平行程式設計期末Project

第19組-B0429017陳蕾、B0429040王昀筠

1. 問題介紹

Q1：撰寫CUDA程式，找出小於1,000,000,000的質數

* + 找出最佳Block和Thread組合
  + 和OpenMP及MPI的Speedup比較

Q2：嘗試用CUDA找出更多的質數(並列出最大質數)

1. 實作方法

6n±1 Method

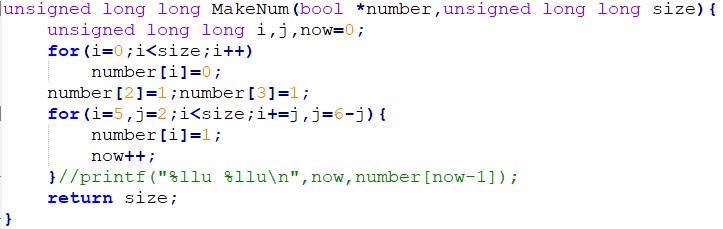
原理:跳過 2 和 3 的倍數

2和3的最小公倍數是6，我們把所有數字分為 :

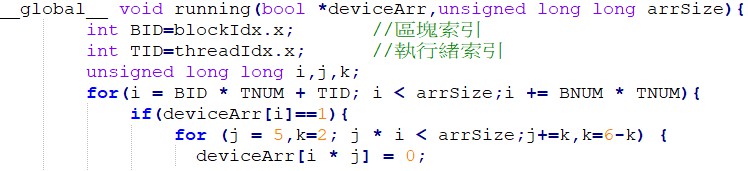
6n 、 6n+1 、 6n+2 、 6n+2 、 6n+3 、 6n+4 、 6n+5

可以看出 6n 、 6n+2 、 6n+3 、 6n+4 是 2 或 3 的倍數，不屬於質數。因此，只要驗證 6n+1 和 6n+5 是不是質數。

首先我們先嘗試做了**[第一版]**:



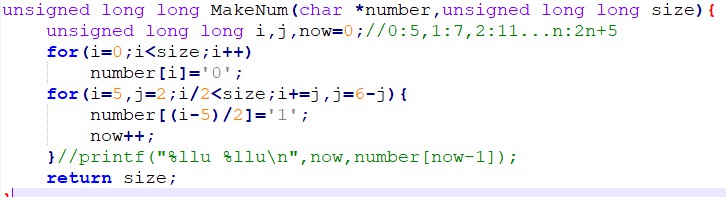
先排除2、3的倍數，剩下是有可能為質數的，先設為’1’，留到等一下再判斷。



這時候從5開始刪除他的倍數，再刪除7的倍數……以此類推。

這個版本只能存到164億以內最大的質數，因為覺得可以存的值有點不夠多，

所以稍微改良一下變成**[第二版]**:

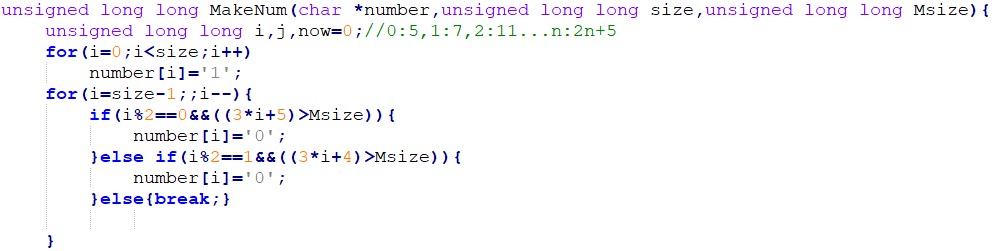
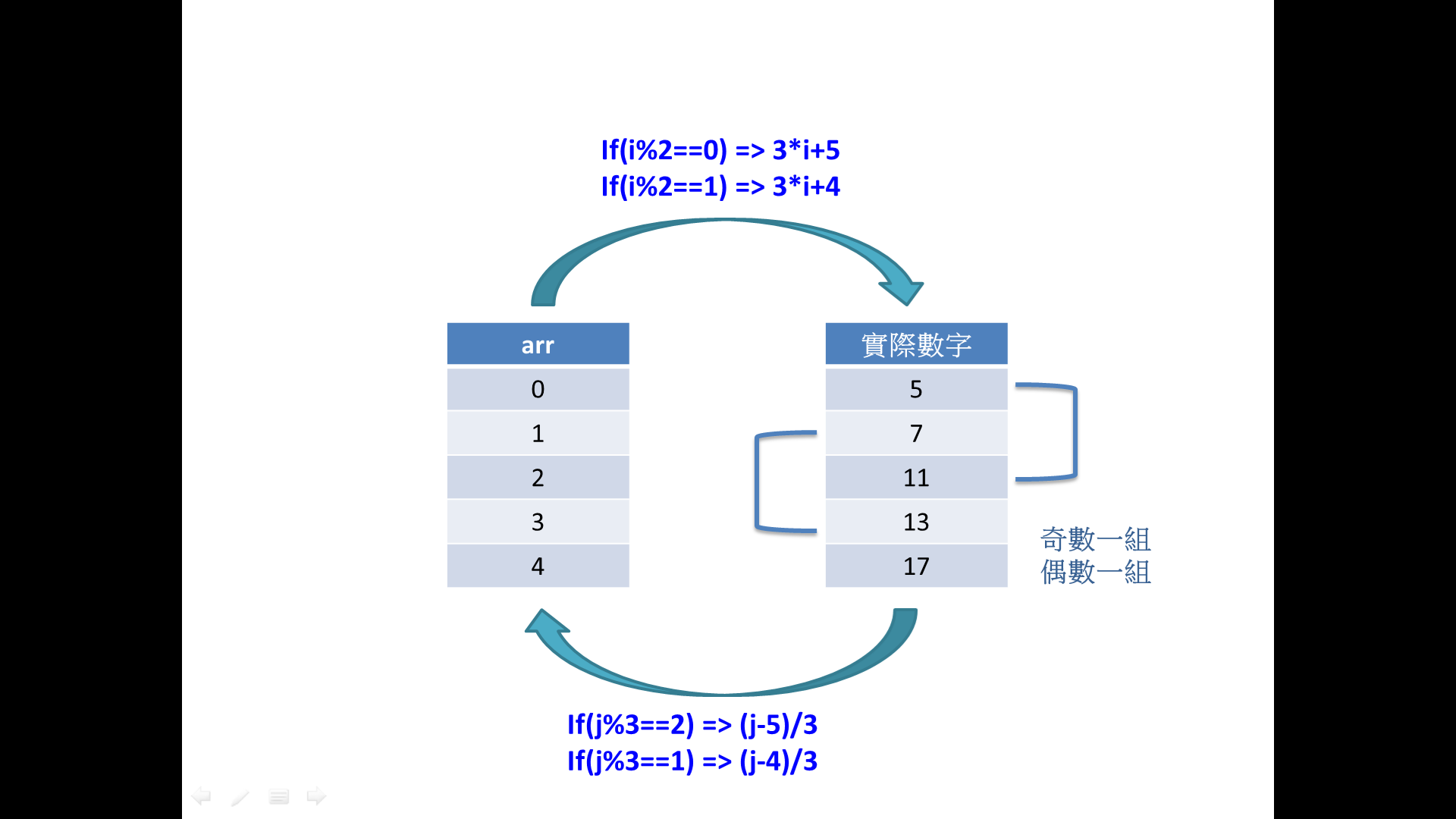


**[版本二]**是將陣列壓縮成幾乎是**[版本一]**的1/2倍，改存剛剛找出來的數列:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Index** | **0** | **1** | **2** | **3** | **……** | **n** |
| 值 | 5 | 7 | 11 | 13 | …… | 2n+5 |

n只要給到上一個版本的一半就好了，這時候可以存的質數變成320億以內了。

但覺得還可以再壓縮，所以又有了**[第三版]**:

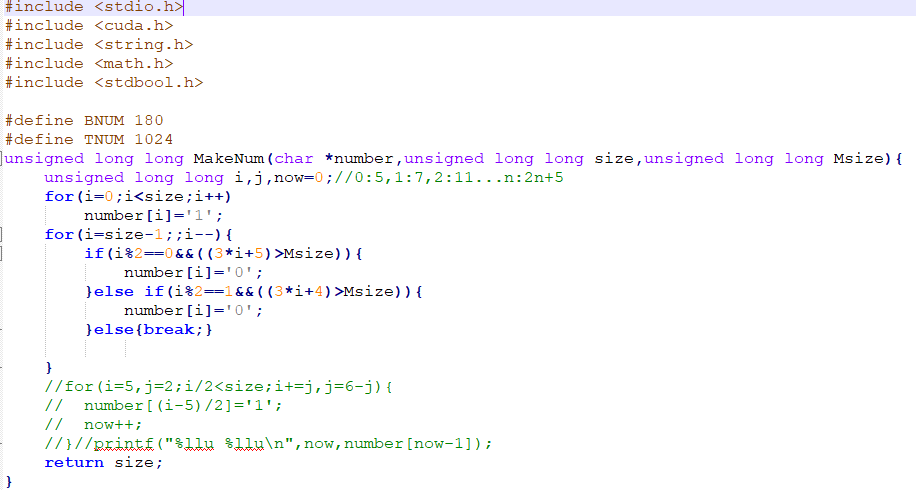


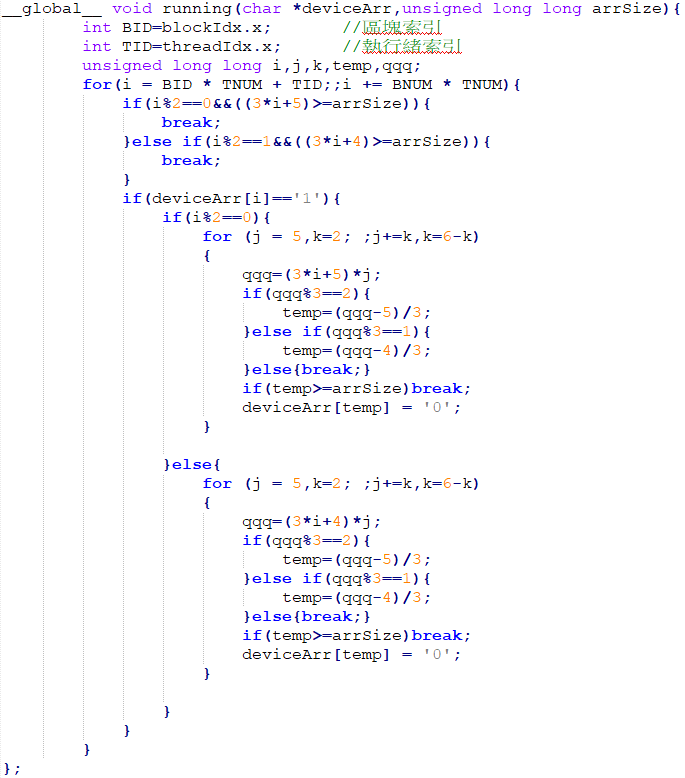
而這個方法比**[第一版]**少了將近1/3的空間，所以可以存到480億。

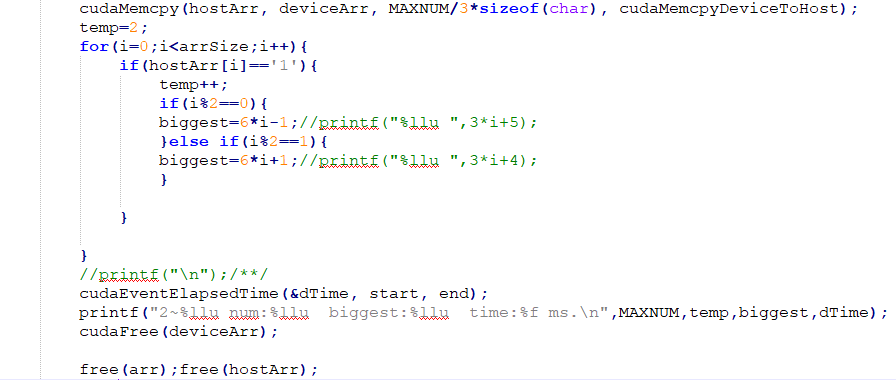
但是，最後因為記憶體受限的關係，所以我們找最大值的方法，決定用unsigned long long可以表示的最大值往下找，只需要有最大值開根號後以下的質數表即可找到unsigned long long以內最大的質數。

因為我們前面三個版本是將先前質數全部存下來，但只能找到480億以內最大質數，如果改成只需要開根號以內的質數表，我們***其實可以找到480億\*480億以內的最大質數***。不過受限於unsigned long long的大小，我們最後只用了50億以內的質數表，就可以找到 ***1千8百多京*** 以內的最大值數了。

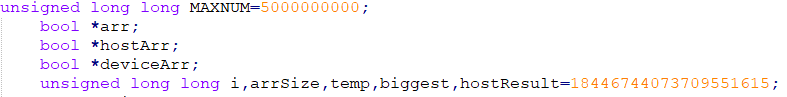
三、程式碼

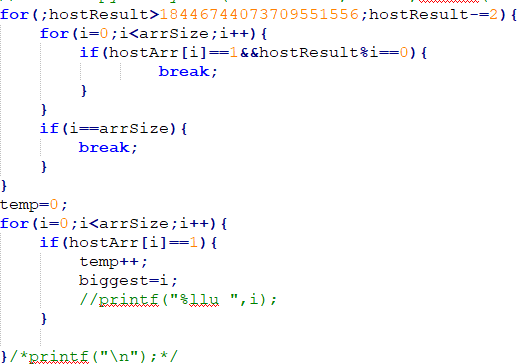
以下是**[版本三]**的程式碼





Q2找最大質數的程式碼跟上述差不多，就是只需要50億以內的質數，就可以了。





四、結果

Q1:

小於1,000,000,000的質數

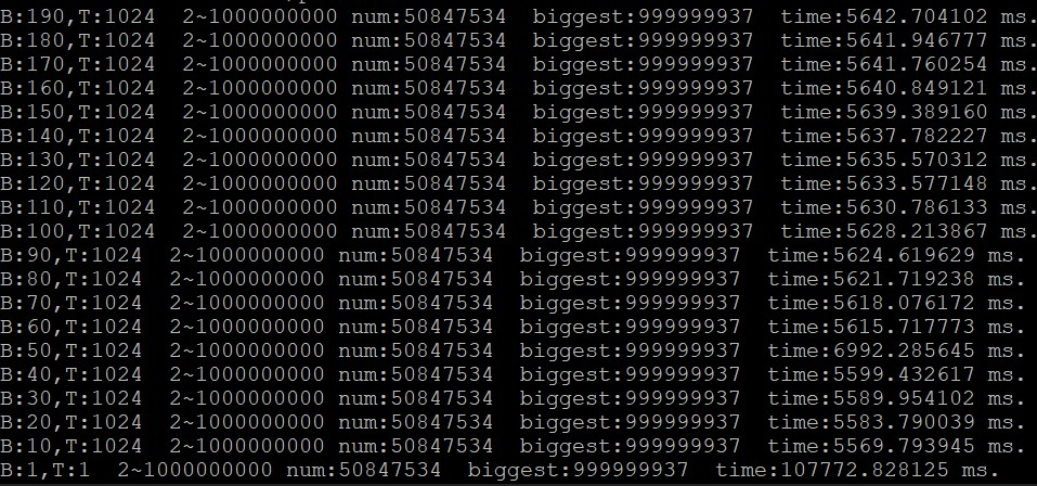
總共：50,847,534個質數

最大：999,999,937

最佳解：

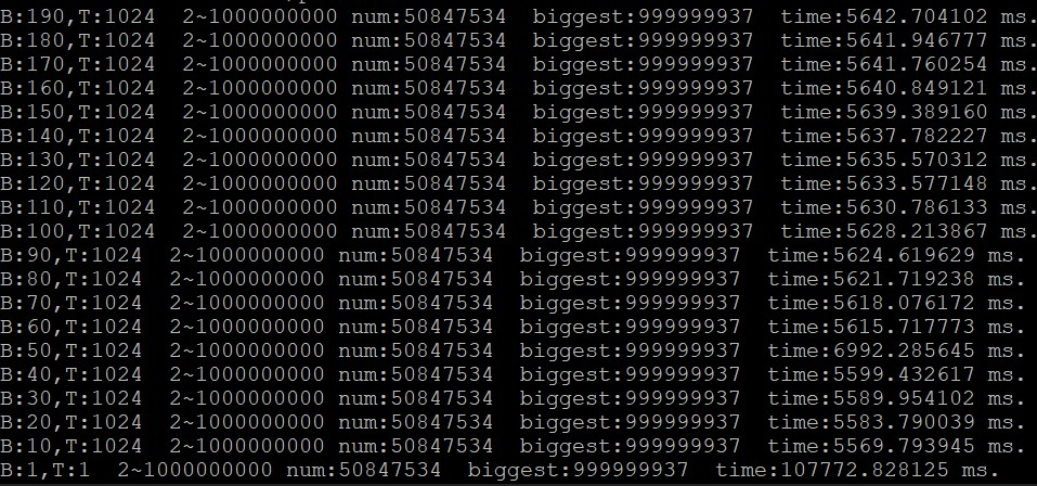
<BNUM=140 TNUM=1024>

Time: 5637.782227 ms



<BNUM=1 TNUM=1>

Time: 107772.828125 ms



Speedup：19.11

Q2:

範圍：2～18,446,744,073,709,551,615

最大：***18,446,744,073,709,551,557***



五、效能分析

MPI\_Speedup

|  |  |
| --- | --- |
| ThreadNum | MPI\_Speedup |
| P2 | 2.08 |
| P4 | 4.13 |
| P8 | 7.85 |

OpenMP\_Speedup

|  |  |
| --- | --- |
| ThreadNum | OpenMP\_Speedup |
| P2 | 1.4 |
| P4 | 2.93 |
| P8 | 5.24 |
| P16 | 9.74 |
| P32 | 15.54 |
| P48 | 20.93 |

比較各個方式的最高Speedup

|  |  |
| --- | --- |
|  | Best Speedup |
| MPI | 7.85 |
| OpenMP | 20.93 |
| CUDA | 19.11 |

七、心得

陳蕾：

這個實驗做了好久阿，第一題還好，只是一開始沒有優化程式碼所以跑了滿久的，而且花最多時間的是server常常被操壞掉，因為有同學寫的時候宣告了大量的Globol memory，所以整個host端的記憶體完全不足(99%)，不是device空間的問題。

還有另一個大問題就是一直被鎖IP，宿舍的網路因為使用太多SSH連線直接被切掉，然後連手機的網路，手機直接被鎖IP，好像是router鎖住的，因為連課程網站都沒有辦法連上，哀，反正真的很多問題。後來開了新的server，沒有多久，下查看的指令就發現8張卡6張壞掉了……。

再來是我們以為要將所有的找到的質數儲存起來，所以一直在解決device記憶體的問題，所以將程式碼優化再優化，到第三版可以達到第一版總儲存量的1/3。沒想到居然不用儲存，那其實第一版就可以找到可以顯示的最大質數，unsigned long long的最大數是18,446,744,073,709,551,615，其實只要有0~4,294,967,296的質數表就可以了……而我們之前的方法已經可以建到48,000,000,000的質數表，整整多了10倍。所以改了一下程式碼，就直接很快地找到了可顯示得最大質數18,446,744,073,709,551,557(1844京)。