SOMAIYA
VIDYAVIHAR UNIVERSITY
K J Somaiya College of Engineering

Somaiya TRUST

Development of sieve of Eratosthenes and sieve of Sundaram's proof

Sieve of Eratosthenes

The Sieve of Eratosthenes is an algorithm that creates a list of numbers from 2 to a given limit, and then marks off the multiples of each prime number, starting with 2. The unmarked numbers that remain in the list are prime. It is a simple and efficient algorithm that has been used for centuries and is still widely used today in modern cryptography and computer science.

Time Complexity of seive of Sundaram

O(nlogn))

Time Complexity of seive of Eratosthenes

O(nlog(logn))

Sieve of Sundaram

The Sieve of Sundaram is an algorithm that creates a table of odd numbers up to a given limit, and then marks out all the numbers that can be expressed in the form of 2p + 1, where p is a prime number. The remaining unmarked numbers are all prime. It was developed by the Indian mathematician S. P. Sundaram in 1934 and is a very efficient algorithm for finding prime numbers, especially for larger limits.

Algorithm of sieve of Eratosthenes

- Create a list of all numbers from 2 to the given limit.
- Set the value of the current prime number to 2.
- Mark all the multiples of the current prime number as not prime.
- Move to the next unmarked number in the list.
- If the next unmarked number is less than or equal to the square root of the limit, set it as the new current prime number and go to step 3.
- If the next unmarked number is greater than the square root of the limit, all the remaining unmarked numbers in the list are prime.

Algorithm of sieve of Sundaram

- Create a list of all odd numbers from 3 to the given limit.

(Note: 2 is not included in the list because it is the only even prime number.)

- Create a variable called "j" and initialize it to 1.
- For each value of "i" from 1 to "j", calculate the value of "n" using the formula: n = i + j + 2ij
- Mark all the values of "n" in the list as not prime.
- If "j" < ("limit" - 2) / (2 * j + 1), increment "j" by 1 and go to step 3.
- All the remaining unmarked numbers in the list are prime.

Enter the value of N: 10
Output: 2, 3, 5, 7

Enter the value of N: 50
Output: 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47

Enter the value of N: 75
Output: 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73

Enter the value of N: 100
Output: 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97

First development of SOE (D1SOE)

- The code implements the Sieve of Sundaram algorithm to find prime numbers up to a given limit.
- An array of size (n_m+1) is initialized with all elements set to false.
- The mean elimination theorem is used to eliminate composite numbers from the array up to a calculated limit.
- The outer loop iterates over values of n from 1 to the calculated limit.
- The inner loop iterates over multiples of (2*n+1) to mark corresponding array elements as true for composite numbers.
- The prime numbers are printed by iterating over the array and printing the values of (2*n + 1) for false array elements.

Second development of SOE (D2SOE)

- The D2SOE algorithm generates prime numbers up to a given limit.
- It is a modified version of the Sieve of Eratosthenes that uses two sieves to eliminate composite numbers.
- It initializes an array of boolean values and marks all multiples of 2 and 3 as composite.
- It uses the mean elimination theorem to determine which numbers to eliminate next.
- It eliminates all multiples of primes up to the square root of the upper limit.
- The remaining numbers in the array are all prime and are printed out by the program.

Proof and development of sieve of Sundaram (DSOS)

- Implements Sieve of Sundaram algorithm to find prime numbers up to a limit n_m.
- Initializes an array of size (n_m+1) with all elements set to false.
- Eliminates composite numbers from the array using the mean elimination theorem.
- Prints all prime numbers by iterating over the array and printing the values of (2*n + 1) for all indices where the array element is false.

N	SOE	D1SOE	D2SOE	DSOS
10^3	10	10	10	9
10^4	92	53	32	54
10^5	978	472	263	450
10^6	12793	6021	3492	5812
10^7	190523	88483	53848	86673
10^8	2160761	1045056	741409	1038911
10^9	24837596	12143628	8471540	12008482
$2 * 10^9$	.....	25136075	17953830	25208467

References: https://arxiv.org/abs/2102.06653

Arya Nair (Roll no: 16010421063 / Batch: A2)