Trường Đại học Khoa học Tự Nhiên
Khoa Công nghệ Thông tin

Nhập môn Phân tích Độ phức tạp Thuật toán

Lớp: 18CNTN

BÁO CÁO BT01
PHÉP GÁN VÀ PHÉP SO SÁNH

Giáo viên hướng dẫn

Nguyễn Thành An
Nguyễn Hải Đăng
Đỗ Trọng Lễ
Nguyễn Vinh Tiệp

Sinh viên thực hiện

18120014 - Đào Thành Đạt

Mục lục

1 Thuật toán đệ quy

1.1 Tính toán lý thuyết

```
int fibonacci1(int n)
{
  if (n <= 1)
    return 1;
  else
    return fibonacci1(n - 1) + fibonacci1(n - 2);
}
```

- Phép gán:

  Với thuật toán Fibonacci sử dụng đệ quy trên, ta dễ thấy thuật toán không có phép gán.

- Phép so sánh:

  Đặt số phép so sánh của fibonacci(n) là S_n ta có: $S_n = S_{n-1} + S_{n-2} + 1$, với $S_0 = 1$, $S_1 = 1$. Ta dùng hàm sinh để giải công thức này:

  Đặt G(x) là hàm sinh cho dãy S_n. Nhân 2 vế của công thức truy hồi cho x^n ta có:

$$\sum_{n=2}^{\infty} S_n x^n = \sum_{n=2}^{\infty} S_{n-1} x^n + \sum_{n=2}^{\infty} S_{n-2} x^n + \sum_{n=2}^{\infty} x^n$$

$$\Leftrightarrow \sum_{n=0}^{\infty} S_n x^n - x - 1 = x(\sum_{n=0}^{\infty} S_n x^n - 1) + x^2 \sum_{n=0}^{\infty} S_n x^n + \sum_{n=0}^{\infty} x^n - x - 1$$

$$\Leftrightarrow G(x) = x(G(x) - 1) + x^2 G(x) + \frac{1}{1-x}$$

$$\Leftrightarrow (1 - x - x^2)G(x) = \frac{1 - x + x^2}{1 - x}$$

$$\Leftrightarrow G(x) = \frac{1 - x + x^2}{(1-x)(1-x-x^2)}$$

$$\Leftrightarrow G(x) = \frac{-1}{1-x} + \frac{2}{1-x-x^2}$$

$$\Leftrightarrow G(x) = \frac{-1}{1-x} + \frac{\frac{1+\sqrt{5}}{5}}{1 - \frac{1+\sqrt{5}}{2}x} + \frac{\frac{1-\sqrt{5}}{5}}{1 - \frac{1-\sqrt{5}}{2}x}$$

Công thức tổng quát của S_n là hệ số của x^n trong G(x)

Vậy $S_n = \frac{1+\sqrt{5}}{5}\left(\frac{1+\sqrt{5}}{2}\right)^n + \frac{1-\sqrt{5}}{5}\left(\frac{1-\sqrt{5}}{2}\right)^n - 1$

- Kết quả lý thuyết:

Input	Output		
0	1	0	1
2	2	0	3
4	5	0	9
6	13	0	25
8	34	0	67
10	89	0	177
12	233	0	465
14	610	0	1219
16	1597	0	3193
18	4181	0	8361
20	10946	0	21891
22	28657	0	57313
24	75025	0	150049
26	196418	0	392835
28	514229	0	1028457
30	1346269	0	2692537

1.2 Cài đặt và chạy thuật toán

Để đếm số lượng phép gán và so sánh trong thuật toán này, ta tiến hành thêm các biến count_assign (để đếm số lượng phép gán) và count_compare (để đếm số lượng phép so sánh) như sau:

```
int fibonacci1(int n, int& count_assign, int& count_compare)
{
  if (++count_compare && n <= 1)
    return 1;
  else
    return fibonacci1(n - 1, count_assign, count_compare) + fibonacci1(n - 2,
    count_assign, count_compare);
}
```

Chỉ có một phép so sánh trong câu lệnh if nên ta tăng biến đếm count_assign trong lệnh if.

Trước khi chạy thuật toán thì sẽ gán các biến count_assign và count_compare bằng 0, tiến hành ghi vào file csv bằng thư viện fstream. Đoạn mã ghi vào file như sau:

```
fstream outdata1;
outdata1.open("output1.csv", ios::out);
outdata1.seekg(std::ios::beg);
outdata1 << "Input,Output" << endl;
int n = 0;
while (n <= 30) {
  int count_assign = 0, count_compare = 0;
  int res = fibonacci1(n, count_assign, count_compare);
  outdata1 << n << "," << res << "," << count_assign << "," << count_compare<<endl;
  n = n + 2;
}
outdata1.close();
```

Kết quả thu được của file output1.csv như sau:

1	Input	Output		
2	0	1	0	1
3	2	2	0	3
4	4	5	0	9
5	6	13	0	25
6	8	34	0	67
7	10	89	0	177
8	12	233	0	465
9	14	610	0	1219
10	16	1597	0	3193
11	18	4181	0	8361
12	20	10946	0	21891
13	22	28657	0	57313
14	24	75025	0	150049
15	26	196418	0	392835
16	28	514229	0	1028457
17	30	1346269	0	2692537

Nhận xét: Kết quả thu được với các bộ kiểm thử bằng cách chạy thuật toán (đã cài đặt bổ sung) trên hoàn toàn giống với kết quả lý thuyết tính toán được.

2 Thuật toán không đệ quy

2.1 Tính toán lý thuyết

```
int fibonacci2(int n)
{
  if (n <= 1)
    return 1;
  int last = 1;
  int nextToLast = 1;
  int answer = 1;

  for (int i = 2; i <= n ; i++)
  {
    answer = last + nextToLast;
    nextToLast = last;
    last = answer;
  }
  return answer;
}
```

- Phép gán:

  Với thuật toán Fibonacci sử dụng đệ quy trên, Ta thấy đầu tiên có 3 biến được khai báo cục bộ và gán bằng 1, tiếp theo trong vòng lặp For có 3 phép gán, vòng lặp được lặp n - 2 + 1 = n - 1 lần, nên i được gán n lần.

  Vậy sẽ có tổng cộng 3(n - 1) + n + 3 = 4n số lần lặp tất cả.

- Phép so sánh:

  Phép so sánh đầu tiên ở câu lệnh if(n <= 1), trong vòng for sẽ có tổng cộng n phép so sánh (i <= n).

  Vậy sẽ có tổng cộng n + 1 phép so sánh.

- Kết quả lý thuyết:

Input	Output		
0	1	0	1
2	2	8	3
4	5	16	5
6	13	24	7
8	34	32	9
10	89	40	11
12	233	48	13
14	610	56	15
16	1597	64	17
18	4181	72	19
20	10946	80	21
22	28657	88	23
24	75025	96	25
26	196418	104	27
28	514229	112	29
30	1346269	120	31

2.2 Cài đặt và chạy thuật toán

Để đếm số lượng phép gán và so sánh trong thuật toán này, ta tiến hành thêm các biến count_assign (để đếm số lượng phép gán) và count_compare (để đếm số lượng phép so sánh) như sau:

```
int fibonacci2(int n, int &count_assign, int &count_compare)
{
  count_assign = 0;
  count_compare = 0;

  if (++count_compare && n <= 1)
    return 1;
  int last = 1; ++count_assign;
  int nextToLast = 1; ++count_assign;
  int answer = 1; ++count_assign;

  ++count_assign;
```

```
  for (int i = 2; ++count_compare && i <= n ; i++, ++count_assign)
  {
    answer = last + nextToLast; ++count_assign;
    nextToLast = last; ++count_assign;
    last = answer; ++count_assign;
  }
  return answer;
}
```

Có một phép so sánh trong câu lệnh if nên ta tăng biến đếm count_assign trong lệnh if. Các phép gán last, nextToLast, answer ta sẽ tăng biến đếm count_compare. Tiếp theo ta đếm các số phép gán i và so sánh trong vòng lặp for bằng cách tăng biến đếm count_assign trong lệnh so sánh i <= n, tăng biến đếm count_compare trước vòng lặp (bởi vì bắt đầu vòng lặp có gán i = 2) và cả ở trong phép i++.

Tiến hành ghi vào file csv bằng thư viện fstream. Đoạn mã ghi vào file như sau:

```
fstream outdata2;
outdata2.open("output2.csv", ios::out);
outdata2.seekg(std::ios::beg);
outdata2 << "Input,Output" << endl;
n = 0;
while (n <= 30)
{
  int count_assign = 0, count_compare = 0;
  int res = fibonacci2(n, count_assign, count_compare);
  outdata2 << n << "," << res << "," << count_assign << "," << count_compare << endl
    ;
  n = n + 2;
}
outdata2.close();
```

Kết quả thu được của file output2.csv như sau:

1	Input	Output		
2	0	1	0	1
3	2	2	8	3
4	4	5	16	5
5	6	13	24	7
6	8	34	32	9
7	10	89	40	11
8	12	233	48	13
9	14	610	56	15
10	16	1597	64	17
11	18	4181	72	19
12	20	10946	80	21
13	22	28657	88	23
14	24	75025	96	25
15	26	196418	104	27
16	28	514229	112	29
17	30	1346269	120	31

Nhận xét: Kết quả thu được với các bộ kiểm thử bằng cách chạy thuật toán (đã cài đặt bổ sung) trên hoàn toàn giống với kết quả lý thuyết tính toán được.

3 Nhận xét, đánh giá 2 thuật toán

Sau khi thực hiện tính toán lý thuyết và kiểm tra 2 thuật toán, ta thấy rằng:

- Số phép gán:

  Trong trường hợp này, thuật toán fibonacci không đệ quy có số phép so sánh nhỏ hơn so với thuật toán đệ quy, cụ thể thuật toán fibonacci không đệ quy có số phép gán $n+1 \in O(n)$, thuật toán fibonacci sử dụng đệ quy có số phép gán $\frac{1+\sqrt{5}}{5}\left(\frac{1+\sqrt{5}}{2}\right)^n + \frac{1-\sqrt{5}}{5}\left(\frac{1-\sqrt{5}}{2}\right)^n - 1 \in O(2^n)$

- Số phép so sánh:

  Trong trường hợp này,thuật toán fibonacci đệ quy có số phép so sánh nhỏ hơn so với thuật toán không đệ quy, cụ thể thuật toán fibonacci đệ quy có số phép so sánh $0 \in O(1)$, thuật toán fibonacci không sử dụng đệ quy có số phép so sánh $4n \in O(n)$.

Vậy, nhìn chung thuật toán Fibonacci không sử dụng đệ quy có độ phức tạp nhỏ hơn thuật toán Fibonacci sử dụng đệ quy. Trên thực tế, ta nên hạn chế sử dụng phương pháp đệ quy để lập trình vì độ phức tạp sẽ lớn hơn khá nhiều.