Phân tích tiệm cận trong Cấu trúc dữ liệu và Giải thuật

Chương trước chúng ta đã cùng tìm hiểu về phân tích lý thuyết và một số khái niệm về độ phức tạp thời gian và độ phức tạp bộ nhớ trong phân tích giải thuật. Chương này mình sẽ trình bày về Phân tích tiệm cận trong Cấu trúc dữ liệu và Giải thuật.

Phân tích tiệm cận là gì ?

Phân tích tiệm cận của một giải thuật là khái niệm giúp chúng ta ước lượng được thời gian chạy (Running Time) của một giải thuật. Sử dụng phân tích tiệm cận, chúng ta có thể đưa ra kết luận tốt nhất về các tình huống trường hợp tốt nhất, trường hợp trung bình, trường hợp xấu nhất của một giải thuật. Để tham khảo về các trường hợp này, bạn có thể tìm hiểu chương **Cấu trúc dữ liệu là gì ?**.

Phân tích tiệm cận tức là tiệm cận dữ liệu đầu vào (Input), tức là nếu giải thuật không có Input thì kết luận cuối cùng sẽ là giải thuật sẽ chạy trong một lượng thời gian cụ thể và là hằng số. Ngoài nhân tố Input, các nhân tố khác được xem như là không đổi.

Phân tích tiệm cận nói đến việc ước lượng thời gian chạy của bất kỳ phép tính nào trong các bước tính toán. Ví dụ, thời gian chạy của một phép tính nào đó được ước lượng là một hàm**f(n)** và với một phép tính khác là hàm **g(n^2)**. Điều này có nghĩa là thời gian chạy của phép tính đầu tiên sẽ tăng tuyến tính với sự tăng lên của n và thời gian chạy của phép tính thứ hai sẽ tăng theo hàm mũ khi n tăng lên. Tương tự, khi n là khá nhỏ thì thời gian chạy của hai phép tính là gần như nhau.

Thường thì thời gian cần thiết bởi một giải thuật được chia thành 3 loại:

- **Trường hợp tốt nhất**: là thời gian nhỏ nhất cần thiết để thực thi chương trình.
- **Trường hợp trung bình**: là thời gian trung bình cần thiết để thực thi chương trình.
- **Trường hợp xấu nhất**: là thời gian tối đa cần thiết để thực thi chương trình.

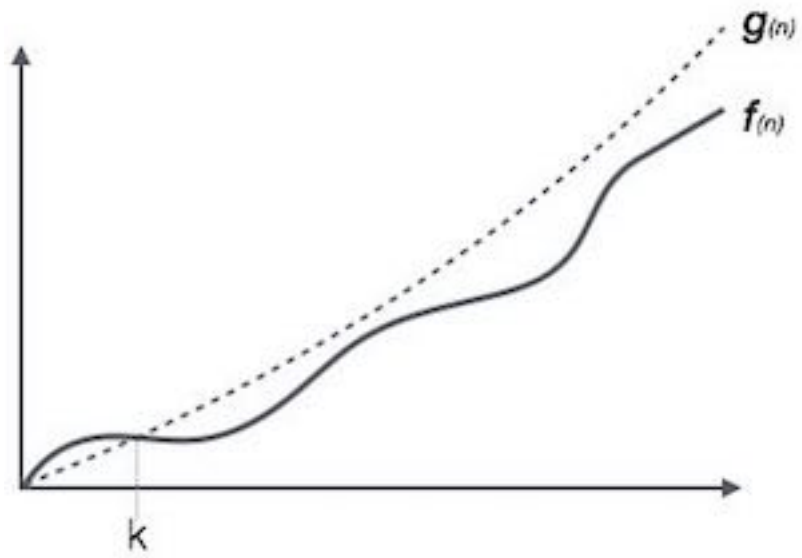
Asymptotic Notation trong Cấu trúc dữ liệu và giải thuật

Dưới đây là các Asymptotic Notation được sử dụng phổ biến trong việc ước lượng độ phức tạp thời gian chạy của một giải thuật:

- O Notation
- Ω Notation
- θ Notation

Big Oh Notation, O trong Cấu trúc dữ liệu và giải thuật

O(n) là một cách để biểu diễn tiệm cận trên của thời gian chạy của một thuật toán. Nó ước lượng độ phức tạp thời gian trường hợp xấu nhất hay chính là lượng thời gian dài nhất cần thiết bởi một giải thuật (thực thi từ bắt đầu cho đến khi kết thúc). Đồ thị biểu diễn như sau:

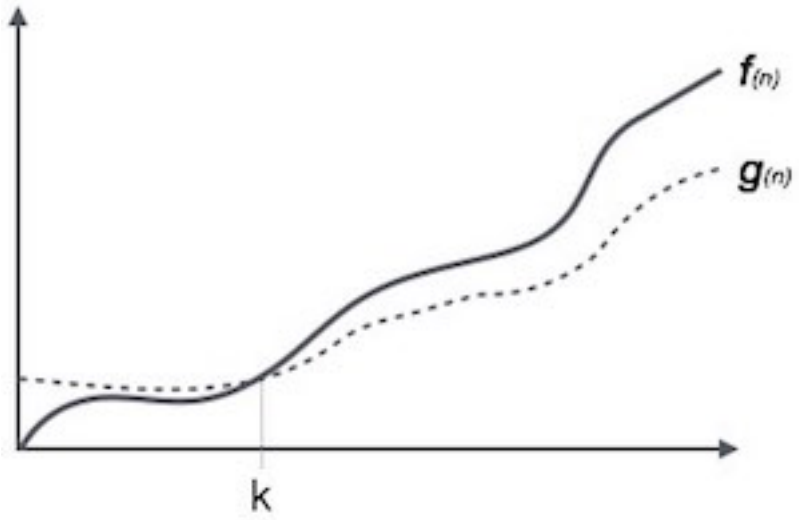


Ví dụ, gọi f(n) và g(n) là các hàm không giảm định nghĩa trên các số nguyên dương (tất cả các hàm thời gian đều thỏa mãn các điều kiện này):

```
O(f(n)) = { g(n) : nếu tồn tại c > 0 và n₀ sao cho g(n) ≤ c.f(n) với mọi n > n₀.
}
```

Omega Notation, Ω trong Cấu trúc dữ liệu và giải thuật

The Ω(n) là một cách để biểu diễn tiệm cận dưới của thời gian chạy của một giải thuật. Nó ước lượng độ phức tạp thời gian trường hợp tốt nhất hay chính là lượng thời gian ngắn nhất cần thiết bởi một giải thuật. Đồ thị biểu diễn như sau:

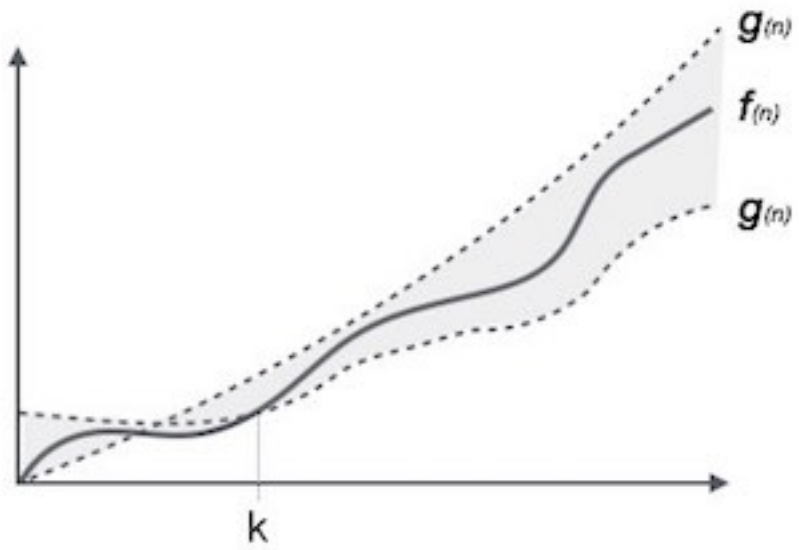


Ví dụ, với một hàm f(n):

```
Ω(f(n)) ≥ { g(n) : nếu tồn tại c > 0 và n₀ sao cho g(n) ≤ c.f(n) với mọi n > n₀.
}
```

Theta Notation, θ trong Cấu trúc dữ liệu và giải thuật

The θ(n) là cách để biểu diễn cả tiệm cận trên và tiệm cận dưới của thời gian chạy của một giải thuật. Bạn nhìn vào đồ thì sau:




```
θ(f(n)) = { g(n) nếu và chỉ nếu g(n) =  O(f(n)) và g(n) = Ω(f(n)) với mọi n >
n₀. }
```


Một số Asymptotic Notation phổ biến trong cấu trúc dữ liệu và giải thuật

hằng số	-	O(1)
logarit	-	O(log n)
Tuyến tính (Linear)	-	O(n)
n log n	-	O(n log n)
Bậc hai (Quadratic)	-	$O(n^2)$
Bậc 3 (cubic)	-	$O(n^3)$
Đa thức (polynomial)	-	$n^{O(1)}$
Hàm mũ (exponential)	-	$2^{O(n)}$