.

The illustration on the right shows the 6 binary necklaces with 4 beads and the corresponding equivalence classes of strings. The representatives are colored and green indicates a Lyndon word.



The number of binary necklaces for $n=1,2,\dots,15$ is 2, 3, 4, 6, 8, 14, 20, 36, 60, 108, 188, 352, 632, 1182, 2192. This is sequence [A000031\(M0564\)](#) in Neil J. Sloane's [database](#) of integer sequences.

The number of binary Lyndon words for $n=1,2,\dots,15$ is 2, 1, 2, 3, 6, 9, 18, 30, 56, 99, 186, 335, 630, 1161, 2182. This is sequence [A001037\(M0116\)](#) in Neil J. Sloane's [database](#) of integer sequences.

There are explicit formulae for both of these sequences, given below over a *k*-ary alphabet:

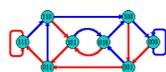
$$L_k(n) = \frac{1}{n} \sum_{d|n} \mu(n/d) k^d, \quad N_k(n) = \frac{1}{n} \sum_{d|n} \phi(n/d) k^d.$$

Lyndon words
Necklaces

The function $\mu(m)$ is the Möbius function: 0 if *m* is the product of non-distinct primes, +1 if it is the product of an even number of distinct primes, and -1 otherwise. For example, $\mu(1) = 1$, $\mu(4) = 0$, $\mu(6) = +1$, and $\mu(7) = -1$. The function $\phi(m)$ is Euler's totient function, the number of numbers *k* in the range $1 \leq k \leq m$ that are relatively prime to *m*. For example $\phi(1) = |\{1\}| = 1$, and $\phi(6) = |\{1,5\}| = 2$.

A *k*-ary De Bruijn sequence of order *n* is a circular *k*-ary string containing every *k*-ary string of length *n* as a substring exactly once. Thus a De Bruijn sequence has length k^n . For $n = 3$ and $k = 2$, the lexicographically smallest De Bruijn sequence is 00010111.

Every De Bruijn sequence corresponds to a Eulerian cycle on a De Bruijn graph, one of which is shown below (for $n = 4$ and $k = 2$).



At first glance De Bruijn sequences appear to have little to do with necklaces or Lyndon words. However, Fredericksen has proven that the lexicographic sequence of Lyndon words of lengths divisible by *n* gives the lexicographically least De Bruijn sequence, and that is the one output by our program.

The algorithm used is from the Combinatorial Generation book. It is a recursive algorithm and generates necklaces in lexicographic order, as does the first efficient necklace generation algorithm, the iterative algorithm of Fredricksen, Maiorana, and Kelsen.

Lyndon and Necklace Factorizations

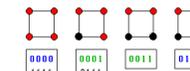
Given a string it is often useful to compute its necklace. Such applications arise in several diverse areas such as graph drawing, where it is used to help determine the symmetries of a graph to be drawn in the plane. To accomplish this task, we make use of necklace factorizations. Every word *a* has a unique necklace factorization $a = a_1 a_2 \dots a_m$, such that each a_i is a necklace and $a_i > a_j$ for all $i < j$. Similarly, every word can be factored uniquely into Lyndon words except that the factors satisfy $a_i \geq a_j$ for all $i < j$. The inequalities between words are with respect to lexicographic order.

Duval has developed an elegant and efficient algorithm for factoring a word that allows us to compute the necklace of a string.

Unlabelled Necklaces

Unlabelled necklaces are an equivalence classes of necklaces under permutation of alphabet symbols. Note that by permuting the alphabet symbols of a necklace and then rotating the string into its lexicographically smallest position, the result is a necklace representative. Among all permutations of alphabet symbols that resulting necklace that is lexicographically smallest is used as the representative of an unlabelled necklace. For example, here is an equivalence class of unlabelled ternary necklaces: { 011222, 022111, 001112, 002221, 000122, 000211 }. The lexicographically smallest of these is 000122 and so it is chosen as the representative.

The illustration on the right shows the 4 unlabelled binary necklaces with 4 beads and the corresponding equivalence classes of necklaces. The representatives are colored and green indicates an unlabelled Lyndon word.



The number of unlabelled necklaces for $n=1,2,\dots,15$ is 1, 2, 2, 4, 4, 8, 10, 20, 30, 56, 94, 180, 316, 596, 1096. This is sequence [A000013](#) in Neil J. Sloane's [database](#) of integer sequences.

The number of unlabelled Lyndon words for $n=1,2,\dots,15$ is 1, 1, 1, 2, 3, 5, 9, 16, 28, 51, 93, 170, 315, 585, 1091. This is sequence [A000048](#) in Neil J. Sloane's [database](#) of integer sequences.

Necklaces with Fixed Density

In many applications not all necklaces are required, but rather only those of fixed density (the number of zeroes is fixed). In the more general case, one may want a list of necklaces where the number of occurrences for every character is fixed.

To count fixed density necklaces we let $N(n_0, n_1, \dots, n_{k-1})$ denote the number of necklaces composed of n_i occurrences of the symbol *i*, for $i = 0 \dots k-1$. Let the density of the necklace $d = n_1 + n_2 + \dots + n_{k-1}$ and $n_0 = n - d$. It is known that

$$N(n_0, n_1, \dots, n_{k-1}) = \frac{1}{n} \sum_{j | \gcd(n_0, n_1, \dots, n_{k-1})} \phi(j) \left(\frac{(n/j)!}{(n_0/j)! (n_1/j)! \dots (n_{k-1}/j)!} \right)$$

To get the number of fixed density necklaces with length *n* and density *d*, we sum over all possible values of n_1, n_2, \dots, n_{k-1}

$$N_k(n, d) = \sum_{n_1 + \dots + n_{k-1} = d} N(n_0, n_1, \dots, n_{k-1})$$

The number of fixed density Lyndon words are counted similarly

$$L_k(n_0, n_1, \dots, n_{k-1}) = \frac{1}{n} \sum_{j | \gcd(n_0, n_1, \dots, n_{k-1})} \mu(j) \left(\frac{(n/j)!}{(n_0/j)! (n_1/j)! \dots (n_{k-1}/j)!} \right)$$

$$L_k(n, d) = \sum_{n_1 + \dots + n_{k-1} = d} L(n_0, n_1, \dots, n_{k-1})$$

For the binary case, these formulas simplify as follow:

$$L_2(n, d) = \frac{1}{n} \sum_{j | \gcd(n, d)} \mu(j) \binom{n/j}{d/j} \quad N_2(n, d) = \frac{1}{n} \sum_{j | \gcd(n, d)} \phi(j) \binom{n/j}{d/j}$$

<http://www.theory.cs.uvic.ca/~cos/inf/neck/NecklaceInfo.html>

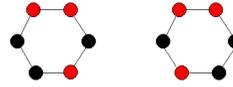
 JUL 2018
 13
 2015 2017 2018

Necklaces that do not contain a specified sequence as a substring are known as necklaces with forbidden sequences. Currently we restrict the forbidden sequence to be of the form 0^k1. For example, when considering all necklaces with $n=4$ and $k=2$ with the restriction that there are no 00 (or 0²) substrings we get the set {0101, 0111, 1111}. Notice that this is precisely the set of necklaces that start with 0, where 1 k

The number of necklaces with no 00 subsequence for $n=1,2,\dots,15$ is 1, 2, 2, 3, 3, 5, 5, 8, 10, 15, 19, 31, 41, 64, 94. This is sequence [A000358/M0564](#) in Neil J. Sloane's [database](#) of integer sequences.

Bracelets

A *bracelet* is a necklace that can be turned over.



The formula for a bracelet is:

$$B(n) = \begin{cases} \frac{1}{2} N_k(n) + \frac{1}{4} (k+1)k^{n/2} & \text{if } n \text{ is even} \\ \frac{1}{2} N_k(n) + \frac{1}{2} k^{(n+1)/2} & \text{if } n \text{ is odd} \end{cases}$$

where $N_k(n) = \frac{1}{n} \sum_{d|n} \phi(n/d) k^d$,

For $n = 1, 2, \dots, 20$ the number of bracelets is 2, 3, 4, 6, 8, 13, 18, 30, 46, 78, 126, 224, 380, 687, 1224, 2250, 4112, 7685, 14310, 27012. This is sequence [A000029/M0563](#) in Neil J. Sloane's [database](#) of integer sequences.

For $n = 1, 2, \dots, 20$ the number of Lyndon bracelets is 2, 1, 2, 3, 6, 8, 16, 24, 42, 69, 124, 208, 378, 668, 1214, 2220, 4110, 7630, 14308, 26931. This is sequence [A001371/M0115](#) in Neil J. Sloane's [database](#) of integer sequences.

The number of nonisomorphic unit interval graphs is the same as the number of bracelets with n black and $n-1$ white beads.

Information about the number of Lyndon words with given trace (sum of characters) mod q may be found [here](#).

- Programs available:
- Necklaces, Lyndon words, De Bruijn sequences: [Pascal program](#) ✓, [C program](#) ✓
 - Lyndon and necklace factorizations: [Pascal program](#) ✗, [C program](#) ✗
 - Unlabelled necklaces and Lyndon words: [Pascal program](#) ✓, [C program](#) ✓
 - Fixed density necklaces and Lyndon words: [Pascal program](#) ✗, [C program](#) ✓



Questions?? Email [The wizard of COS](#).
 (Please note that the suffix XXXX must be removed from the preceding email address.)
 It was last updated Wednesday, 10-May-2006 10:32:14 PDT.
 There have been 36164 visitors to this page since May 16, 2000.
 ©Frank Ruskey, 1995-2003.