

Tính toán song song

Nhóm 5 - Lớp: 137959

Giảng viên hướng dẫn: Thầy Đoàn Duy Trung

Sinh viên thực hiện:

Nguyễn Thị Duyên	20195866
Trần Thị Hồng	20195880
Nguyễn Lan Hương	20195884
Nguyễn Như Thuận	20195925
Trần Thị Hồng Vân	20195941



Mục lục

Lời nói đầu	2
I Bảng kết quả	4
1 Bài thực hành chung	4
1.1 Liệt kê các số chính phương	4
1.2 Tính gần đúng tích phân xác định	12
2 Bài tập nhóm riêng	24
2.1 Kết quả chạy chương trình	24
3 Bài tập thêm	32
3.1 Kết quả chạy chương trình Liệt kê số Fibonacci	32
II Mã nguồn và Thuật toán	36
1 Bài thực hành chung	36
1.1 Liệt kê các số chính phương	36
1.2 Tính gần đúng tích phân xác định	38
2 Bài tập nhóm riêng	40
2.1 Sơ đồ thuật toán	40
2.2 Mã nguồn	43
3 Bài tập thêm	46
3.1 Thuật toán	46
3.2 Mã nguồn	48
III Nhận xét	50
Tài liệu tham khảo	51

Lời nói đầu

Tính toán song song là một học phần quan trọng trong ngành công nghệ thông tin. Công nghệ và kĩ thuật ngày càng phát triển, dữ liệu ngày càng lớn đòi hỏi cần phải có một kĩ thuật tính toán giúp cho việc đưa ra kết quả được nhanh hơn, có ý nghĩa về mặt thời gian, tận dụng được tài nguyên phi cục bộ, tiết kiệm chi phí và vượt ra ngoài được giới hạn bộ nhớ của máy tính, ... Từ đó đã dẫn đến sự ra đời của kĩ thuật tính toán song song và điển hình đó là kĩ thuật tính toán lưới (Grid Computing).

Tính toán lưới là việc sử dụng các tài nguyên máy tính được phân phối rộng rãi để đạt được một mục tiêu chung. một lưới tính toán, có thể được coi là một hệ thống phân tán với những công việc độc lập với nhau. Việc sử dụng Kỹ thuật tính toán lưới là rất quan trọng, đáp ứng được nhu cầu tích hợp tài nguyên, nhu cầu về tính thông lượng cao, khả năng tận dụng tài nguyên nhàn rỗi và sự cộng tác giữa các tổ chức. Một ứng dụng sử dụng tính toán lưới sẽ giúp cho việc thực thi chương trình được nhanh hơn, tiết kiệm thời gian và tài nguyên, có thể sử lý song song, cân bằng tài nguyên và quản lý được các dữ liệu không đồng nhất độc lập với nhau. Tuy nhiên, việc sử dụng và điều phối các ứng dụng trên lưới có thể là một nhiệm vụ phức tạp, đặc biệt là khi điều phối luồng thông tin trên các tài nguyên máy tính phân tán.

Chúng ta có thể lập trình tính toán lưới trên nhiều nền tảng khác nhau. Và trong phạm vi của bài báo cáo này, chúng em đã lập trình tnhs toán lưới trên nền tảng Alchemi sử dụng ngôn ngữ lập trình C Sharp (C#). Nội dung của bài báo cáo bao gồm quá trình chạy chương trình trên Alchemi, kết quả chạy chương trình và mã nguồn của các chương trình sau:

- Bài thực hành chung
 - Liệt kê các số chính phương
 - Tính gần đúng tích phân xác định
- Bài tập nhóm riêng
 - Tổng hợp vị trí cụm từ SAMI
- Bài tập thêm
 - Liệt kê dãy Fibonaci

Để có thể hoàn thành bài báo cáo này, em xin được gửi lời cảm ơn chân thành và sâu sắc đến thầy **TS. Đoàn Duy Trung**, thầy đã tận tình giảng dạy và hướng dẫn em trong suốt quá trình học tập và làm bài báo cáo.

Bảng phân chia công việc

Tên thành viên	Nhiệm vụ
Nguyễn Thị Duyên	Code bài tập thêm Viết thuật toán Bài tập thêm Hỗ trợ code Bài tập riêng
Trần Thị Hồng	Code bài tập riêng Viết chi tiết thuật toán Bài tập riêng
Nguyễn Lan Hương	Code Bài tập chung 1 Chạy kết quả Hỗ trợ Báo cáo
Nguyễn Như Thuận	Code Bài tập chung 2 Chạy kết quả Trình bày.
Trần Thị Hồng Vân	Làm báo cáo Hỗ trợ Code Bài tập chung Chạy kết quả

1 Bài thực hành chung

- Ở phần bài tập chung khi thực hiện chúng em sử dụng Hamachi tạo ra một mạng LAN ảo và các bạn Join vào mạng ảo này chính vì vậy thời gian chạy thu được lâu.
- Phần bài tập riêng chúng em đã tập trung và cùng chạy trên một mạng Wi-Fi nên không cần tạo ra mạng ảo. Điều đó làm giảm độ trễ khi các Executor gửi kết quả tính toán về Manager.

1.1 Liệt kê các số chính phương

Liệt kê các số chính phương từ 1 đến n , với n nhập từ bàn phím. Phân chia đoạn $[1; n]$ thành các đoạn để chạy trên các Executor trong Lưới.

Bảng 1: Kết quả chạy chương trình liệt kê các số chính phương

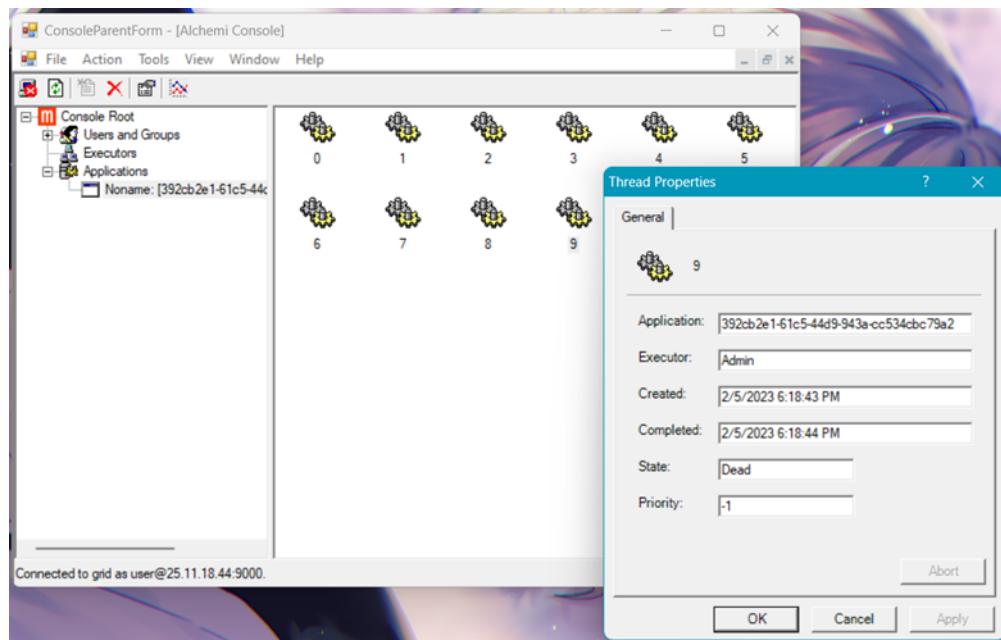
STT	Số nút	Số nguyên n	Số luồng	Thời gian (s)
1	2	1 000	10	13.3900231
2	2	2 000	10	16.2479310
3	3	1 000	10	13.0386316
4	3	2 000	10	14.4945216

Ảnh chụp màn hình khi chạy chương trình

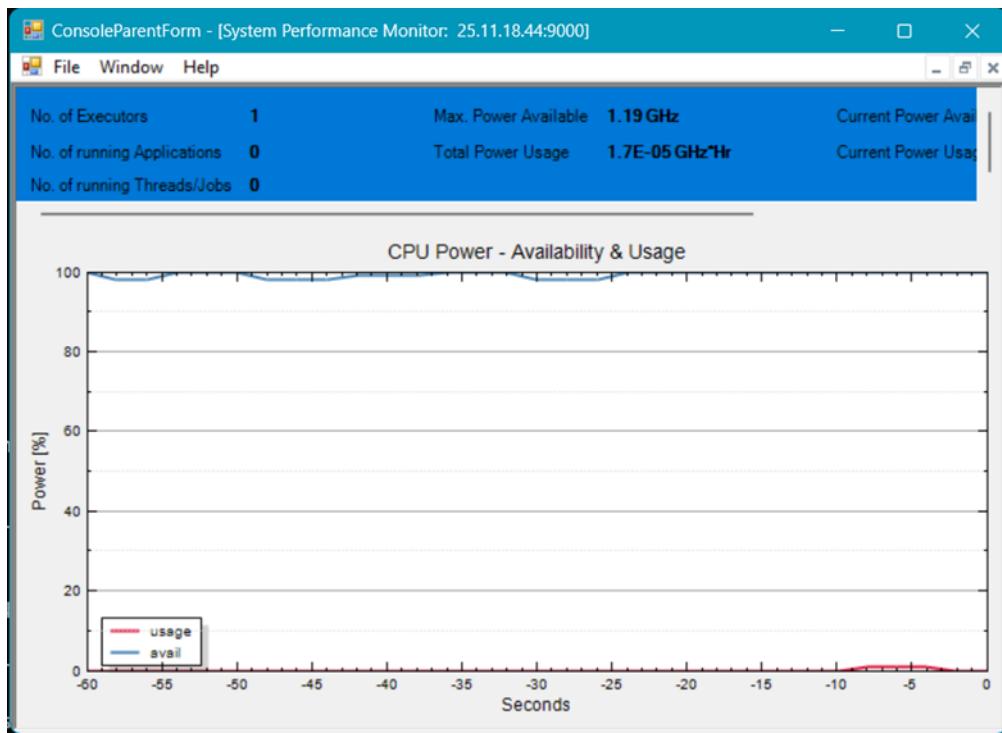
- 2 nút, n = 1000, 10 luồng:

```
C:\Users\Admin\Desktop\Prin > + - 
The numbers from 301 to 400 have been checked.
The perfect squares are: 324, 361, 400,
The numbers from 401 to 500 have been checked.
The perfect squares are: 441, 484,
The numbers from 501 to 600 have been checked.
The perfect squares are: 529, 576,
The numbers from 601 to 700 have been checked.
The perfect squares are: 625, 676,
The numbers from 701 to 800 have been checked.
The perfect squares are: 729, 784,
The numbers from 801 to 900 have been checked.
The perfect squares are: 841, 900,
The numbers from 901 to 1000 have been checked.
The perfect squares are: 961,
-----
The list of perfect squares is: 1, 4, 9, 16, 25, 36, 49, 64, 81, 100, 121, 144, 169, 196, 225, 256, 289, 324, 361, 400, 441, 484, 529, 576, 625, 676, 729, 784, 841, 900, 961,
-----
The program execution time is (seconds): 00:00:13.3900231
```

Hình 1: Kết quả chạy chương trình liệt kê các số chính phương



Hình 2: Các luồng chương trình đã chạy hiển thị trên Alchemi Console



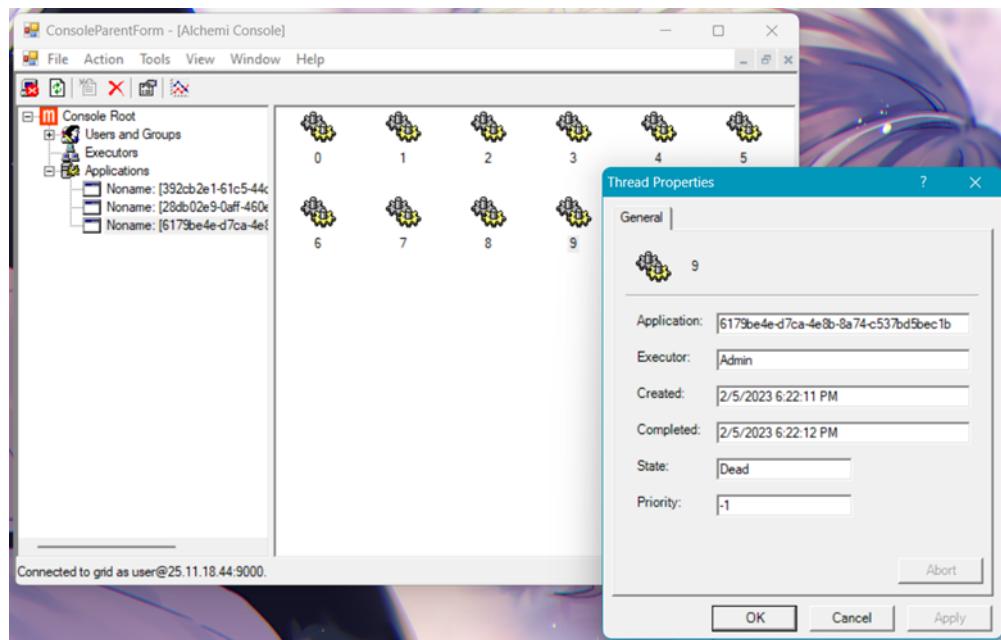
Hình 3: CPU Power - Availability & Usage

- 2 nút, n = 2000, 10 luồng:

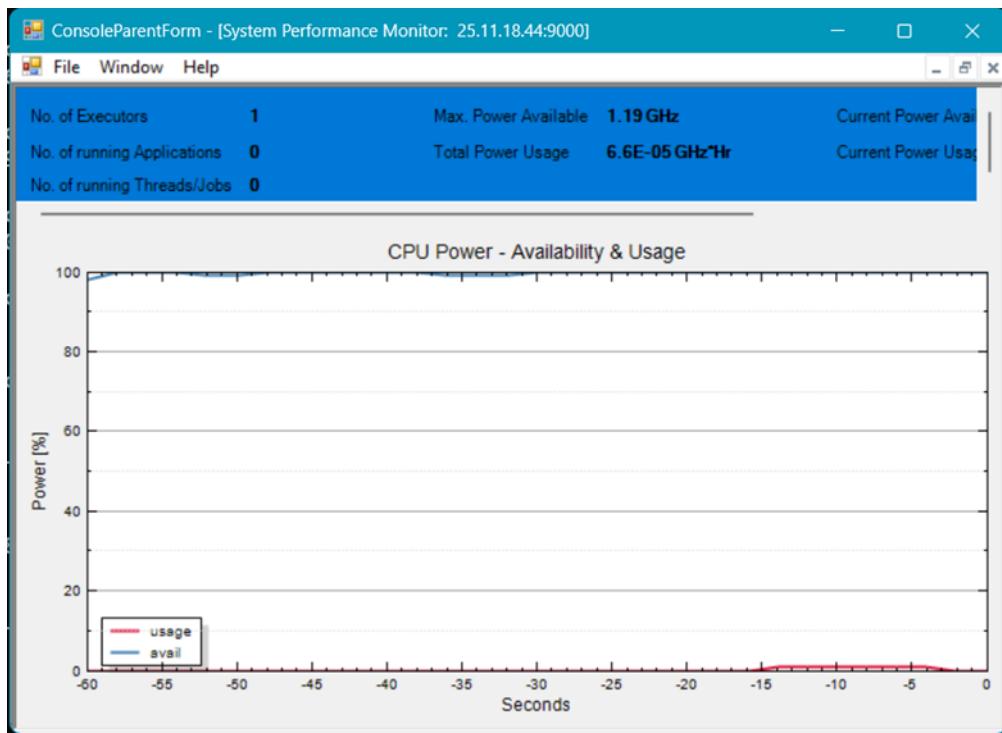
```

C:\Users\Admin\Desktop\Prin > + -
The perfect squares are: 625, 676, 729, 784,
The numbers from 801 to 1000 have been checked.
The perfect squares are: 841, 900, 961,
The numbers from 1001 to 1200 have been checked.
The perfect squares are: 1024, 1089, 1156,
The numbers from 1201 to 1400 have been checked.
The perfect squares are: 1225, 1296, 1369,
The numbers from 1401 to 1600 have been checked.
The perfect squares are: 1444, 1521, 1600,
The numbers from 1601 to 1800 have been checked.
The perfect squares are: 1681, 1764,
The numbers from 1801 to 2000 have been checked.
The perfect squares are: 1849, 1936,
-----
The list of perfect squares is: 1, 4, 9, 16, 25, 36, 49, 64, 81, 100, 121, 144, 169, 196, 225, 256, 289, 324, 361, 400, 441, 484, 529, 576, 625, 676, 729, 784, 841, 900, 961, 1024, 1089, 1156, 1225, 1296, 1369, 1444, 1521, 1600, 1681, 1764, 1849, 1936,
-----
The program execution time is (seconds): 00:00:16.2479310
|
```

Hình 4: Kết quả chạy chương trình liệt kê các số chính phương



Hình 5: Các luồng chương trình đã chạy hiển thị trên Alchemi Console



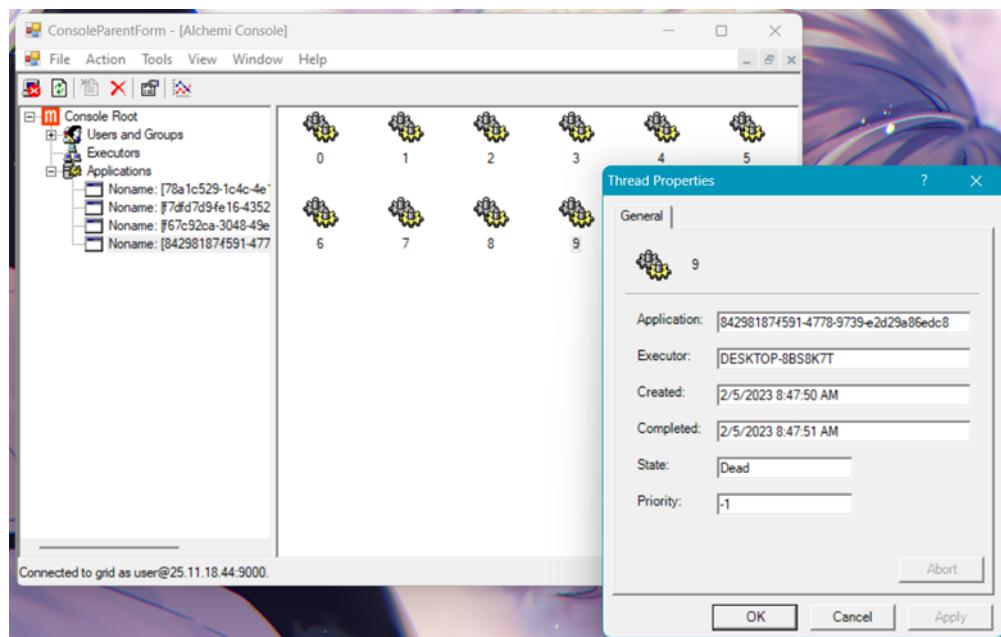
Hình 6: CPU Power - Availability & Usage

- 3 nút, n = 1000, 10 luồng:

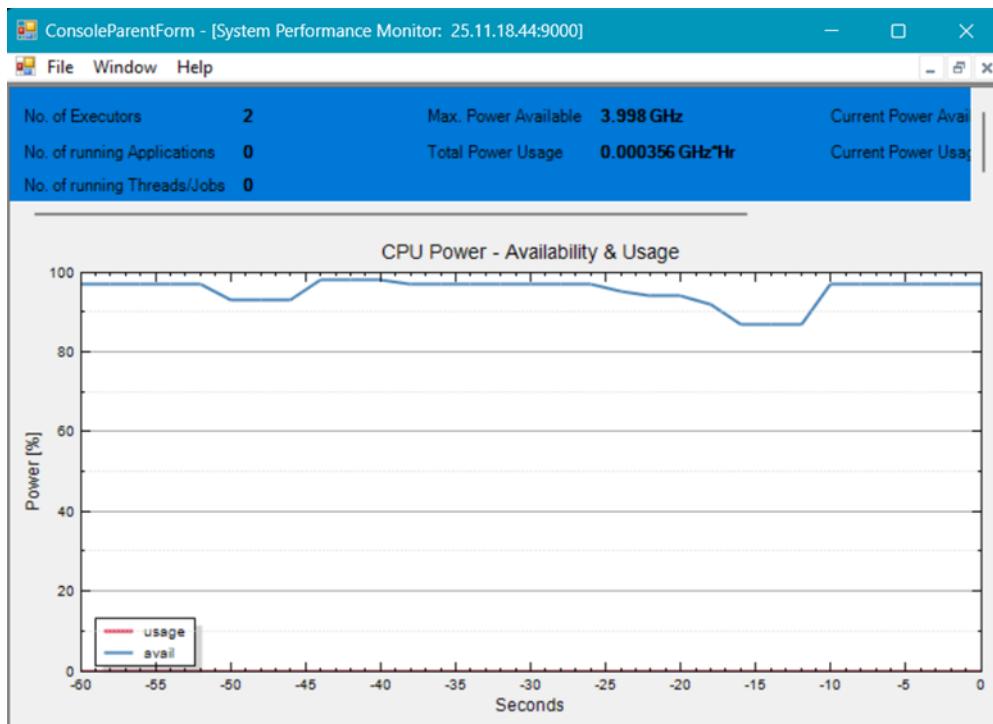
```

C:\Users\Admin\Desktop\Prin > + -
The numbers from 401 to 500 have been checked.
The perfect squares are: 441, 484,
The numbers from 501 to 600 have been checked.
The perfect squares are: 529, 576,
The numbers from 601 to 700 have been checked.
The perfect squares are: 625, 676,
The numbers from 701 to 800 have been checked.
The perfect squares are: 729, 784,
The numbers from 1 to 100 have been checked.
The perfect squares are: 1, 4, 9, 16, 25, 36, 49, 64, 81, 100,
The numbers from 801 to 900 have been checked.
The perfect squares are: 841, 900,
The numbers from 901 to 1000 have been checked.
The perfect squares are: 961,
-----
The list of perfect squares is: 121, 144, 169, 196, 225, 256, 289, 324, 361, 400, 441, 484, 529, 576, 625, 676, 729, 784
, 1, 4, 9, 16, 25, 36, 49, 64, 81, 100, 841, 900, 961,
-----
The program execution time is (seconds): 00:00:13.0386361
|
```

Hình 7: Kết quả chạy chương trình liệt kê các số chính phương



Hình 8: Các luồng chương trình đã chạy hiển thị trên Alchemi Console



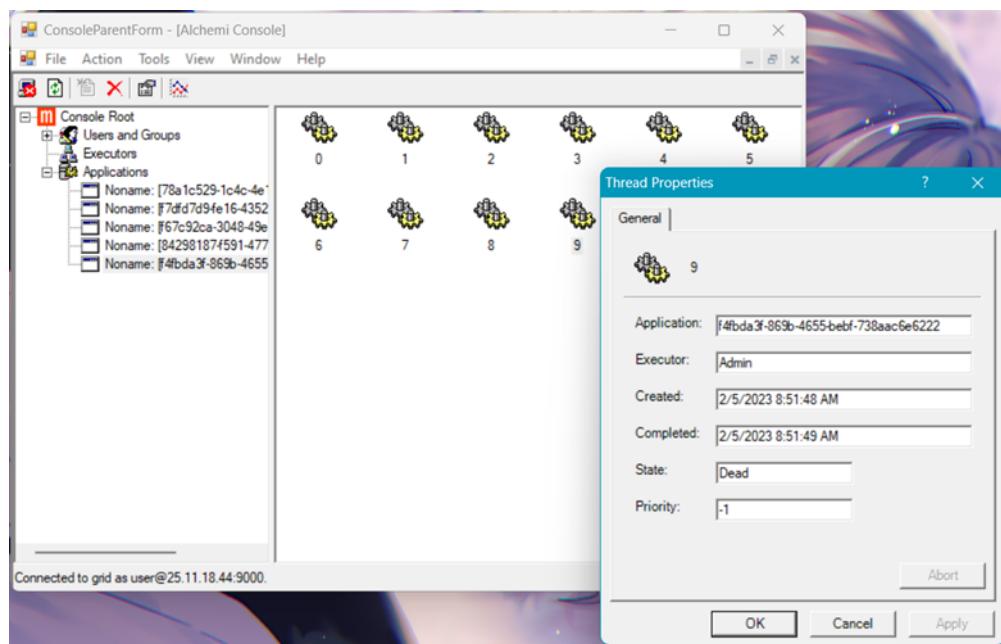
Hình 9: CPU Power - Availability & Usage

- 3 nút, n = 2000, 10 luồng:

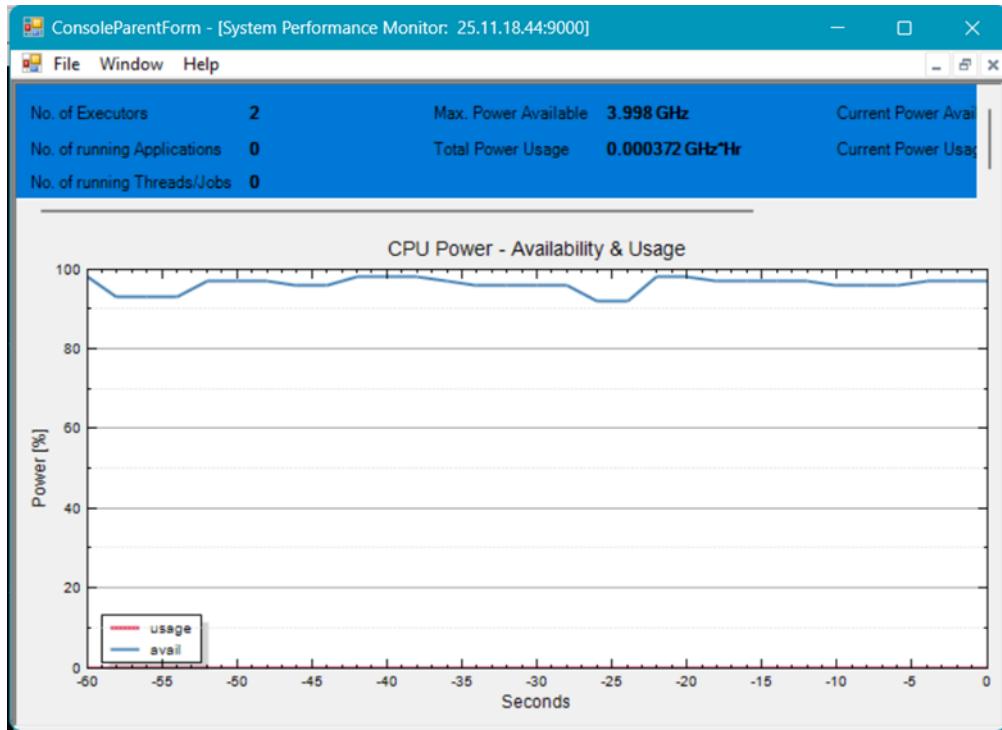
```

C:\Users\Admin\Desktop\Prin > + - 
The perfect squares are: 841, 900, 961,
The numbers from 1001 to 1200 have been checked.
The perfect squares are: 1024, 1089, 1156,
The numbers from 1201 to 1400 have been checked.
The perfect squares are: 1225, 1296, 1369,
The numbers from 1401 to 1600 have been checked.
The perfect squares are: 1444, 1521, 1600,
The numbers from 1601 to 1800 have been checked.
The perfect squares are: 1681, 1764,
The numbers from 201 to 400 have been checked.
The perfect squares are: 225, 256, 289, 324, 361, 400,
The numbers from 1801 to 2000 have been checked.
The perfect squares are: 1849, 1936,
-----
The list of perfect squares is: 1, 4, 9, 16, 25, 36, 49, 64, 81, 100, 121, 144, 169, 196, 441, 484, 529, 576, 625, 676, 729, 784, 841, 900, 961, 1024, 1089, 1156, 1225, 1296, 1369, 1444, 1521, 1600, 1681, 1764, 225, 256, 289, 324, 361, 400, 1849, 1936,
-----
The program execution time is (seconds): 00:00:14.4945216
|
```

Hình 10: Kết quả chạy chương trình liệt kê các số chính phương



Hình 11: Các luồng chương trình đã chạy hiển thị trên Alchemi Console



Hình 12: CPU Power - Availability & Usage

1.2 Tính gần đúng tích phân xác định

Cho n nhập từ bàn phím. Tính gần đúng tích phân sau:

$$\int_0^n f(x)dx$$

Ở đó hàm $f(x)$ tuỳ ý.

Phân chia $[0; n]$ vào các Executor để chạy trong Lưới.

Chúng em lựa chọn tính tích phân hàm $f(x) = x^2$ từ 0 đến 10 bằng phương pháp hình thang với số đoạn được chia trong mỗi tích phân thành phần là n và độ dài mỗi đoạn thành phần là 1.

Bảng 2: Kết quả chạy chương trình tính tích phân xác định

STT	Số nút	Số nguyên n	Số luồng	Thời gian (s)
1	2	100 000	10	16.2558739
2	2	200 000	10	15.4882708
3	2	300 000	10	16.1801615
4	3	100 000	10	10.2602318
5	3	200 000	10	11.6283737
6	3	300 000	10	10.8417074

Ảnh chụp màn hình khi chạy chương trình

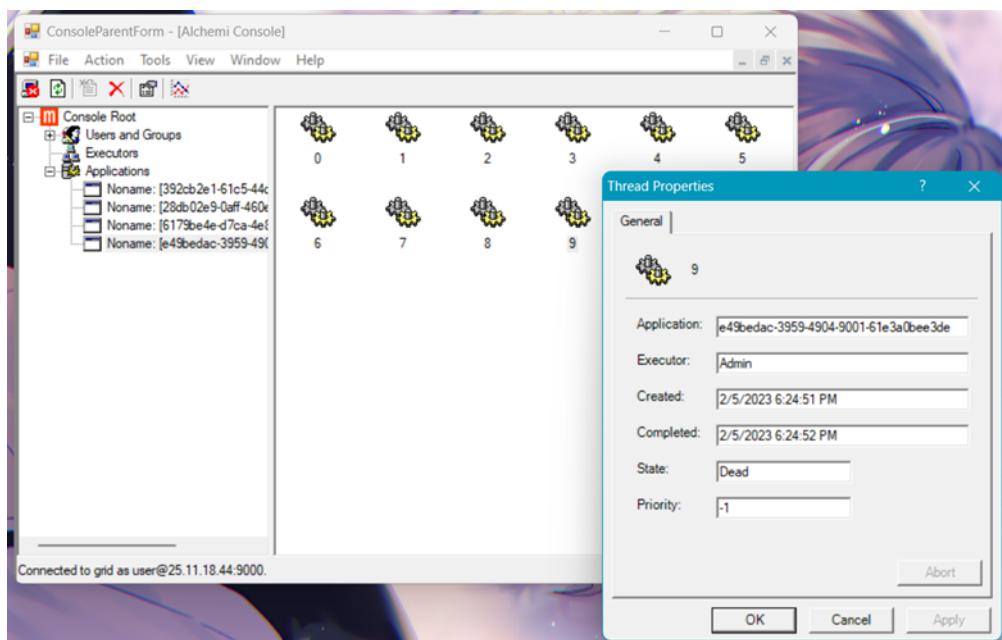
- 2 nút, n = 100 000, 10 luồng:

```

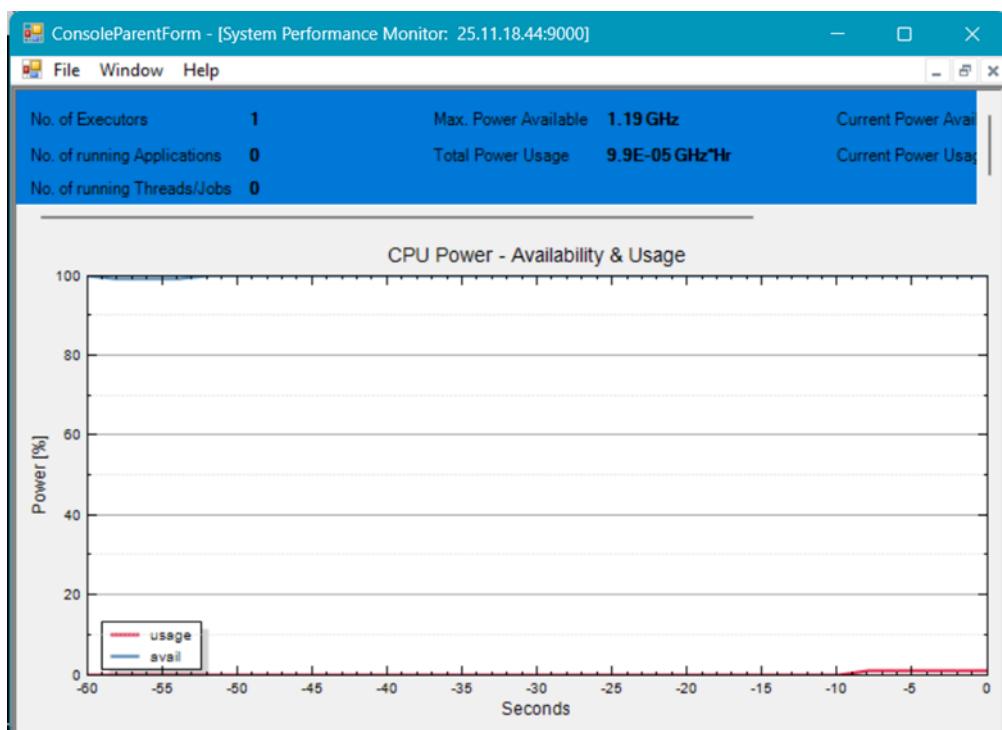
C:\Users\Admin\Desktop\Prin
The value on the division is: 7.33308333500001
The division from 3 to 3.9999 has been calculated.
The value on the division is: 13.3329833349999
The division from 4 to 4.9999 has been calculated.
The value on the division is: 21.3328833350001
The division from 5 to 5.9999 has been calculated.
The value on the division is: 31.3327833350001
The division from 6 to 6.9999 has been calculated.
The value on the division is: 43.3326833349999
The division from 7 to 7.9999 has been calculated.
The value on the division is: 57.3325833350001
The division from 8 to 8.9999 has been calculated.
The value on the division is: 73.3324833350001
The division from 9 to 10 has been calculated.
The value on the division is: 91.342483335

-----
Integral value is: 343.33843335
The program execution time is (seconds): 00:00:16.2558739
|
```

Hình 13: Kết quả chạy chương trình tính tích phân xác định



Hình 14: Các luồng chương trình đã chạy hiển thị trên Alchemi Console



Hình 15: CPU Power - Availability & Usage

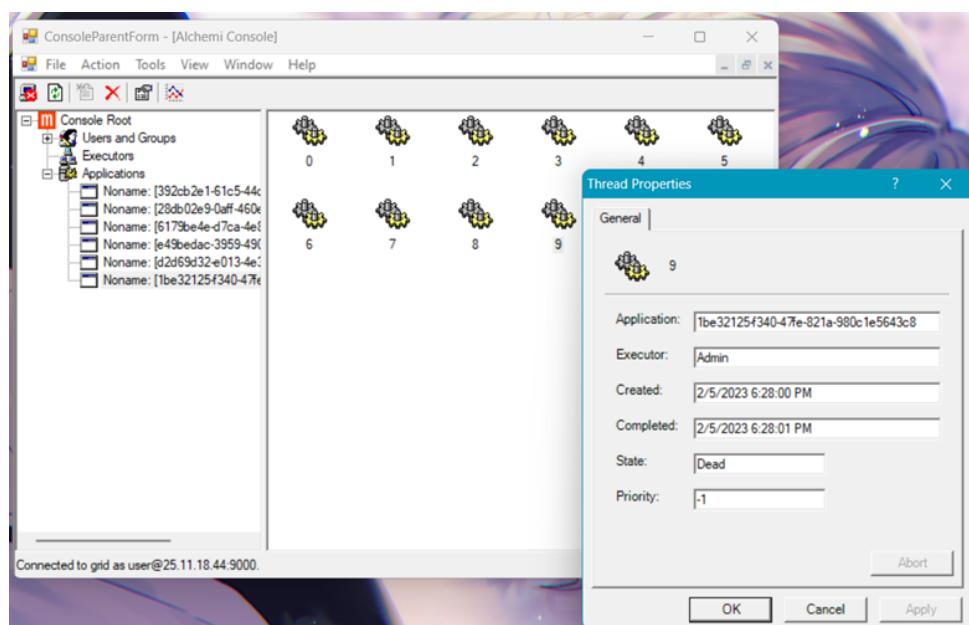
- 2 nút, $n = 200\ 000$, 10 luồng:

```

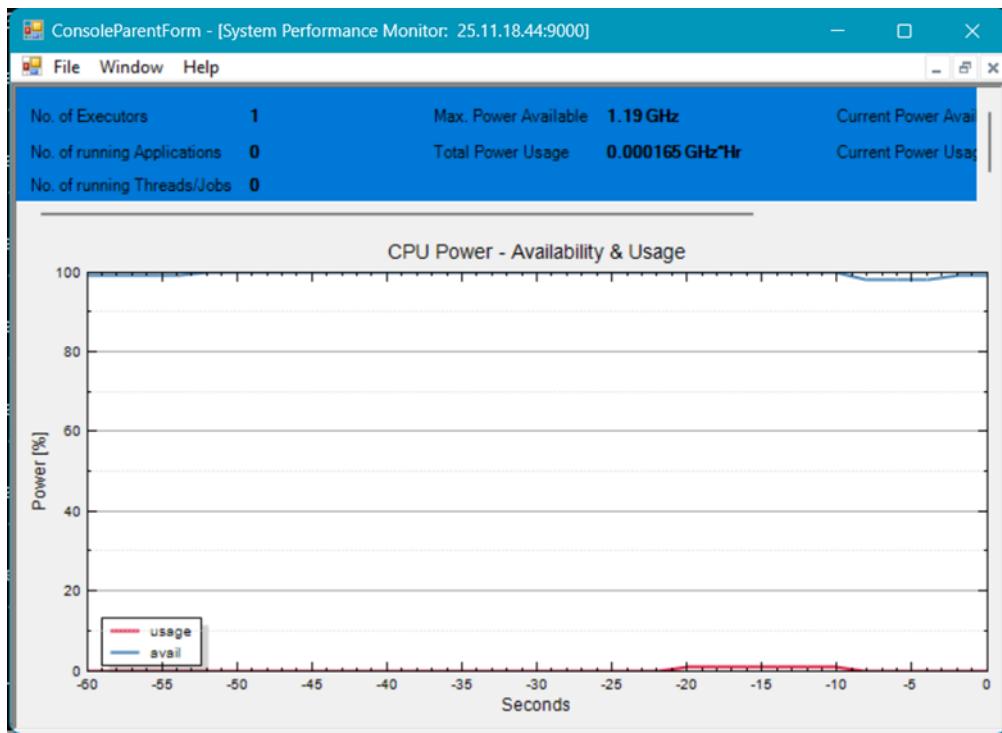
C:\Users\Admin\Desktop\Prin > + -
The value on the division is: 7.33320833375002
The division from 3 to 3.99995 has been calculated.
The value on the division is: 13.33315833375
The division from 4 to 4.99995 has been calculated.
The value on the division is: 21.3331683337499
The division from 5 to 5.99995 has been calculated.
The value on the division is: 31.33305833375
The division from 6 to 6.99995 has been calculated.
The value on the division is: 43.3330083337501
The division from 7 to 7.99995 has been calculated.
The value on the division is: 57.3329583337501
The division from 8 to 8.99995 has been calculated.
The value on the division is: 73.3329083337504
The division from 9 to 10 has been calculated.
The value on the division is: 91.33790833375

-----
Integral value is: 343.335883337501
The program execution time is (seconds): 00:00:15.4882708
|
```

Hình 16: Kết quả chạy chương trình tính tích phân xác định



Hình 17: Các luồng chương trình đã chạy hiển thị trên Alchemi Console



Hình 18: CPU Power - Availability & Usage

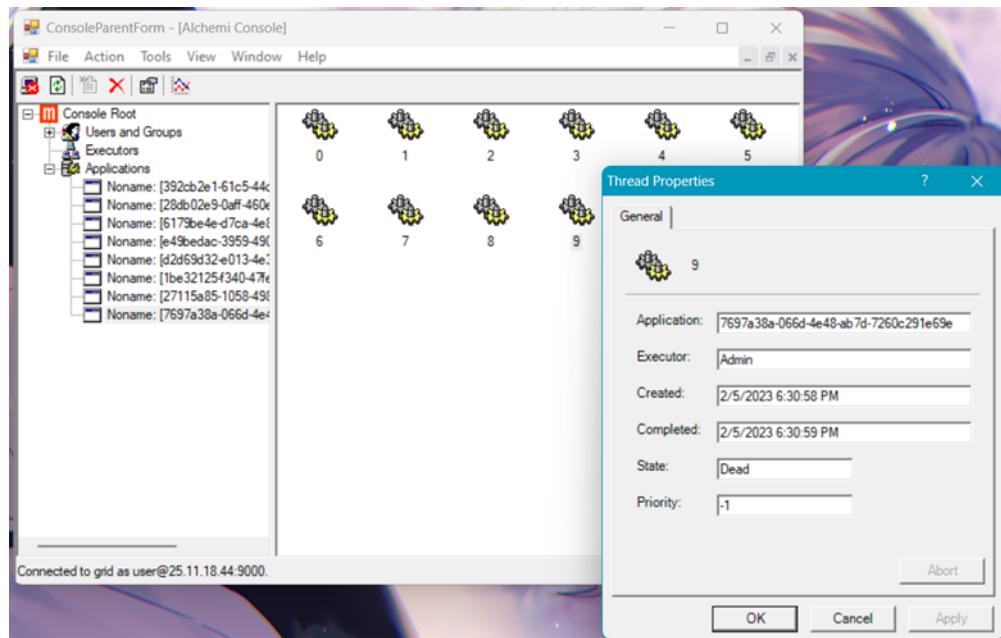
- 2 nút, $n = 300\ 000$, 10 luồng:

```

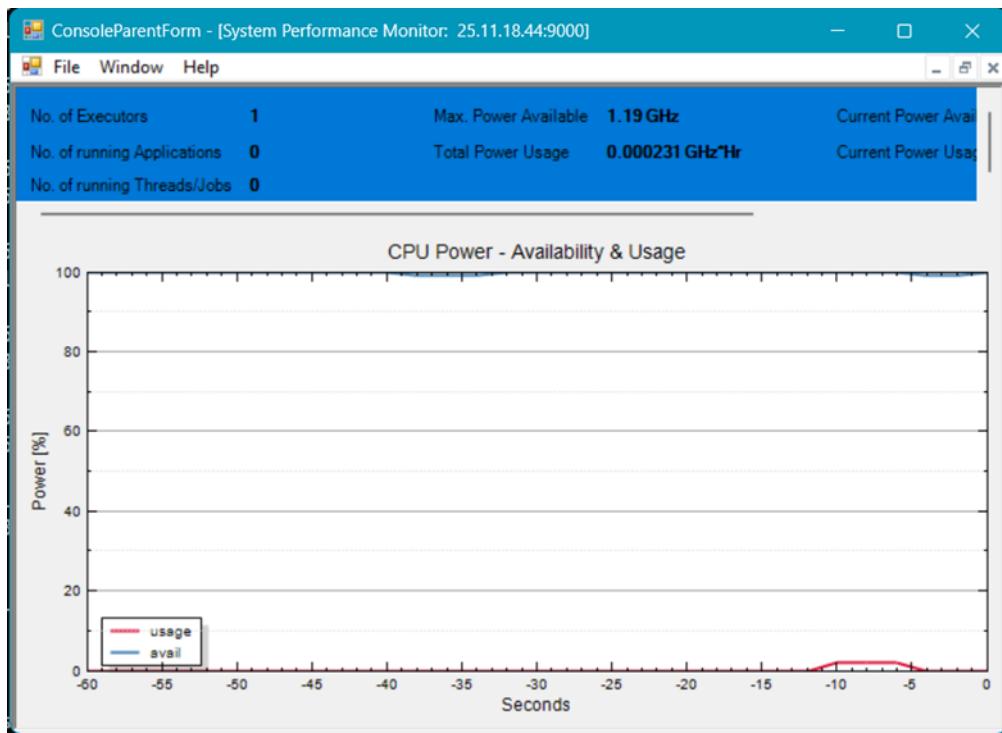
C:\Users\Admin\Desktop\Prin > + -
The value on the division is: 7.33325000018516
The division from 3 to 3.99996666666667 has been calculated.
The value on the division is: 13.3332166668518
The division from 4 to 4.99996666666667 has been calculated.
The value on the division is: 21.3331833335186
The division from 5 to 5.99996666666667 has been calculated.
The value on the division is: 31.3331500001852
The division from 6 to 6.99996666666667 has been calculated.
The value on the division is: 43.3331166668519
The division from 7 to 7.99996666666667 has been calculated.
The value on the division is: 57.3330833335183
The division from 8 to 8.99996666666667 has been calculated.
The value on the division is: 73.3330500001849
The division from 9 to 10 has been calculated.
The value on the division is: 91.3363833335187

-----
Integral value is: 343.335033335185
The program execution time is (seconds): 00:00:16.1801615
|
```

Hình 19: Kết quả chạy chương trình tính tích phân xác định



Hình 20: Các luồng chương trình đã chạy hiển thị trên Alchemi Console



Hình 21: CPU Power - Availability & Usage

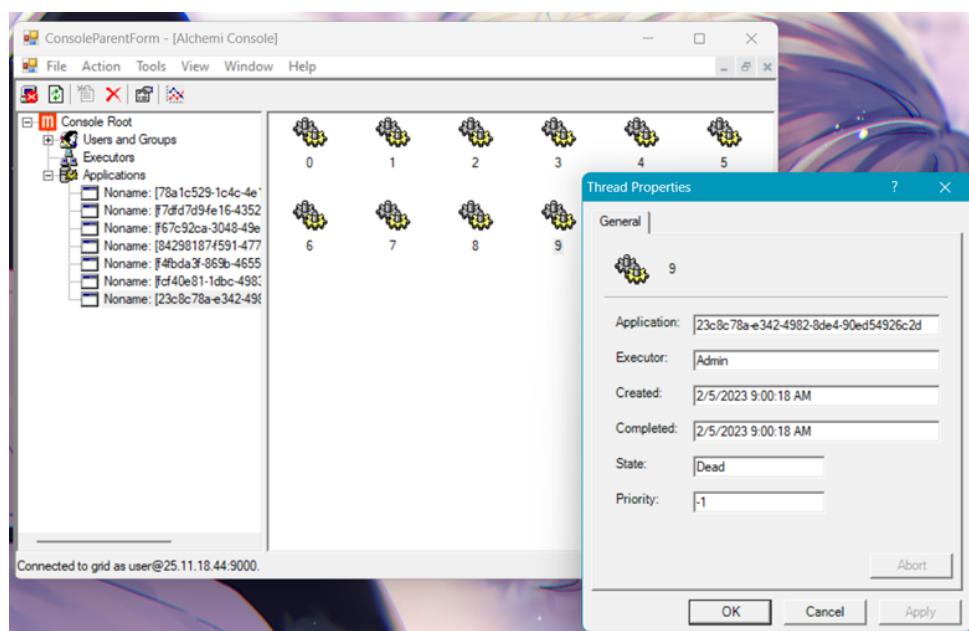
- 3 nút, $n = 100\ 000$, 10 luồng:

```

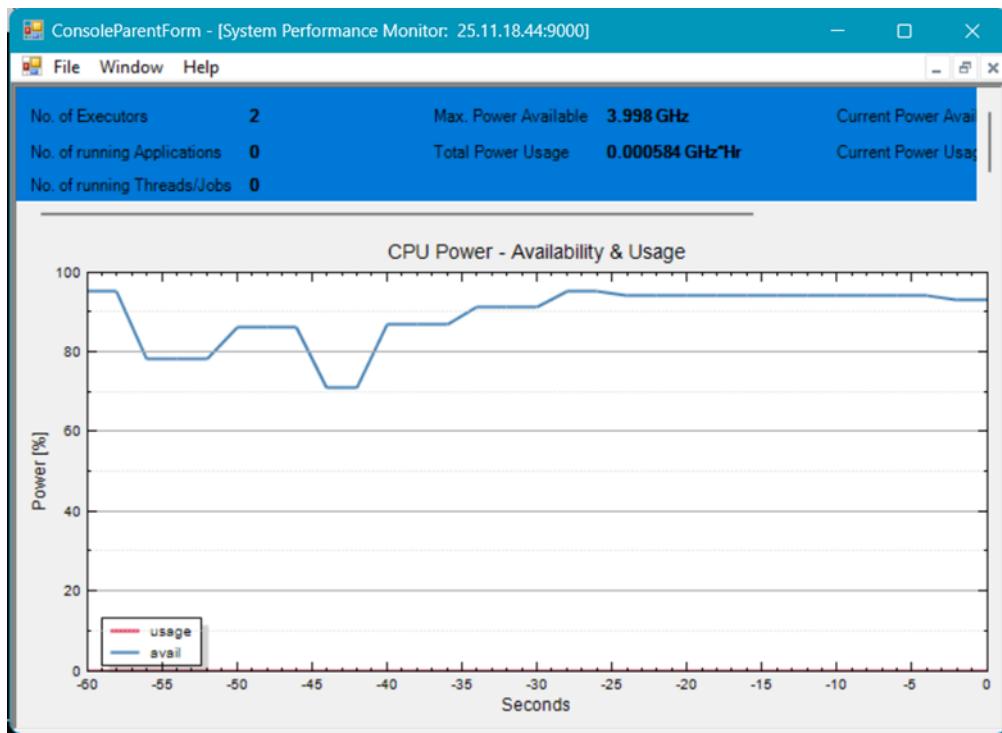
C:\Users\Admin\Desktop\Prin > + -
The value on the division is: 6.3330833335
The division from 3 to 3.9999 has been calculated.
The value on the division is: 12.3329833349999
The division from 5 to 5.9999 has been calculated.
The value on the division is: 30.3327833350001
The division from 4 to 4.9999 has been calculated.
The value on the division is: 20.3328833350001
The division from 6 to 6.9999 has been calculated.
The value on the division is: 42.3326833349999
The division from 7 to 7.9999 has been calculated.
The value on the division is: 56.3325833350001
The division from 8 to 8.9999 has been calculated.
The value on the division is: 72.3324833350002
The division from 9 to 10 has been calculated.
The value on the division is: 90.342383335

-----
Integral value is: 333.33833335
The program execution time is (seconds): 00:00:10.2602318
|
```

Hình 22: Kết quả chạy chương trình tính tích phân xác định



Hình 23: Các luồng chương trình đã chạy hiển thị trên Alchemi Console



Hình 24: CPU Power - Availability & Usage

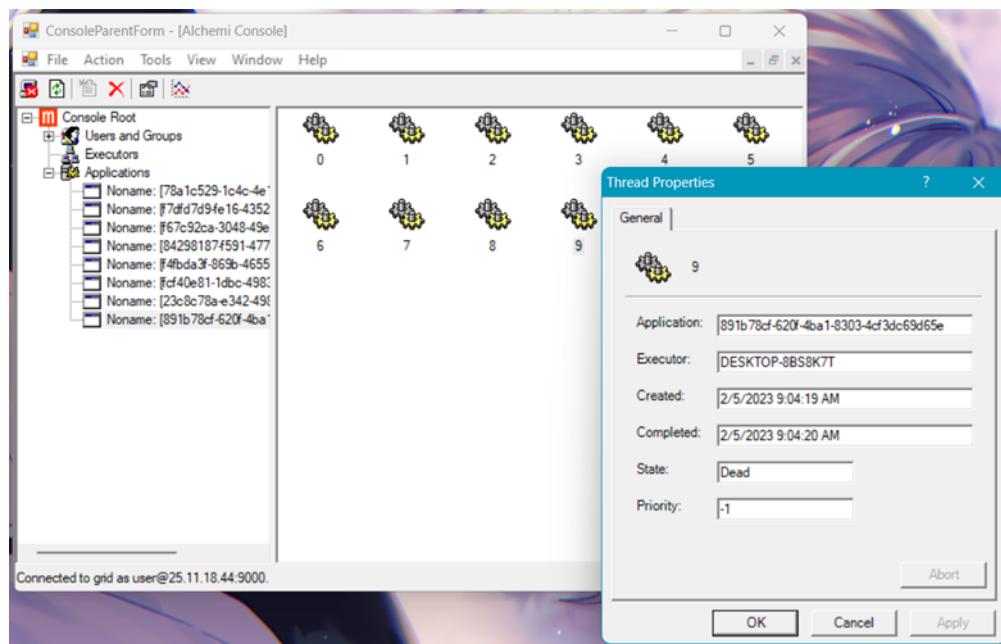
- 3 nút, $n = 200\ 000$, 10 luồng:

```

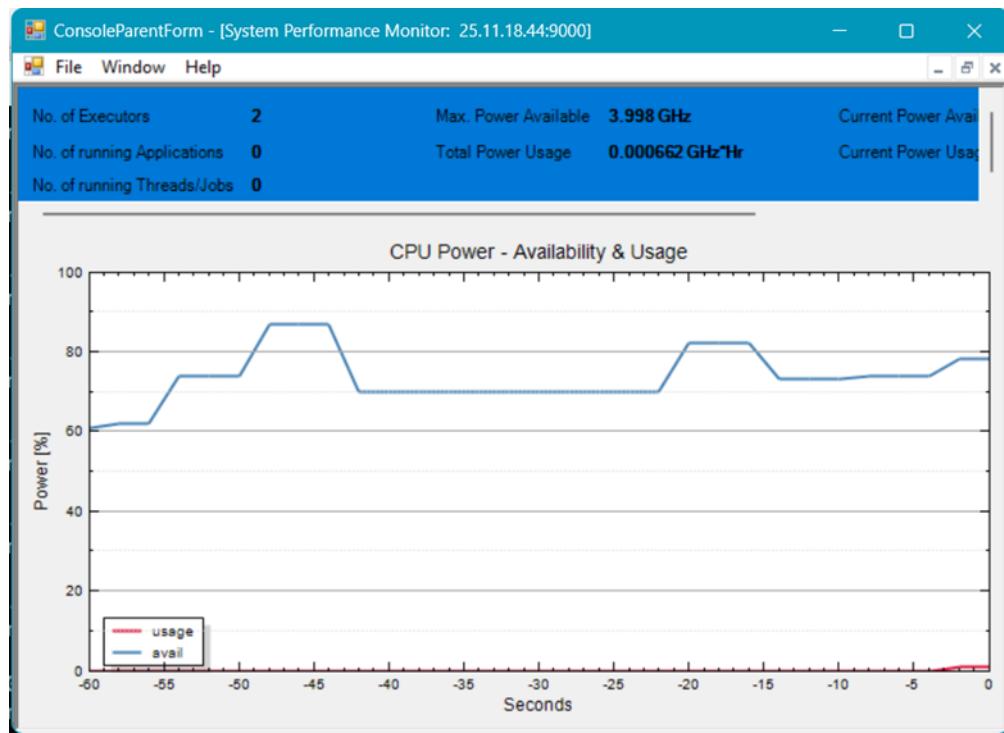
C:\Users\Admin\Desktop\Prin > + -
The value on the division is: 6.33320833375
The division from 3 to 3.99995 has been calculated.
The value on the division is: 12.33315833375
The division from 4 to 4.99995 has been calculated.
The value on the division is: 20.3331083337499
The division from 5 to 5.99995 has been calculated.
The value on the division is: 30.3330583337499
The division from 6 to 6.99995 has been calculated.
The value on the division is: 42.3330083337501
The division from 7 to 7.99995 has been calculated.
The value on the division is: 56.33295833375
The division from 8 to 8.99995 has been calculated.
The value on the division is: 72.3329083337503
The division from 9 to 10 has been calculated.
The value on the division is: 90.33785833375

-----
Integral value is: 333.3358333375
The program execution time is (seconds): 00:00:11.6283737
|
```

Hình 25: Kết quả chạy chương trình tính tích phân xác định



Hình 26: Các luồng chương trình đã chạy hiển thị trên Alchemi Console



Hình 27: CPU Power - Availability & Usage

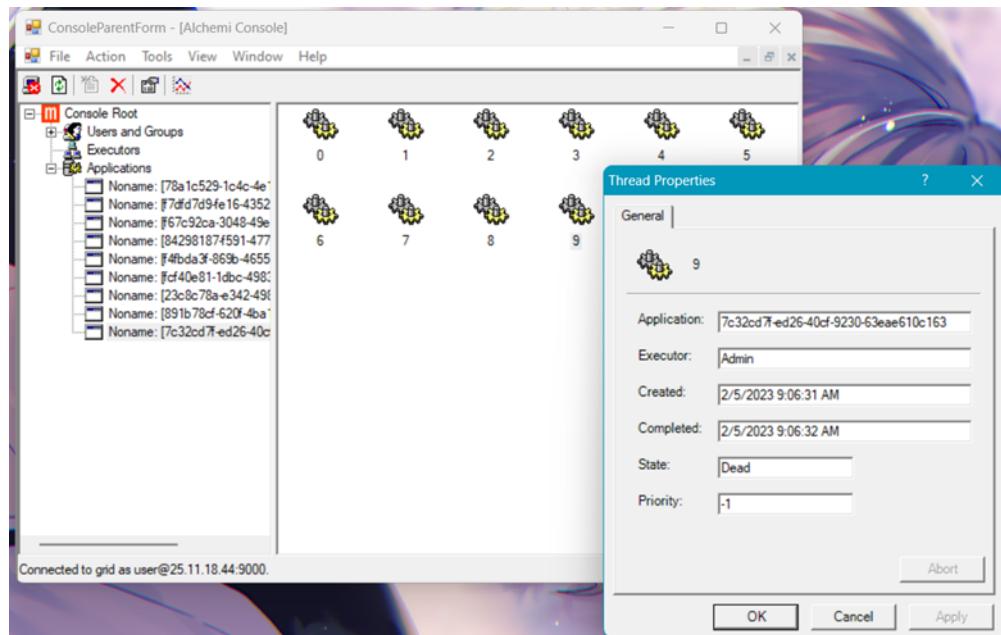
- 3 nút, $n = 300\ 000$, 10 luồng:

```

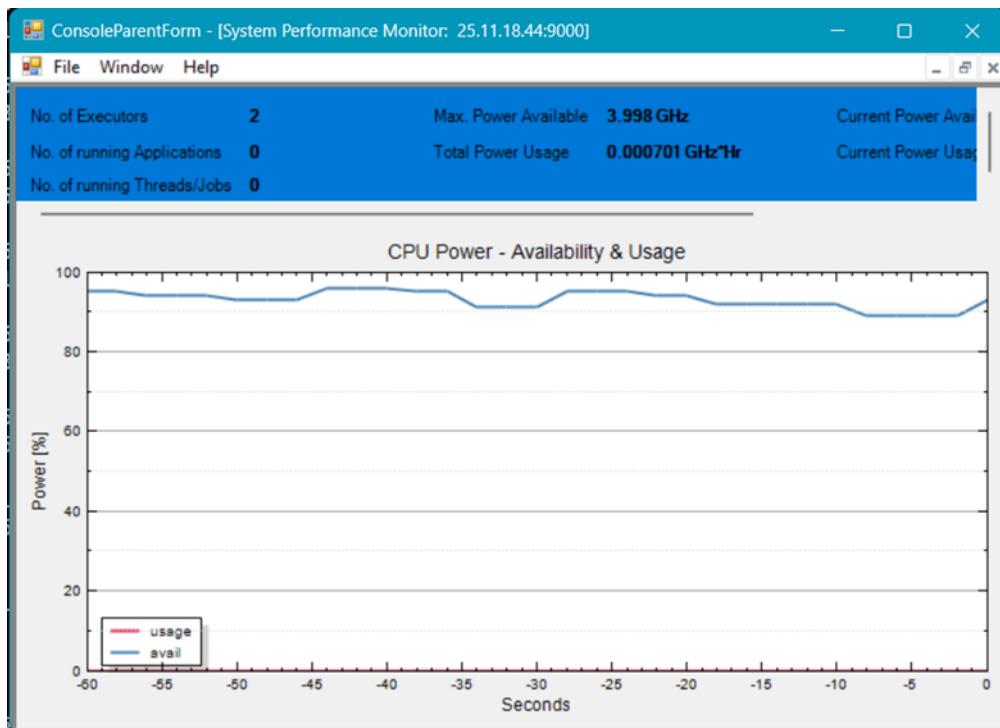
C:\Users\Admin\Desktop\Prin > 
The value on the division is: 2.33328333351852
The division from 3 to 3.99996666666667 has been calculated.
The value on the division is: 12.3332166668519
The division from 4 to 4.99996666666667 has been calculated.
The value on the division is: 20.3331833335186
The division from 5 to 5.99996666666667 has been calculated.
The value on the division is: 30.3331500001852
The division from 6 to 6.99996666666667 has been calculated.
The value on the division is: 42.3331166668519
The division from 7 to 7.99996666666667 has been calculated.
The value on the division is: 56.3330833335183
The division from 8 to 8.99996666666667 has been calculated.
The value on the division is: 72.3330500001847
The division from 9 to 10 has been calculated.
The value on the division is: 90.3363500001853

-----
Integral value is: 333.33500001851
The program execution time is (seconds): 00:00:10.8417074
|
```

Hình 28: Kết quả chạy chương trình tính tích phân xác định



Hình 29: Các luồng chương trình đã chạy hiển thị trên Alchemi Console



Hình 30: CPU Power - Availability & Usage

2 Bài tập nhóm riêng

Đề 5. Đọc vào 1 đoạn văn bản dưới dạng file input.txt, tìm kiếm cụm từ SAMI xuất hiện đầu tiên trong 1 câu (1 câu kết thúc bởi dấu chấm). Phân chia văn bản thành nhiều k văn bản nhỏ hơn (k nhập từ bàn phím), mỗi văn bản nhỏ một vài câu. Yêu cầu mỗi văn bản nhỏ thực hiện trên 1 nút tính toán. Tổng hợp vị trí các cụm từ SAMI trên từng dòng vào file output.txt, nếu không có cụm từ SAMI thì dòng đó để NULL.

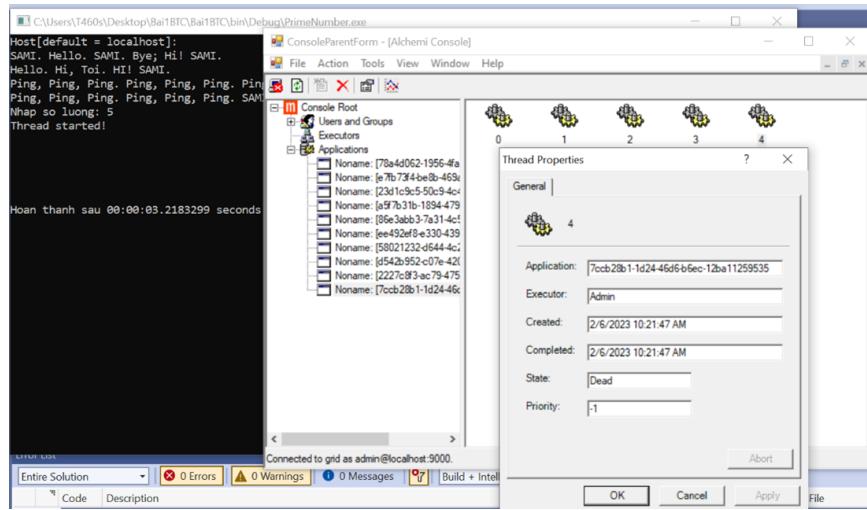
2.1 Kết quả chạy chương trình

Bảng 3: Kết quả chạy chương trình tổng hợp vị trí cụm từ SAMI với file input1.txt

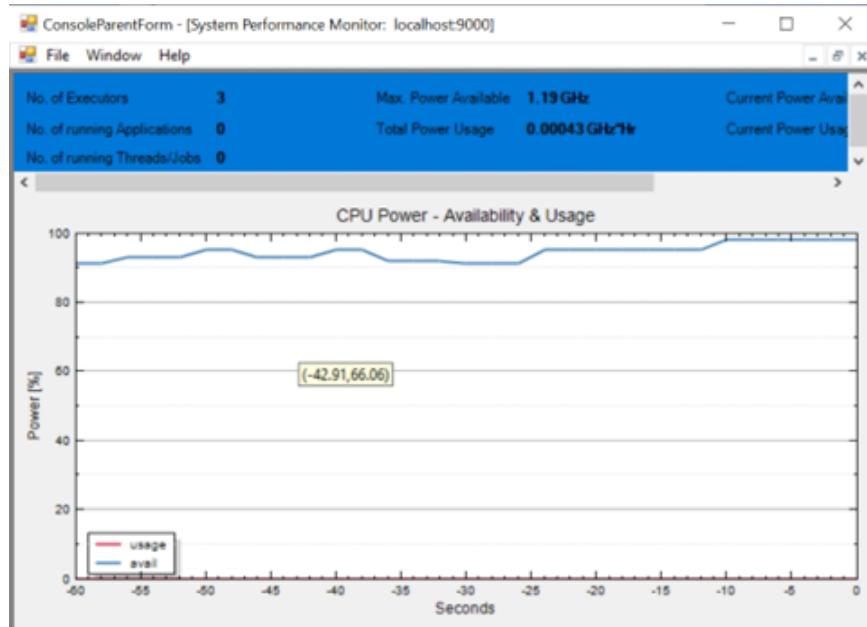
STT	Số nút	Số luồng - k	Thời gian (s)
1	2	5	3.2183299
2	3	5	3.3266272

Ảnh chụp màn hình khi chạy chương trình

- 2 nút, k = 5 luồng, input1.txt:



Hình 31: Kết quả chạy chương trình và số luồng



Hình 32: CPU Power - Availability & Usage

- 3 nút, k = 5 luồng, input1.txt:

The terminal window shows the following output:

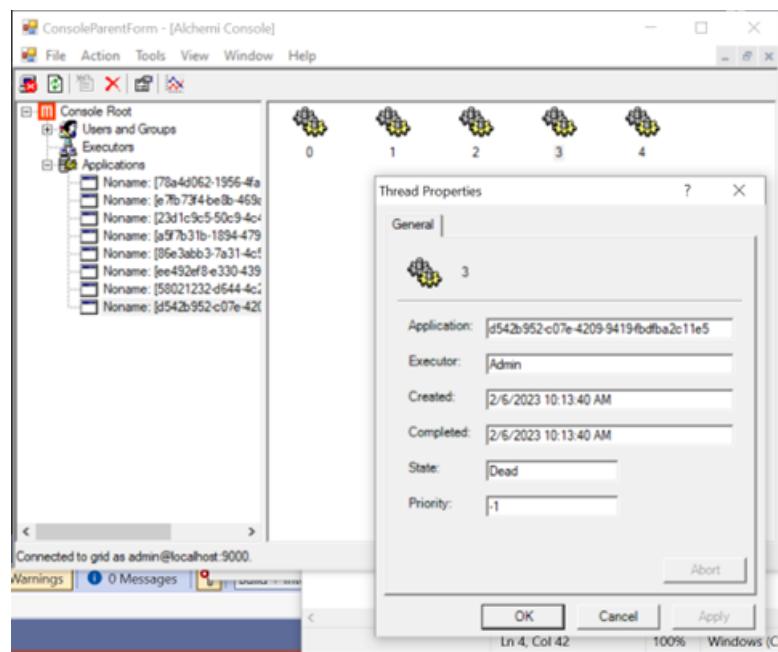
```
C:\Users\T440U\Desktop\Bu1IEC\Bu1IEC\Jan\Debug>PrimeNumber.exe
host[default * Localhost];
SAM! Hello SAM! Bye; H! SAM!
Hello. Hi, T! SAM!
Ping, Ping, Ping, Ping, Ping. Ping, Ping, Ping,
Ping, Ping, Ping, Ping, Ping. Ping, Ping, SAM!
Thap so luong: 5
Thread started!

van thanh sau 00:00:03.3266272 seconds.
```

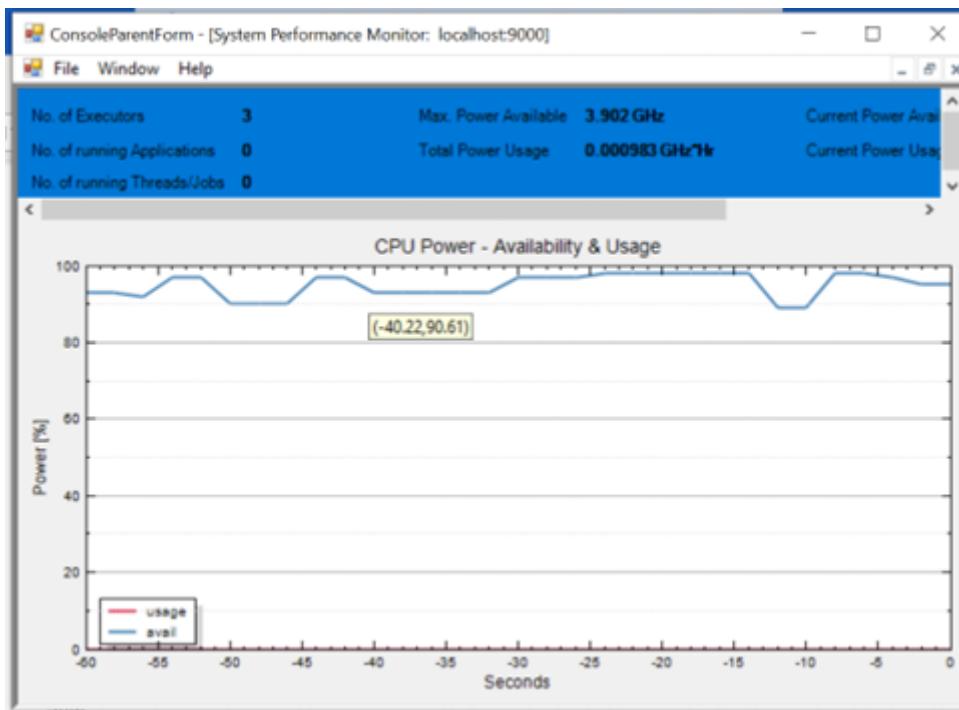
The Notepad window titled "input1.txt" contains the following text:

```
1
NULL
3
NULL
NULL
2
NULL
NULL
NULL
NULL
NULL
1
```

Hình 33: Kết quả chạy chương trình và file input, output



Hình 34: Các luồng chương trình đã chạy hiển thị trên Alchemi Console



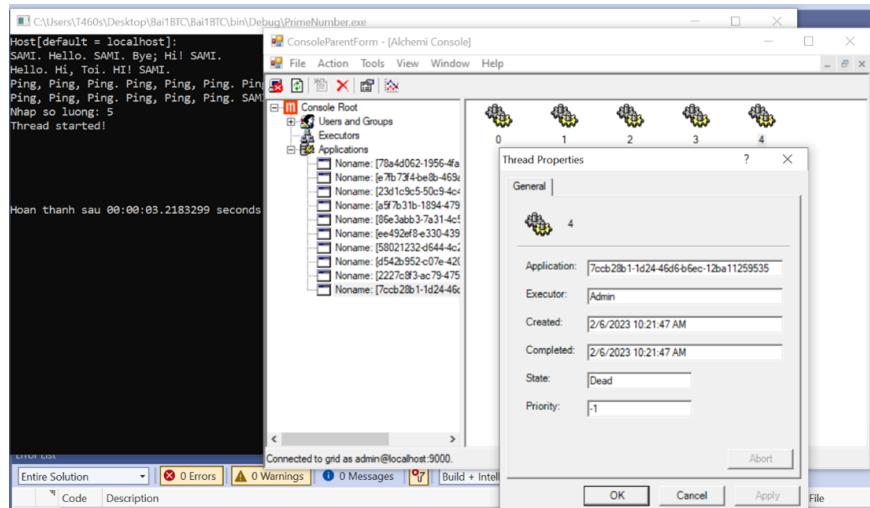
Hình 35: CPU Power - Availability & Usage

Bảng 4: Kết quả chạy chương trình tổng hợp vị trí cụm từ SAMI với file input2.txt

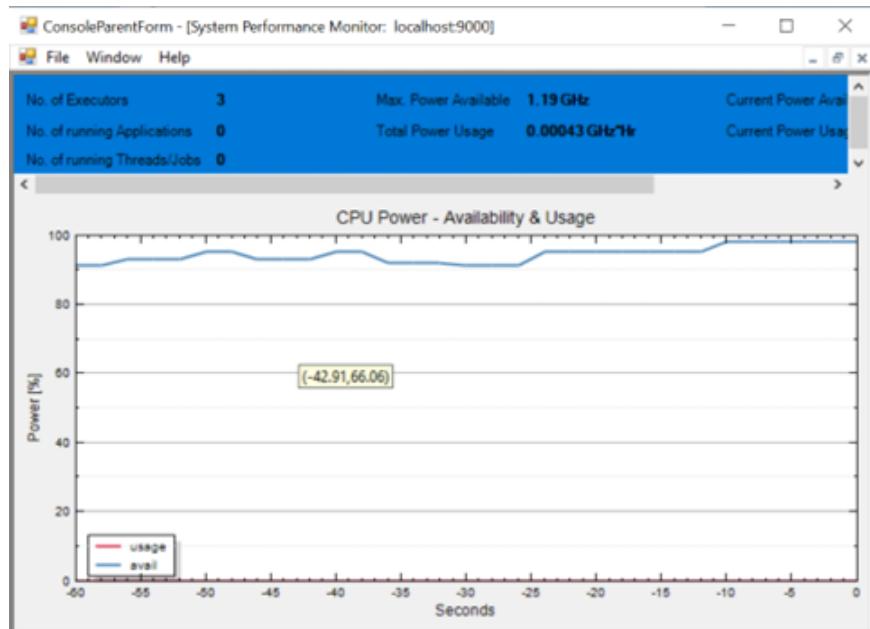
STT	Số nút	Số luồng - k	Thời gian (s)
1	2	4	3.2306804
2	3	5	3.2435525
3	3	10	3.5118910

Ảnh chụp màn hình khi chạy chương trình

- 2 nút, k = 5 luồng, input2.txt:



Hình 36: Kết quả chạy chương trình và số luồng



Hình 37: CPU Power - Availability & Usage

- 3 nút, k = 5 luồng, input2.txt:

```

C:\Users\TAIGC\Desktop\Ba1\Java\Java\src\Main.java
[default > localhost]
JMT Hello. SAMI. Bye; Hi! SAMI.
Hello. Hi. Tel. Hi! SAMI.
Ping. Ping. Ping. Ping. Ping. Ping. Ping. Ping. Ping.
Ping. Ping. Ping. Ping. Ping. Ping. Ping. Ping. Ping. SAMI.
Vép so luong: 5
Thread started!
1
NULL
1
3
NULL
SAMI. Hello. SAMI. Bye; Hi! SAMI.
NULL
Hello. Hi. Tel. Hi! SAMI.
2
NULL
NULL
NULL
NULL
NULL
1

```

run thanh sau 00:00:03.3266272 seconds.

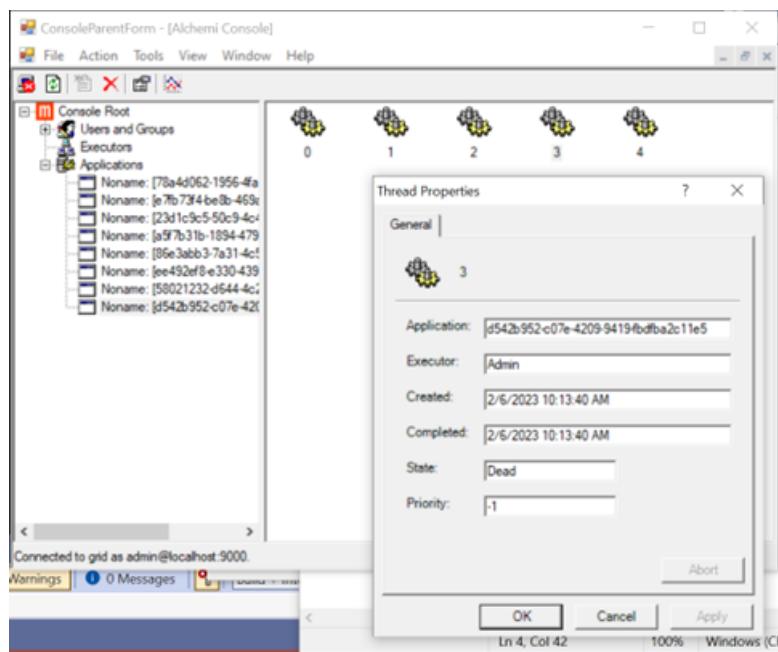
output - Notepad

File Edit Format View Help

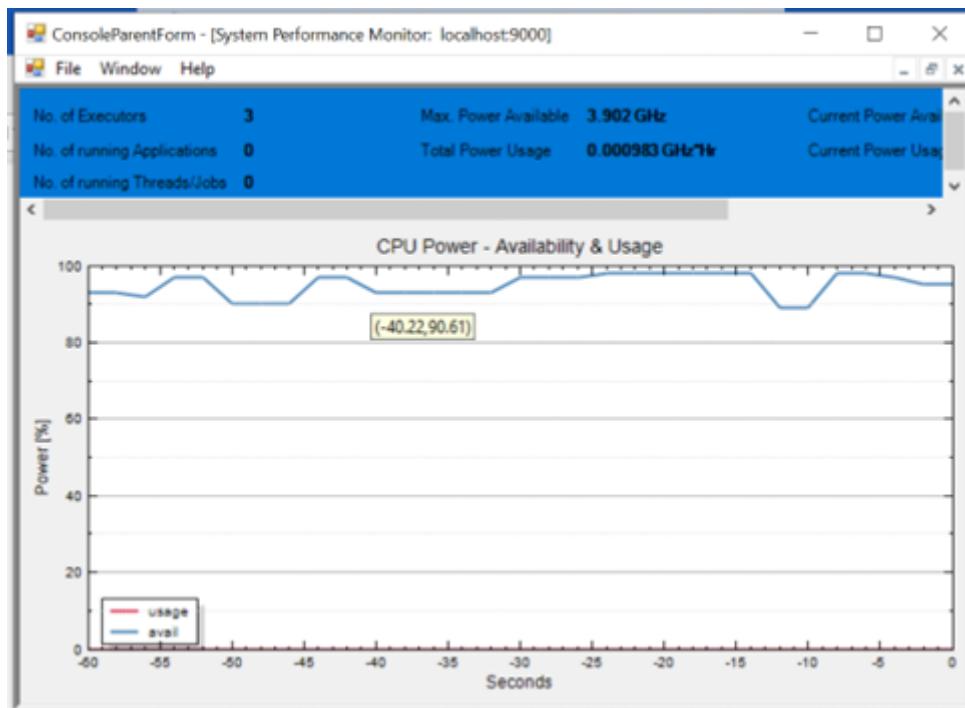
File Edit Format View Help

Locals Watch Tasks

Hình 38: Kết quả chạy chương trình và file input, output



Hình 39: Các luồng chương trình đã chạy hiển thị trên Alchemi Console



Hình 40: CPU Power - Availability & Usage

- 3 nút, k = 10 luồng, input2.txt:

```

C:\Users\T460\Desktop\Ba18TC\Ba18TC\src\Debug\Program1 - rifi - Notepad
host[default = localhost]:
say. Hello. SAMI. Bye; Hi! SAMI.
Hello. Hi. Toi. Hi! SAMI.
Ping. Ping. Ping. Ping. Ping. Ping. Ping. Ping.
Ping. Ping. Ping. Ping. Ping. Ping. Ping. Ping. Ping.
Ping. Ping. Ping. Ping. Ping. Ping. SAMI.Hi! Hello. SAMI.
SAM. Hello. SAMI. Bye; Hi! SAMI.
Hello. Hi. Toi. Hi! SAMI.

Thap so luong: 10
Thread started!

Hoan thanh sau 00:00:03.5118010 seconds.

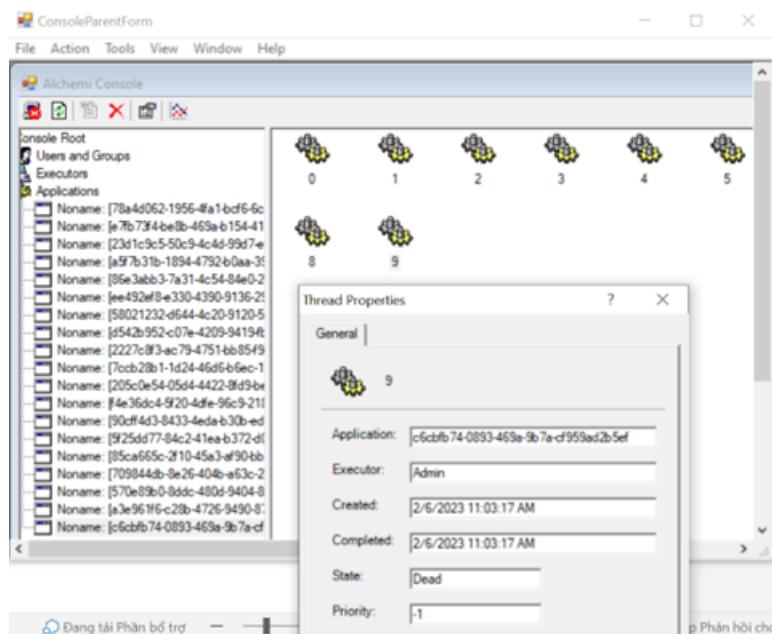
File Edit Format View Help
1
NULL
1
3
NULL
NULL
2
NULL
NULL
NULL
NULL
1
3
NULL
NULL
1
3
NULL
NULL
2
NULL

File Edit Format View Help
1
NULL
1
3
NULL
NULL
2
NULL
NULL
NULL
NULL
1
3
NULL
NULL
1
3
NULL
NULL
2
NULL

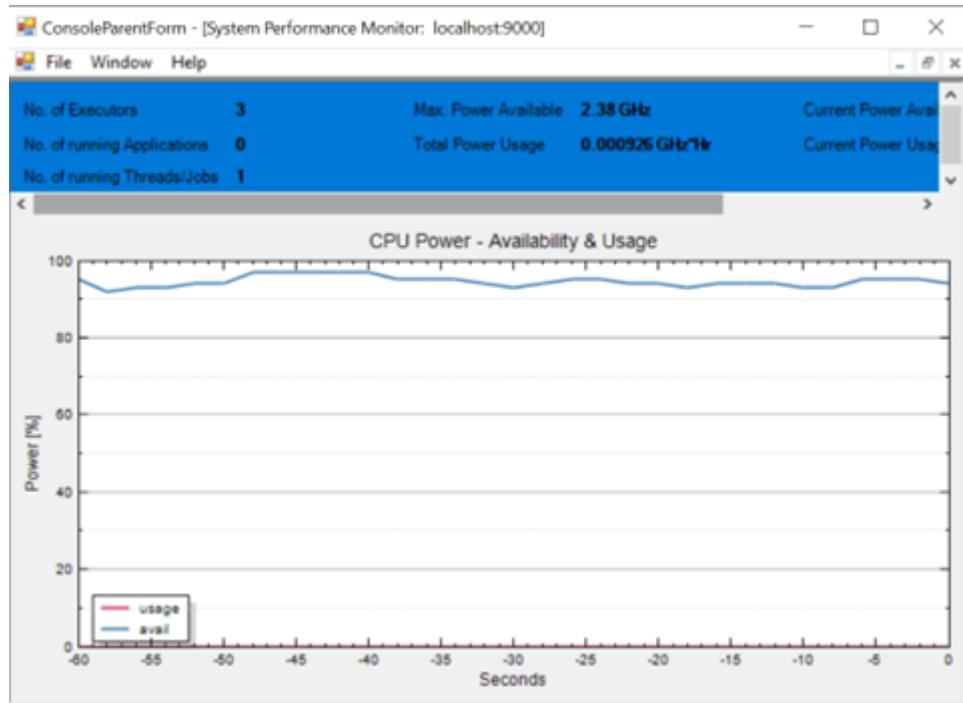
Solution - 0 Errors | 0 Warnings | 0 Warnings
Code Description
Locals Watch 1 Tasks

```

Hình 41: Kết quả chạy chương trình và file input, output



Hình 42: Các luồng chương trình đã chạy hiển thị trên Alchemi Console



Hình 43: CPU Power - Availability & Usage

3 Bài tập thêm

Đề 5. Liệt kê dãy Fibonacci đến n số cho trước (n - nhập từ bàn phím).

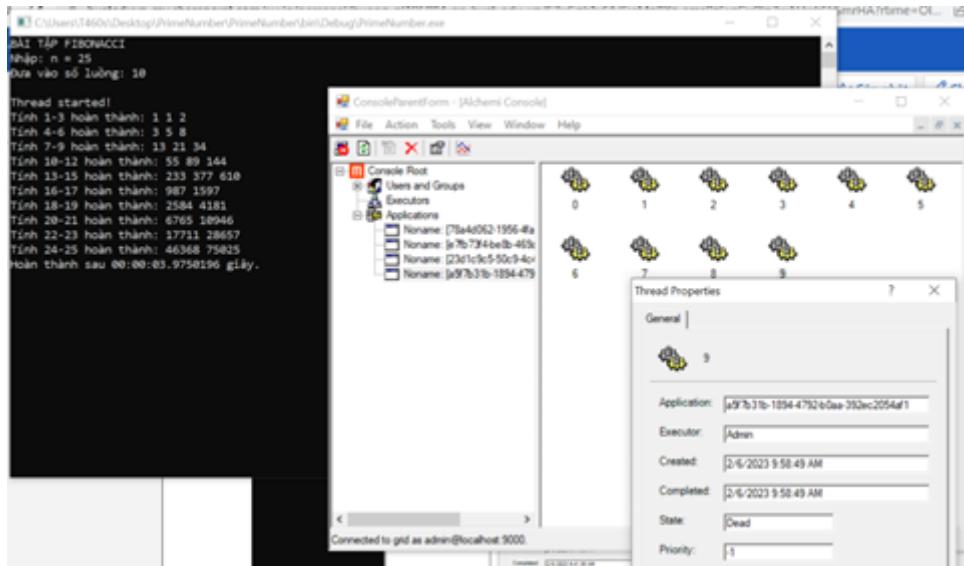
3.1 Kết quả chạy chương trình Liệt kê số Fibonacci

Bảng 5: Kết quả chạy chương trình liệt kê dãy Fibonacci đến số n cho trước

STT	Số nút	Số nguyên n	Số luồng	thời gian (s)
2	2	25	10	3.9750196
1	3	10	6	3.2014023
3	3	25	10	3.4733107

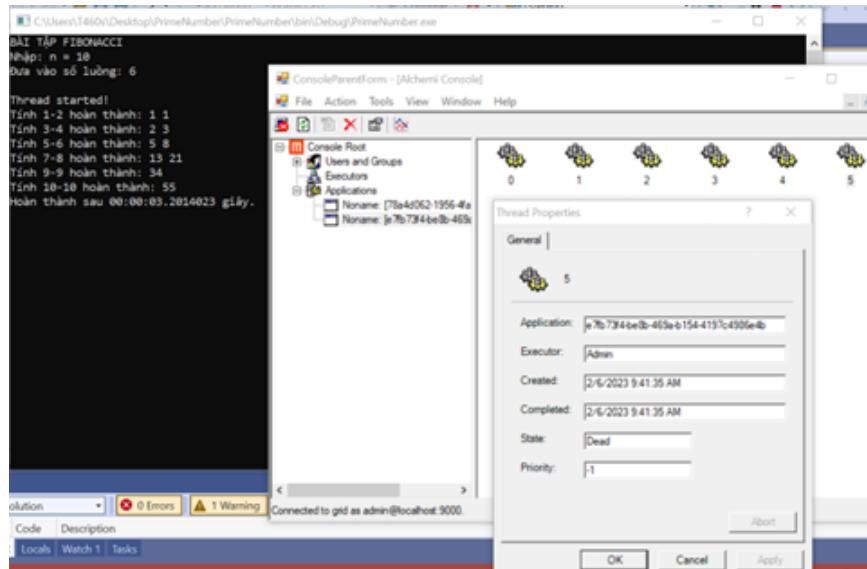
Ảnh chụp màn hình khi chạy chương trình

- 2 nút, $n = 10$, 6 luồng:

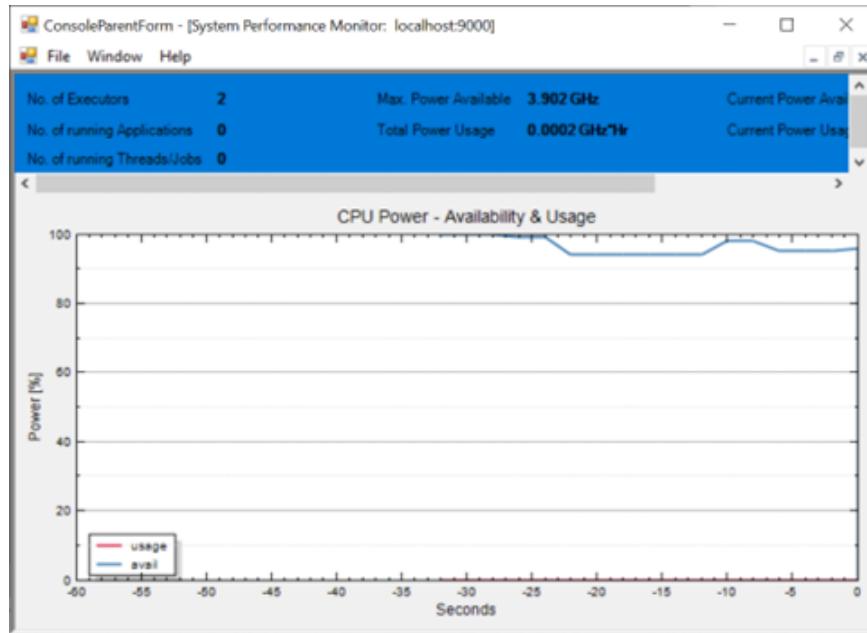


Hình 44: Kết quả chạy chương trình và số luồng liệt kê các số Fibonacci

- 3 nút, $n = 10$, 6 luồng:

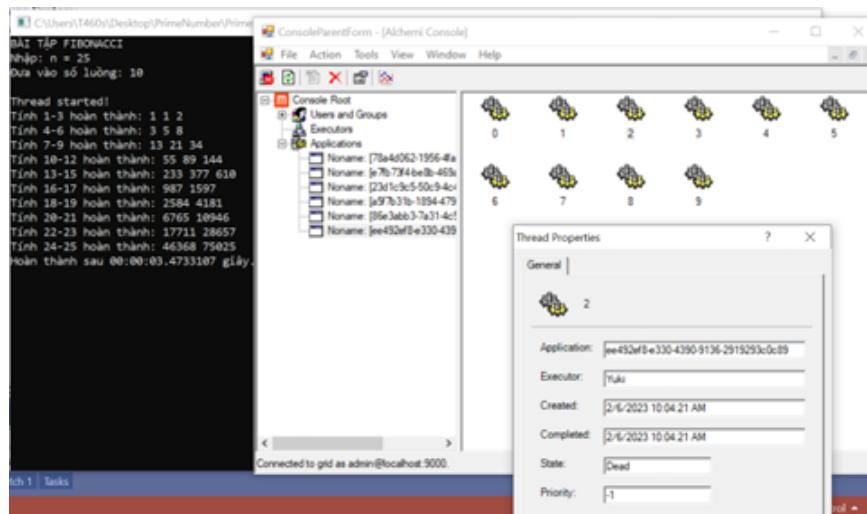


Hình 45: Kết quả chạy chương trình và số luồng liệt kê các số Fibonacci

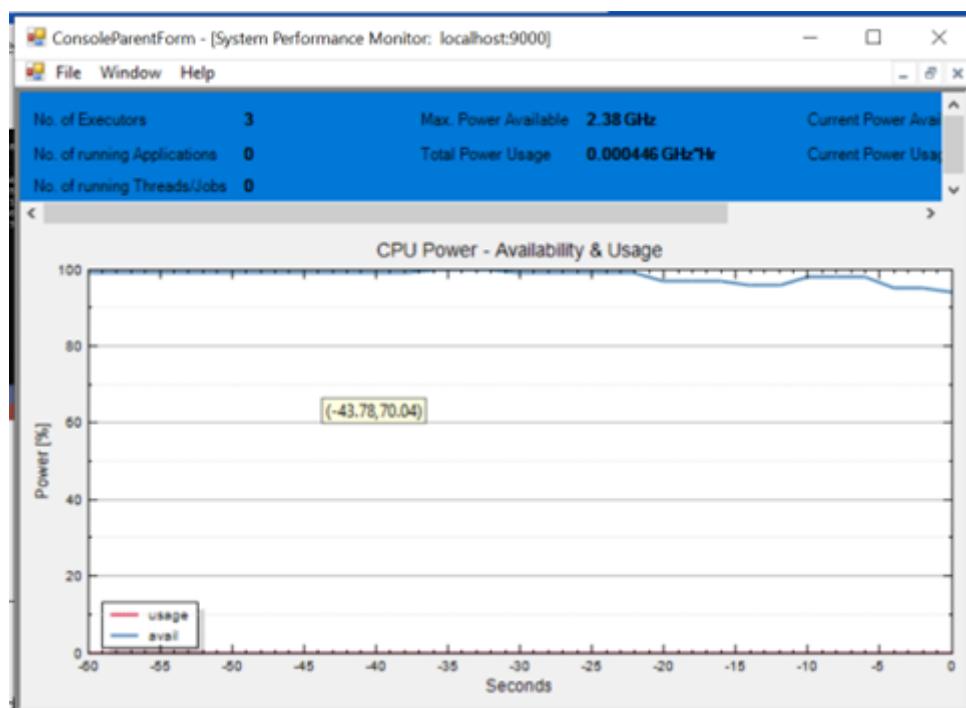


Hình 46: CPU Power - Availability & Usage

- 3 nút, $n = 25$, 10 luồng:



Hình 47: Kết quả chạy chương trình và số luồng liệt kê các số Fibonacci



Hinh 48: CPU Power - Availability & Usage

1 Bài thực hành chung

1.1 Liệt kê các số chính phương

```
1 using System;
2 using System.Collections.Generic;
3 using System.Text;
4 using Alchemi.Core;
5 using Alchemi.Core.Owner;
6
7 namespace ListPerfectSquare
8 {
9     class ListPerfectSquare : GApplication
10    {
11        public static GApplication App = new GApplication();
12        private static int NumberOfThread;
13        private static DateTime start;
14        private static List<int> ListPerfectSquareNumber = new List<int>();
15
16        [STAThread]
17        static void Main(string[] args)
18        {
19            int n;
20            int firstNumber = 1;
21            int lastNumber;
22            int numberofElement;
23            string host;
24            Console.Write("Host[localhost]: ");
25            host = Console.ReadLine();
26            if (host.Length < 1)
27                host = "localhost";
28            Console.Write("Enter the natural number n: ");
29            n = Convert.ToInt32(Console.ReadLine());
30            Console.Write("Enter the number of threads: ");
31            NumberOfThread = Convert.ToInt32(Console.ReadLine());
32
33            numberofElement = Convert.ToInt32(n / NumberOfThread);
34
35            for (int i = 0; i < NumberOfThread - 1; i++)
36            {
37                lastNumber = firstNumber + numberofElement;
38                App.Threads.Add(new PerfectSquareNumberCheck(firstNumber, lastNumber - 1));
39                firstNumber = lastNumber;
40            }
41            App.Threads.Add(new PerfectSquareNumberCheck(firstNumber, n));
42
43            //-----
44            App.Connection = new GConnection(host, 9000, "user", "user");
45            App.Manifest.Add(new ModuleDependency(typeof(PerfectSquareNumberCheck).Module));
46            App.ThreadFinish += new GThreadFinish(App_ThreadFinish);
47            App.ApplicationFinish += new GApplicationFinish(App_ApplicationFinish);
```

```

48         start = DateTime.Now;
49         Console.WriteLine("Thread started!");
50         Console.WriteLine("\n-----");
51         App.Start();
52         Console.ReadLine();
53     }
54     private static void App_ThreadFinish(GThread thread)
55     {
56         PerfectSquareNumberCheck PSNC = (PerfectSquareNumberCheck)thread;
57         Console.WriteLine("\nThe numbers from {0} to {1} have been checked.",
58             PSNC.firstNumber, PSNC.lastNumber);
59         Console.Write("The perfect squares are: ");
60         for (int i = 0; i < PSNC.ListPerfectSquare.Count; i++)
61         {
62             Console.Write(PSNC.ListPerfectSquare[i] + ", ");
63             ListPerfectSquareNumber.Add(PSNC.ListPerfectSquare[i]);
64         }
65         Console.WriteLine();
66     private static void App_ApplicationFinish()
67     {
68         Console.WriteLine("\n-----\n");
69         Console.Write("The list of perfect squares is: ");
70         for (int i = 0; i < ListPerfectSquareNumber.Count; i++)
71             Console.Write(ListPerfectSquareNumber[i] + ", ");
72
73         Console.WriteLine("\n\n-----");
74         Console.WriteLine("\nThe program execution time is (seconds): {0}", DateTime.Now
75 - start);
76     }
77 }
78 //Class that checks if a natural number is a perfect square
79 [Serializable]
80 class PerfectSquareNumberCheck : GThread
81 {
82     public int firstNumber;
83     public int lastNumber;
84     public List<int> ListPerfectSquare = new List<int>();
85     public PerfectSquareNumberCheck(int firstNumber, int lastNumber)
86     {
87         this.firstNumber = firstNumber;
88         this.lastNumber = lastNumber;
89     }
90     public override void Start()
91     {
92         int temp;
93         for (int i = firstNumber; i <= lastNumber; i++)
94         {
95             temp = Convert.ToInt32(Math.Sqrt(i));
96             if (temp * temp == i)
97                 ListPerfectSquare.Add(i);
98         }
99     }
100 }
101 }
```

1.2 Tính gần đúng tích phân xác định

```
1 using System;
2 using System.Collections.Generic;
3 using System.Text;
4 using Alchemi.Core;
5 using Alchemi.Core.Owner;
6
7 namespace IntegralValue
8 {
9     class IntegralValue : GApplication
10    {
11        public static GApplication App = new GApplication();
12        private static DateTime start;
13
14        private static long NumberOfThread;
15        private static long NumberOfElementThreads;
16        private static long NumberofLoop;
17        private static double integralResult = 0;
18
19        public static double Function(double x)
20            => Math.Pow(x, 2) + 1;
21
22        [STAThread]
23        static void Main(string[] args)
24        {
25            double n;
26            double lengthOfSubdivisons;
27            long firstNumber = 0;
28            long lastNumber;
29            string host;
30            Console.Write("Host[localhost]: ");
31            host = Console.ReadLine();
32            if (host.Length < 1)
33                host = "localhost";
34            Console.Write("Enter the upper bound n = ");
35            n = Convert.ToDouble(Console.ReadLine());
36            Console.Write("Enter the number of subdivisions: ");
37            NumberofLoop = Convert.ToInt64(Console.ReadLine());
38            Console.Write("Enter the number of threads: ");
39            NumberOfThread = Convert.ToInt64(Console.ReadLine());
40
41            lengthOfSubdivisons = n / NumberofLoop;
42            NumberOfElementThreads = Convert.ToInt64(NumberOfLoop / NumberOfThread);
43
44            for (long i = 0; i < NumberOfThread - 1; i++)
45            {
46                lastNumber = firstNumber + NumberOfElementThreads;
47                App.Threads.Add(new CalculateIntegral(firstNumber, lastNumber - 1,
lengthOfSubdivisons));
48                firstNumber = lastNumber;
49            }
50            App.Threads.Add(new CalculateIntegral(firstNumber, NumberofLoop,
lengthOfSubdivisons));
51
52            //-----
53            App.Connection = new GConnection(host, 9000, "user", "user");
54            App.Manifest.Add(new ModuleDependency(typeof(CalculateIntegral).Module));
```

```

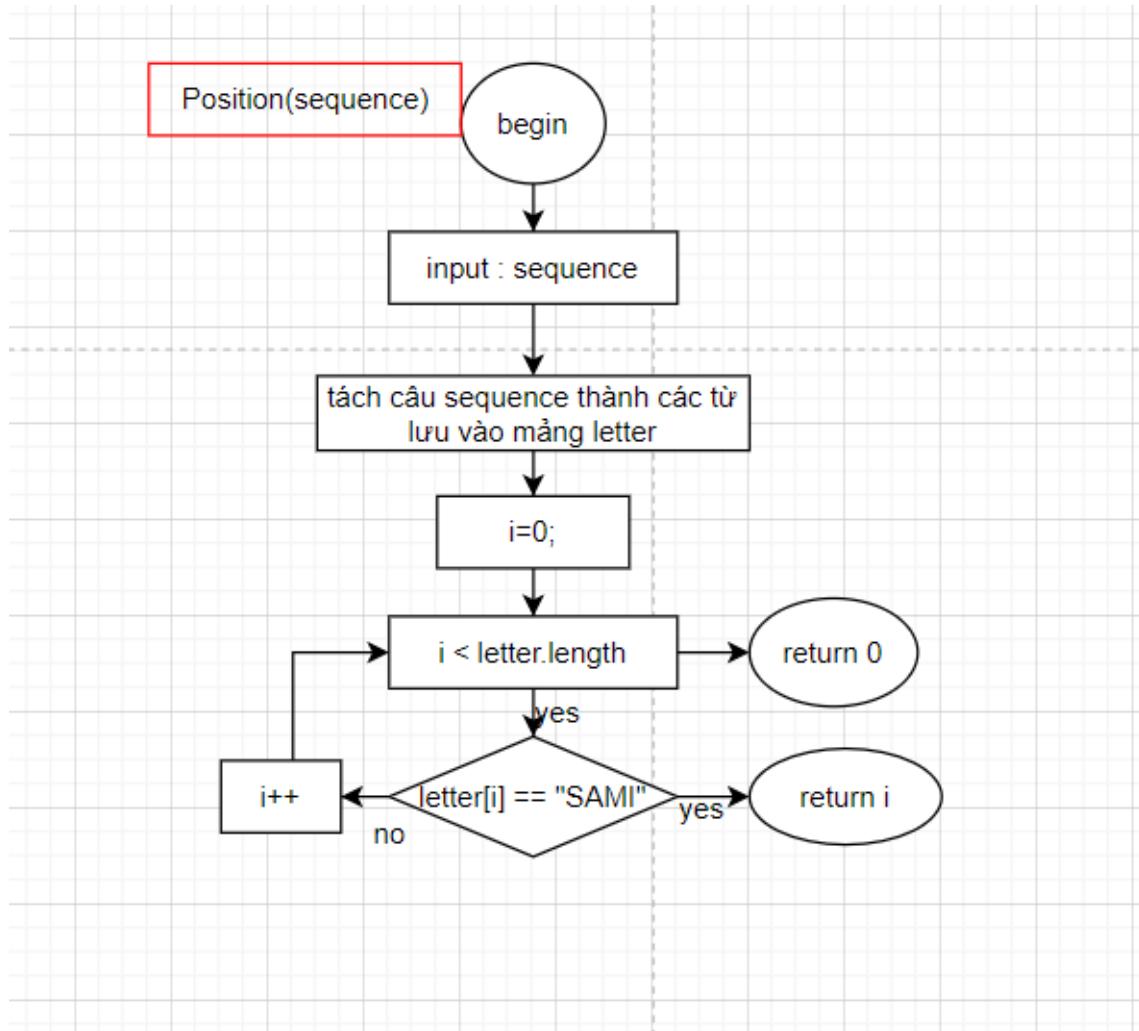
55     App.ThreadFinish += new GThreadFinish(App_ThreadFinish);
56     App.ApplicationFinish += new GApplicationFinish(App_ApplicationFinish);
57     start = DateTime.Now;
58     Console.WriteLine("Thread started!");
59
60     Console.WriteLine("\n-----\n");
61     App.Start();
62     Console.ReadLine();
63 }
64
65     private static void App_ThreadFinish(GThread thread)
66     {
67         CalculateIntegral CI = (CalculateIntegral)thread;
68         Console.WriteLine("The division from {0} to {1} has been calculated.",
69             CI.firstNumber * CI.lengthOfSubdivisons, CI.lastNumber * CI.lengthOfSubdivisons);
70         Console.WriteLine("The value on the division is: {0}", CI.result);
71         integralResult += CI.result;
72         Console.WriteLine();
73     }
74
75     private static void App_ApplicationFinish()
76     {
77
78         Console.WriteLine("\n-----");
79         Console.WriteLine("\nIntegral value is: {0}", integralResult);
80         Console.WriteLine("\nThe program execution time is (seconds): {0}", DateTime.Now
81             - start);
82     }
83
84     [Serializable]
85     class CalculateIntegral : GThread
86     {
87
88         public long firstNumber;
89         public long lastNumber;
90         public double lengthOfSubdivisons;
91         public double result;
92
93         public CalculateIntegral(long FirstNumber, long LastNumber, double
94             lengthOfSubdivisons)
95         {
96             this.firstNumber = FirstNumber;
97             this.lastNumber = LastNumber;
98             this.lengthOfSubdivisons = lengthOfSubdivisons;
99         }
100        public override void Start()
101        {
102            result = 0;
103            for (long i = firstNumber; i <= lastNumber; i++)
104                result += IntegralValue.Function(i * lengthOfSubdivisons);
105            result = result * lengthOfSubdivisons;
106        }
107    }
108 }

```

2 Bài tập nhóm riêng

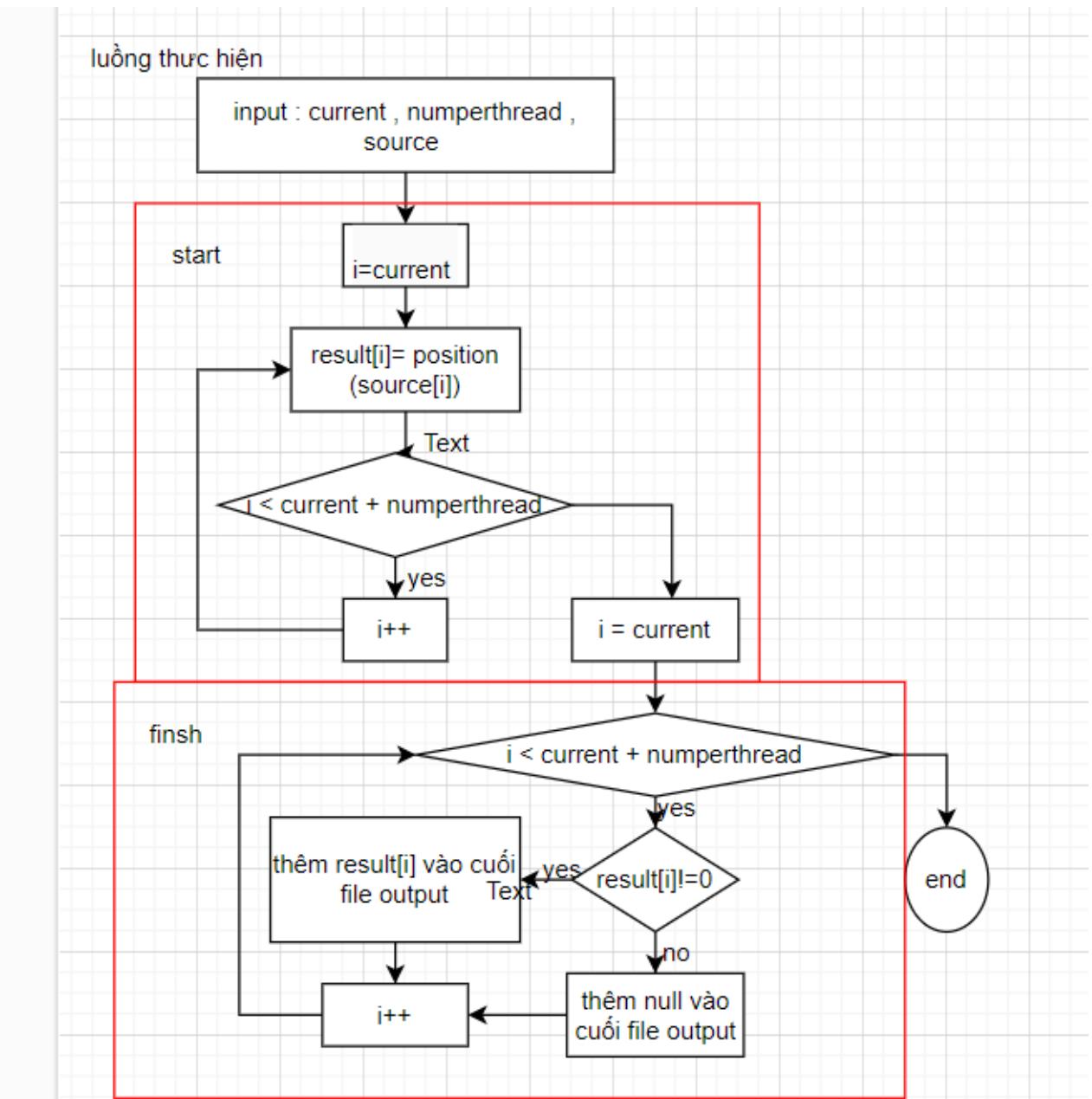
2.1 Sơ đồ thuật toán

Đầu tiên ta xây dựng hàm tìm vị trí từ SAMI trong 1 câu . Hàm Position có input đầu vào là 1 câu . Output đầu ra là vị trí từ SAMI trong câu nếu không có từ SAMI thì trả về 0 :



Hình 49: Tìm vị trí từ SAMI trong câu

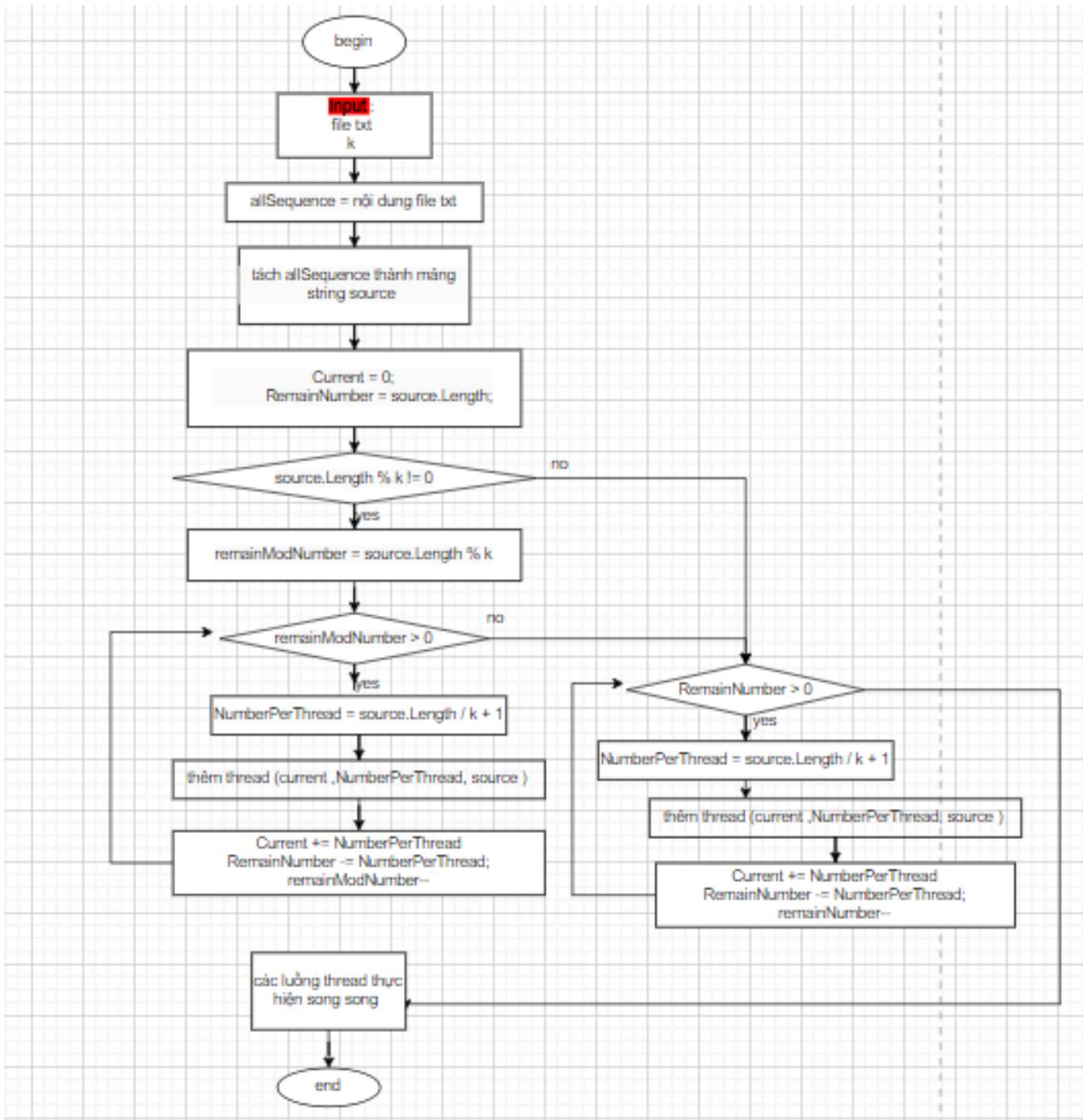
Tiếp theo ta thực hiện xây dựng hoạt động thực hiện trên mỗi luồng :



Hình 50: Các luồng hoạt động

Đầu vào mỗi luồng là current , numperthread , source . Mỗi luồng sẽ thực hiện tìm kiếm từ trong các câu từ câu thứ current đến câu thứ numperthread + current -1 . Sau đó thực hiện ghi kết quả vào file nếu câu không có từ SAMI thì thực hiện viết NULL vào file .

Cuối cùng là hàm chính :



Hình 51: Hàm chính

2.2 Mā nguồn

```
1 using System;
2 using System.Collections.Generic;
3 using System.IO;
4 using System.Text;
5 using Alchemi.Core;
6 using Alchemi.Core.Owner;
7
8
9 namespace Bai1BTC
10 {
11     class SoChinhPhuong : GApplication
12     {
13         public static GApplication App = new GApplication();
14         private static int[] matrix;
15         private static DateTime start;
16
17         [STAThread]
18         static void Main(string[] args)
19         {
20             string host;
21             Console.Write("Host[default = localhost]:");
22             host = Console.ReadLine();
23             if (host.Length < 1)
24             {
25                 host = "localhost";
26             }
27
28             //doc van ban
29             FileStream fs = new FileStream("D:\\PrimeNumber\\PrimeNumber\\nifi.txt",
30             FileMode.Open);
31             StreamReader rd = new StreamReader(fs, Encoding.UTF8);
32             String allSequence = rd.ReadToEnd(); // ReadLine() ch c 1 dng u thoy,
ReadToEnd l c ht
33                                         // Console.WriteLine(giatri);
34
35             Console.WriteLine(allSequence);
36             rd.Close();
37
38             // duoc coi la so luong
39             Console.Write("Nhap so luong: ");
40             int k = Int32.Parse(Console.ReadLine()); // 2
41
42             // allSequence = "Hello! SAMI. Toi. SAMI. Bye, SAMI. School
43             string[] source = allSequence.Split(new char[] { '.' },
StringSplitOptions.RemoveEmptyEntries);
44             // { 'Hello!', 'Toi', 'SAMI', 'Bye, SAMI', 'School' }
45             int Current = 0;
46             int RemainNumber = source.Length;
47
48             if (source.Length % k != 0)
49             {
50                 int remainModNumber = source.Length % k;
51                 while ((remainModNumber--) > 0)
52                 {
53                     int NumberPerThread = source.Length / k + 1;
App.Threads.Add(new PositionLetter(Current, NumberPerThread, source));
```

```

54             Current += NumberPerThread;
55             RemainNumber -= NumberPerThread;
56         }
57     }
58
59     while (RemainNumber > 0)
60     {
61         int NumberPerThread = source.Length / k;
62         App.Threads.Add(new PositionLetter(Current, NumberPerThread, source));
63         Current += NumberPerThread;
64         RemainNumber -= NumberPerThread;
65     }
66
67     App.Connection = new GConnection(host, 9000, "user", "user");
68     App.Manifest.Add(new ModuleDependency(typeof(PositionLetter).Module));
69     App.ThreadFinish += new GThreadFinish(App_ThreadFinish);
70     App.ApplicationFinish += new GApplicationFinish(App_ApplicationFinish);
71     start = DateTime.Now;
72     Console.WriteLine("Thread started!");
73     App.Start();
74     Console.ReadLine();
75 }
76 private static void App_ThreadFinish(GThread thread)
77 {
78     PositionLetter pnc = (PositionLetter)thread;
79
80     int count = pnc.NumPerThread;
81     for (int i = pnc.StartNums; i < pnc.StartNums + pnc.NumPerThread; i++)
82     {
83         if (pnc.Result[i] != 0)
84         {
85             using (StreamWriter w = File.AppendText("D:\\output.txt"))
86             {
87                 w.WriteLine(pnc.Result[i]);
88             }
89         }
90         else
91         {
92             using (StreamWriter w = File.AppendText("D:\\output.txt"))
93             {
94                 w.WriteLine("NULL");
95             }
96         }
97     }
98 }
99 Console.WriteLine();
100
101
102 /*
103 for (int i = 0; i < pnc.NumPerThread; i++)
104 {
105     string path = "D:\\output.txt";
106     string data = File.ReadAllText(path);
107     string withHeader;
108     if (pnc.result[i] ==0)
109         withHeader = "NULL"+"\n" + data;
110     else
111

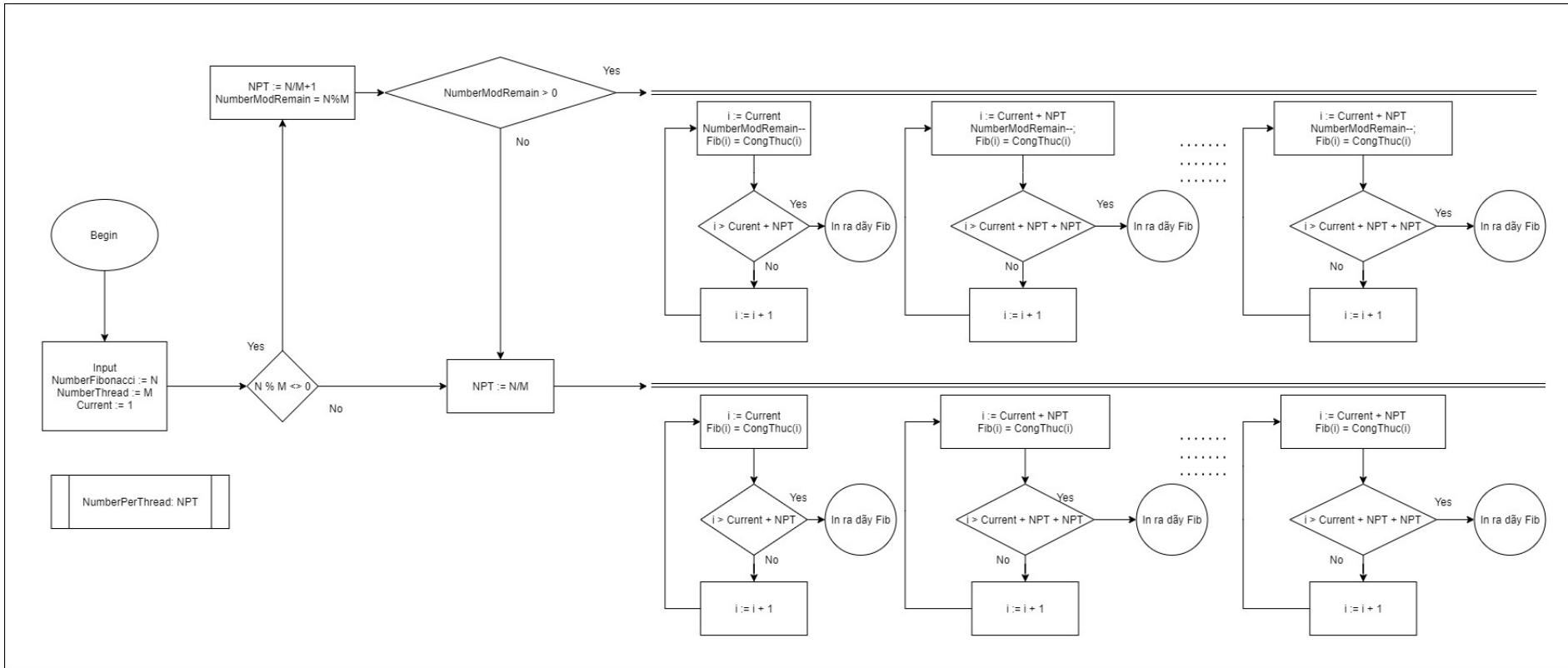
```

```

112             withHeader = pnc.result[i]+"\n" + data;
113             File.WriteAllText(path,withHeader);
114         }
115     */
116     }
117 
118     private static void App_ApplicationFinish()
119     {
120         Console.WriteLine("Hoan thanh sau {0} seconds.", DateTime.Now - start);
121     }
122 }
123 
124 [Serializable]
125 class PositionLetter : GThread
126 {
127     public int StartNums { get; set; }
128     public int NumPerThread { get; set; }
129     public string[] Sequence { get; set; }
130     public int[] Result { get; set; }
131 
132 
133     public PositionLetter(int startNums, int numPerThread, string[] sequence)
134     {
135         Sequence = sequence;
136         StartNums = startNums;
137         NumPerThread = numPerThread;
138     }
139 
140     static int Position(string sequence)
141     {
142         string[] letter = sequence.Split(new char[] { '?', '!', ' ', ';' , ':' , ',' , ' ' }, StringSplitOptions.RemoveEmptyEntries);
143         for (int i = 0; i < letter.Length; i++)
144         {
145             if (letter[i] == "SAMI")
146                 return i + 1;
147             }
148             return 0;
149         }
150 
151     public override void Start()
152     {
153         Result = new int[1000];
154         for (int i = StartNums; i < StartNums + NumPerThread; i++)
155         {
156             Result[i] = Position(Sequence[i]);
157         }
158 
159     }
160 
161 }
162 }
```

3 Bài tập thêm

3.1 Thuật toán



Hình 52: Sơ đồ khái chương trình liệt kê các số Fibonacci

3.2 Mã nguồn

```
1 using System;
2 using Alchemi.Core.Owner;
3
4 namespace FiboNumber
5 {
6     class FiboNumber : GApplication
7     {
8         public static GApplication App = new GApplication();
9         private static int NumThread;
10        private static DateTime start;
11
12        [STAThread]
13        static void Main(string[] args)
14        {
15            Console.OutputEncoding = System.Text.Encoding.UTF8;
16            Console.WriteLine("BI TP FIBONACCI");
17            Console.Write("Nhập: n = ");
18            int n = Int32.Parse(Console.ReadLine());
19            Console.Write("a    vo    s    lung : ");
20            NumThread = Int32.Parse(Console.ReadLine());
21
22            Console.WriteLine();
23            int NumRemain = n; // 8 5 => 2, 2, 2, 1, 1
24            int NumCur = 1;
25            int ThreadCur = 1;
26
27            if (n % NumThread != 0) {
28                int NumPerThread = n / NumThread + 1;
29                while (ThreadCur * NumPerThread < n)
30                {
31                    App.Threads.Add(new FiboNumberCheck(NumCur, NumCur + NumPerThread - 1));
32                    NumCur += NumPerThread;
33                    NumRemain -= NumPerThread;
34                    ThreadCur += 1;
35                }
36            }
37
38            while (NumRemain > 0)
39            {
40                int NumPerThread = n / NumThread;
41                App.Threads.Add(new FiboNumberCheck(NumCur, NumCur + NumPerThread - 1));
42                NumCur += NumPerThread;
43                NumRemain -= NumPerThread;
44            }
45
46
47
48            App.Connection = new GConnection("localhost", 9000, "user", "user");
49            App.Manifest.Add(new ModuleDependency(typeof(FiboNumberCheck).Module));
50            App.ThreadFinish += new GThreadFinish(App_ThreadFinish);
51            App.ApplicationFinish += new GApplicationFinish(App_ApplicationFinish);
52            start = DateTime.Now;
53            Console.WriteLine("Thread started!");
54            App.Start();
55            Console.ReadLine();
56        }
```

```

57 }
58
59     private static void App_ThreadFinish(GThread thread)
60 {
61         FiboNumberCheck fnc = (FiboNumberCheck)thread;
62         Console.WriteLine("Tnh {0}-{1} hon thnh: ", fnc.Begin, fnc.End);
63         for (int i = fnc.Begin; i <= fnc.End; i++)
64         {
65             Console.WriteLine( fnc.Fibos[i] + " ");
66         }
67         Console.WriteLine();
68     }
69     private static void App_ApplicationFinish()
70 {
71     Console.WriteLine("Hon thnh sau {0} giy.", DateTime.Now - start);
72 }
73 }
74 [Serializable]
75 class FiboNumberCheck : GThread
76 {
77     public int Begin { get; set; }
78     public int End { get; set; }
79     public long[] Fibos = new long[100];
80     public FiboNumberCheck(int begin, int end)
81     {
82         Begin = begin;
83         End = end;
84     }
85     private long Fibo(int m)
86     {
87         double s = 1.0 / Math.Sqrt(5.0);
88         double g1 = (1.0 + Math.Sqrt(5.0)) / 2.0;
89         double g2 = (1.0 - Math.Sqrt(5.0) / 2.0);
90         return (long)Math.Round(s * (Math.Pow(g1, m) - Math.Pow(g2, m)));
91     }
92     public override void Start()
93     {
94         for (int i = Begin; i <= End; i++)
95         {
96             Fibos[i] = Fibo(i);
97         }
98     }
99 }
100 }

```

III

Nhận xét

Qua quá trình tìm hiểu và thực hành lập trình tính toán lưới các bài toán trên nền tảng Alchemi sử dụng ngôn ngữ lập trình C Sharp (C#) chúng em đã rút ra được một số nhận xét:

1. Lập trình tính toán lưới trên Alchemi là khá dễ dàng, mô hình lưới dễ thiết lập, việc quản lý quá trình thực thi của các luồng là rõ ràng giúp cho việc nâng cấp cũng như chỉnh sửa mã nguồn một cách đơn giản nhanh chóng.
2. Đối với những bài tập được giao, nhóm đã cùng nhau trao đổi, phân chia công việc tìm hiểu thuật toán, xây dựng chương trình chạy trên nền tảng Alchemi.
3. Rút ra được cách làm việc cùng nhau tại các nhiều địa điểm bằng cách sử dụng Hamachi để tạo mạng ảo.

Tài liệu tham khảo

- [1] The University of Melbourne Australia, *Alchemi .NET-based Enterprise Grid System and Framework*, Krishna Nadiminti, Akshay Luther, Rakumar Buyya, (2005).
- [2] <https://www.geeksforgeeks.org/>
- [3] <https://stackoverflow.com/>
- [4] Đoàn Duy Trung, *Slide bài giảng Tính toán song song*