ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

Phân tích và thiết kế thuật toán

Bài tập nhóm 10

Nhóm 13: Đỗ Quang Lực - 23520902
Nguyễn Minh Huy - 23520634

Ngày 2 tháng 12 năm 2024

Mục lục

1 Bài 1 **3**
1.1 Cài đặt 3
1.2 Giải thích 3

2 Bài 2 **4**
2.1 Cài đặt 4
2.2 Giải thích 4

1 Bài 1

Đề bài: xây dựng thuật toán kiểm tra số nguyên tố song song,

1.1 Cài đặt

Link cài đặt: https://github.com/doquanglucuit/CS112/blob/main/Nhom10-B1.py

1.2 Giải thích

Cách đơn giản nhất để kiểm tra một số n có phải là số nguyên tố hay không đó là kiểm tra phép chia lấy dư của n cho tất cả các số từ 2 đến $\sqrt{n}$.
Vì số lượng chia hết cho 2 và 3 tương đối nhiều nên có thể kiểm tra trước, và chỉ cần kiểm tra các số từ 5 đến $\sqrt{n}$

Giải pháp song song: Vì mỗi phép chia là độc lập, ta có thể chia các số từ 5 đến $\sqrt{n}$ thành các khoảng khác nhau, rồi đưa cho mỗi tiến trình xử lí một khoảng, rồi tổng hợp kết quả từ các tiến trình để xác định tính nguyên tố của n.

Thử với testcase:

```
X = 1000000000000091
Tuan tu: True, Time: 2.093888s
Song song: True, Time: 1.025508s
X = 10000000000000099
Tuan tu: True, Time: 6.138579s
Song song: True, Time: 1.525487s
X = 100000000000000049
Tuan tu: True, Time: 19.750164s
Song song: True, Time: 3.221590s
```

Thử với nhiều testcase random:

```
Số lượng testcase = 20
Tuần tự: 0.002550s
Song song: 2.437948s
Số lượng testcase = 50
Tuần tự: 0.005856s
Song song: 3.772009s
Số lượng testcase = 100
Tuần tự: 0.035315s
Song song: 7.103456s
Số lượng testcase = 200
Tuần tự: 0.040066s
Song song: 17.150833s
```

Nhận xét: với các testcase lớn, tập số cần kiểm tra lớn, khi sử dụng thuật toán song song sẽ giảm được rất nhiều lượng tính toán. Thử với nhiều số random, sẽ có những trường hợp rơi vào số không phải là số nguyên tố nên thuật toán tuần tự sẽ phản hồi rất nhanh, còn thuật toán song song sẽ tốn chi phí khởi tạo và quản lí luồng.

2 Bài 2

Đề bài: Xây dựng thuật toán nhân hai ma trận song song

2.1 Cài đặt

Link cài đặt: https://github.com/doquanglucuit/CS112/blob/main/Nhom10-B2.py

2.2 Giải thích

Với $A(m,n),\ B(n,p):\ C = A \times B$ là ma trận có kích thước $n \times p$ được tính như sau:

$$C_{ij} = \sum_{k=1}^{n} A_{ik} \cdot B_{kj}$$

Giải pháp song song: Nhận thấy rằng các phần tử trên mỗi hàng của ma trận C được tính độc lập, nên có thể chia thành nhiều phần cho mỗi tiến trình tính toán, rồi tổng hợp kết quả lại.

Thử với A, B kích thước 400×400**:**

```
Với ma trận kích thước 400 x 400:
Tuần tự: 16.366248s
Song song: 2.568292s
```

Thử với nhiều testcase random:

```
Với ma trận kích thước50 × 50:
Tuần tự: 0.017373s
Song song: 0.317572s
Với ma trận kích thước100 × 100:
Tuần tự: 0.233137s
Song song: 0.484688s
Với ma trận kích thước300 × 300:
Tuần tự: 6.572353s
Song song: 1.944793s
Với ma trận kích thước500 × 500:
Tuần tự: 33.358268s
Song song: 3.833209s
```

Nhận xét: có thể thấy rằng với ma trận có kích thước nhỏ do chi phí khởi tạo và quản lí luồng lớn nên mặc dù tính toán nhanh hơn, nhưng lại có thời gian thực thi lớn hơn. Đối với ma trận có kích thước lớn, việc thực thi song song nhanh hơn rất nhiều lần so với thực hiện tuần tự.