

Factorization Batch Algorithm and Prime Numbers Distribution

Shaimaa said soltan¹

¹ Computer Engineer, Toronto, Canada

Correspondence: Shaimaa Soltan, 3050 Constitution Blvd, Mississauga, ON., L4Y 3X1, Canada. Tel: 1-647-801-6063 E-mail: shaimaasultan@hotmail.com

Suggested Reviewers (Optional)

Please suggest 3-5 reviewers for this article. We may select reviewers from the list below in case we have no appropriate reviewers for this topic.

Name:	E-mail:
Affiliation:	

Name:	E-mail:
Affiliation:	

Name:	E-mail:
Affiliation:	

Name:	E-mail:
Affiliation:	

Name:	E-mail:
Affiliation:	

Factorization Batch Algorithm and Prime Numbers Distribution

Abstract

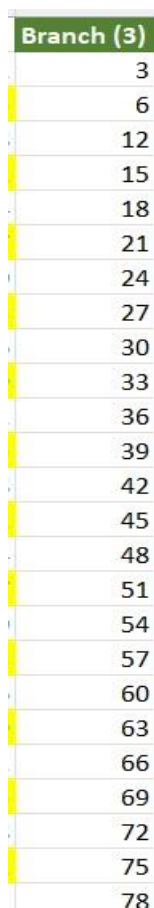
This paper introduces a primary set for Natural numbers that represents the distribution of prime numbers. Using this primary set of Natural numbers, we are going to introduce $O(\sqrt{N})$ batch algorithm to get a set of all prime numbers up until number N. and a batch factorization algorithm for its prime composites.

Keywords: Prime Number, Distribution

1. Introduction

1.1 Pyramid distribution.

In previous paper we showed a pyramid distribution for all natural numbers², and one of the branches or sides of the pyramid was only for natural number 3 and its multipliers. Any number in this branch except natural number 3 for sure is not Prime.



Branch (3)
3
6
12
15
18
21
24
27
30
33
36
39
42
45
48
51
54
57
60
63
66
69
72
75
78

Figure 1. Branch (3) have numbers that are divisible by natural number [3]

And the other two sides for the pyramid include the rest of all natural numbers.

One of the other two sides of the pyramid starts with natural number 1 (Branch (1)), increases with step = 3 (i.e., difference between numbers in this branch = 3)

The other side starts with natural number 2 (Branch (2)) and increases with step = 3 (i.e., difference between numbers in this branch = 3)

Branch (1)	Branch (2)
1	2
4	5
7	8
10	11
13	14
16	17
19	20
22	23
25	26
28	29
31	32
34	35
37	38
40	41
43	44
46	47
49	50
52	53
55	56
58	59
61	62
64	65
67	68
70	71

Figure 2. Pyramid distribution Branch (1) and Branch (2) lists

These two sides or branches (Branch (1) and Branch (2)) include all the prime numbers and some other numbers.

As it shows in Figure 2. If we choose numbers only that are highlighted in yellow, we simply will be going to remove all even numbers from both branches.

By this step these two branches now do not have any number that can be divided by 2 or 3.

Now we need to remove any number that is divisible by 5 from these two Branches (Branch (1) and Branch (2)).

If we rearrange these two branches and only showed the numbers that are highlighted in Yellow

Then we will get a new filtered branch without any even number.

Branch (1)	Branch (2)
1	5
7	11
13	17
19	23
25	29
31	35
37	41
43	47
49	53
55	59
61	65
67	71
73	77
79	83
85	89
91	95
97	101
103	107
109	113
115	119
121	125
127	131
133	137
139	143
145	149
151	155
157	161
163	167
169	173

Branch (1)	Branch (2)
1	5
7	11
13	17
19	23
25	29
31	35
37	41
43	47
49	53
55	59
61	65
67	71
73	77
79	83
85	89
91	95
97	101
103	107
109	113
115	119
121	125
127	131
133	137
139	143
145	149
151	155
157	161
163	167
169	173

Figure 3. Branch (1) and Branch (2) after Filtered out even numbers.

Note (1): These two filter Branches, one starts at 1 with step = 6 and the other one starts at 5 and with step = 6 also.

Note (2): Branch (2) except natural number 5, Branch (2) have the same numbers in Branch (1) but increased by step = 10

Therefore, if we removed any number that is divisible by 5 from Branch (1) we will not going to have any number divisible by 5 in Brach (2) as well, because all numbers in branch (2) are the numbers in branch (1) + 10.

In order to remove all numbers that are divisible by 5. We know that Any number that ends by digit 5 will be divisible by 5.

Therefore, we can refilter Branch (1) to only numbers that ends in digits [1, 7, 3, 9].

Now the new Branch (1) will not be going to have any number that is divisible by 5. And there for Branch (2) will not going to have any number divisible by 5 as well as Branch (2) = Branch (1) + 10.

At this stage, by these simple steps we filtered Branch (1) to numbers not divisible by [2, 3, 5].

Branch (1)	Branch (2)
1	11
7	17
13	23
19	29
31	41
37	47
43	53
49	59
61	71
67	77
73	83
79	89
91	101
97	107
103	113
109	119
121	131
127	137
133	143
139	149
151	161
157	167
163	173
169	179
181	191
187	197
193	203
199	209
211	221
217	227

	Branch (1)		Branch (2)
6+	1	10+	11
6+	7	10+	17
6+	13	10+	23
12+	19	10+	29
6+	31	10+	41
6+	37	10+	47
6+	43	10+	53
12+	49	10+	59
6+	61	10+	71
6+	67	10+	77
6+	73	10+	83
12+	79	10+	89
6+	91	10+	101
6+	97	10+	107
6+	103	10+	113
12+	109	10+	119
6+	121	10+	131
6+	127	10+	137
6+	133	10+	143
12+	139	10+	149
6+	151	10+	161
6+	157	10+	167
6+	163	10+	173
12+	169	10+	179
6+	181	10+	191
6+	187	10+	197
6+	193	10+	203
12+	199	10+	209
6+	211	10+	221
6+	217	10+	227

Figure 4. Branch (1) and Branch (2) after filter out [5]

Note (1): all number in Branch (1) ends in digits [1,7,3,9] in this order.

Note (2): all numbers in Branch (1) if ends in [1 ,7, 3] next number will be by adding 6.

Note (3): all numbers in branch (1) if ends in [9] next number will be by adding 12.

Note (4): all numbers in Branch (2) = 10 + Branch (1)

From these notes Branch (1) is the main Prime numbers distribution Branch.

And by this stage we can reconstruct Branch (1) easily by just these 4 Notes.

To elaborate more about statement “Branch (1) is the main Prime numbers distribution Branch”.

Let us see the results of finding the factors using divide by each number in the Branch. You will see in the next Figure how these numbers in branch (1) keeps shown at each factor.

Branch(1)/37	Branch(1)/31	Branch(1)/19	Branch(1)/13	Branch(1)/7	Branch (1)	Branch (2)	Branch(2)/11	Branch(2)/17	Branch(2)/23	Branch(2)/29	Branch(2)/41	Branch(2)/47
					1	11	1					
				1	7	17	1.545454545	1				
			1	1.85714286	13	23	2.090909091	1.352941176	1			
		1	1.461538462	2.71428571	19	29	2.636363636	1.705882353	1.260869565	1		
	1	1.631578947	2.384615385	4.42857143	31	41	3.727272727	2.411764706	1.782608696	1.413793103	1	
1	1.193548387	1.947368421	2.846153846	5.28571429	37	47	4.272727273	2.764705882	2.043478261	1.620689655	1.146341463	1
1.162162162	1.387096774	2.263157895	3.307692308	6.14285714	43	53	4.818181818	3.117647059	2.304347826	1.827586207	1.292682927	1.127659574
1.324324324	1.580645161	2.578947368	3.769230769	7	49	59	5.363636364	3.470588235	2.565217391	2.034482759	1.43902439	1.255319149
1.648648649	1.967741935	3.210526316	4.692307692	8.71428571	61	71	6.454545455	4.176470588	3.086956522	2.448275862	1.731707317	1.510638298
1.810810811	2.161290323	3.526315789	5.153846154	9.57142857	67	77	7	4.529411765	3.347826087	2.655172414	1.87804878	1.638297872
1.972972973	2.35483871	3.842105263	5.615384615	10.4285714	73	83	7.545454545	4.882352941	3.608695652	2.862068966	2.024390244	1.765957447
2.135135135	2.548387097	4.157894737	6.076923077	11.2857143	79	89	8.090909091	5.235294118	3.869565217	3.068965517	2.170731707	1.893617021
2.459459459	2.935483871	4.789473684	7	13	91	101	9.181818182	5.941176471	4.391304348	3.482758621	2.463414634	2.14893617
2.621621622	3.129032258	5.105263158	7.461538462	13.8571429	97	107	9.727272727	6.294117647	4.652173913	3.689655172	2.609756098	2.276595745
2.783783784	3.322580645	5.421052632	7.923076923	14.7142857	103	113	10.27272727	6.647058824	4.913043478	3.896551724	2.756097561	2.404255319
2.945945946	3.516129032	5.736842105	8.384615385	15.5714286	109	119	10.81818182	7	5.173913043	4.103448276	2.902439024	2.531914894
3.27027027	3.903225806	6.368421053	9.307692308	17.2857143	121	131	11.90909091	7.705882353	5.695652174	4.517241379	3.195121951	2.787234043
3.432432432	4.096774194	6.684210526	9.769230769	18.1428571	127	137	12.45454545	8.058823529	5.956521739	4.724137931	3.341463415	2.914893617
3.594594595	4.290322581	7	10.23076923	19	133	143	13	8.411764706	6.217391304	4.931034483	3.487804878	3.042553191
3.756756757	4.483870968	7.315789474	10.69230769	19.8571429	139	149	13.54545455	8.764705882	6.47826087	5.137931034	3.634146341	3.170212766
4.081081081	4.870967742	7.947368421	11.61538462	21.5714286	151	161	14.63636364	9.470588235	7	5.551724138	3.926829268	3.425531915
4.243243243	5.064516129	8.263157895	12.07692308	22.4285714	157	167	15.18181818	9.823529412	7.260869565	5.75862069	4.073170732	3.553191489
4.405405405	5.258064516	8.578947368	12.53846154	23.2857143	163	173	15.72727273	10.17647059	7.52173913	5.965517241	4.219512195	3.680851064
4.567567568	5.451612903	8.894736842	13	24.1428571	169	179	16.27272727	10.52941176	7.782608696	6.172413793	4.365853659	3.808510638
4.891891892	5.838709677	9.526315789	13.92307692	25.8571429	181	191	17.36363636	11.23529412	8.304347826	6.586206897	4.658536585	4.063829787
5.054054054	6.032258065	9.842105263	14.38461538	26.7142857	187	197	17.90909091	11.58823529	8.565217391	6.793103448	4.804878049	4.191489362
5.216216216	6.225806452	10.15789474	14.84615385	27.5714286	193	203	18.45454545	11.94117647	8.826086957	7	4.951219512	4.319148936
5.378378378	6.419354839	10.47368421	15.30769231	28.4285714	199	209	19	12.29411765	9.086956522	7.206896552	5.097560976	4.446808511
5.702702703	6.806451613	11.10526316	16.23076923	30.1428571	211	221	20.09090909	13	9.608695652	7.620689655	5.390243902	4.70212766
5.864864865	7	11.42105263	16.69230769	31	217	227	20.63636364	13.35294118	9.869565217	7.827586207	5.536585366	4.829787234
6.027027027	7.193548387	11.73684211	17.15384615	31.8571429	223	233	21.18181818	13.70588235	10.13043478	8.034482759	5.682926829	4.957446809
6.189189189	7.387096774	12.05263158	17.61538462	32.7142857	229	239	21.72727273	14.05882353	10.39130435	8.24137931	5.829268293	5.085106383
6.513513514	7.774193548	12.68421053	18.53846154	34.4285714	241	251	22.81818182	14.76470588	10.91304348	8.655172414	6.12195122	5.340425532
6.675675676	7.967741935	13	19	35.2857143	247	257	23.36363636	15.11764706	11.17391304	8.862068966	6.268292683	5.468085106
6.837837838	8.161290323	13.31578947	19.46153846	36.1428571	253	263	23.90909091	15.47058824	11.43478261	9.068965517	6.414634146	5.595744681
7	8.35483871	13.63157895	19.92307692	37	259	269	24.45454545	15.82352941	11.69565217	9.275862069	6.56097561	5.723404255
7.324324324	8.741935484	14.26315789	20.84615385	38.7142857	271	281	25.45454545	16.52941176	12.2173913	9.689655172	6.853658537	5.978723404
7.486486486	8.935483871	14.57894737	21.30769231	39.5714286	277	287	26.09090909	16.88235294	12.47826087	9.896551724	7	6.106382979
7.648648649	9.129032258	14.89473684	21.76923077	40.4285714	283	293	26.63636364	17.23529412	12.73913043	10.10344828	7.146341463	6.234042553
7.810810811	9.322580645	15.21052632	22.3076923	41.2857143	289	299	27.18181818	17.58823529	13	10.31034483	7.292682927	6.361702128

Figure 5. Primary List distributions.

As you see in the previous Figure Branch (1) numbers keeps shown in the distribution even if we are dividing by different number. We will get only natural numbers when we have Branch Factors.

To get only prime numbers from this Branches, we can use this method of dividing by each number in branch (1) and remove all numbers that give us natural numbers. And the list will be getting shorter each time to get closer to Prime numbers list.

Or we can use matrix.

If we add both Branches to our matrix (data frame columns names and row index) we can construct a true false matrix that we can find from it all the factors for these numbers in branch (2) and branch (1). And also, if we filter the rows for rows that have only two true maximum we get our prime numbers list as well of course also [2,3,5] primes.

Also Note that numbers now ordered as two numbers ends with digit [1], one number ends with digit [1] and the numbers after it by 10. Then two numbers end with digit [7], one number end with digit [7] and the number after it by 10. Then two numbers end by digit [3] one number ends by digit [3] and the number after it by 10. Then two numbers end by digit [9] and the number after it by 10.

{ 1 , 11 , 7 , 17 , 13 , 23 , 19 , 29 , 31 , 41 , 37 , 47 , 43 , 53 , 49 , 59 , }

[illegible]

Figure 6. Factorization and prime number data frame matrix.

[illegible]

Figure 7. Factorization and prime number data frame matrix.

This is the full matrix for factoring and finding the primes but in order to find all primes less than number [N] we do not need to use modules for all numbers in the list we only need to construct smaller matrix from [N rows and M columns] where M in Branch (1) list and $\leq \sqrt{n}$.

Given number N. Get list of prime number up until number N.

- 0- Initiate Branch (1) & Branch (2) (two sequence lists (+6, +12, +10)) up until number \leq given number N
- 1- Construct data frame matrix M of [A, B], where A (row names) = Branch lists, and B (columns name) \leq \sqrt{N} and B (column name) in Branches list.
- 2- Calculate (row name) mod (column name) on data frame M.
- 3- filter the data frame matrix for rows that only have exactly one True value.
- 4- return filtered Data frame Matrix M.

And as our branch list is limited list and its length approximate $N/4$.

And we use matrix operations, and we only need to loop on column names to apply Mod function, then our

algorithm $\leq O(\sqrt{N})$

Conclusion

First we showed a primary sequence list of natural numbers that we can use to get a distinct list of prime numbers, and we showed how these numbers are the primary list of prime numbers. Then showed how this distribution can be used to get next most likely prime number based on the described distribution. And how we introduced $O(\sqrt{N})$ batch algorithm to get list of prime numbers up until natural number [N].

References

- [1] A. O. L. Atkin, D. J. Bernstein, prime sieves using binary quadratic forms, mathematics of computation volume 73, number 246, pages 1023–1030 s 0025-5718(03)01501-1 article electronically published on December 19, 2003.
- [2] S. Soltan, Step Pyramid Distribution for Prime Numbers, Journal of Mathematics Research Vol. 14, No. 1 (2022), DOI:10.5539/jmr.v14n1p55

Copyrights

Copyright for this article is retained by the author(s), with first publication rights granted to the journal.

This is an open-access article distributed under the terms and conditions of the Creative Commons Attribution license (<http://creativecommons.org/licenses/by/3.0/>).