quy với nửa trái hoặc nửa phải mảng cộng một. Giả thiết rằng n lũy thừa của 2. Vậy hệ thức truy hồi tính $C(n)$ như sau:

$$C(n) = \begin{cases} 1 & if\ n = 1 \\ C(n/2) + 1 & if\ n \geq 2. \end{cases}$$

Giải hệ thức truy hồi:

$$\begin{aligned} C(n) &= C(n/2) + 1 \\ &= (C(n/4) + 1) + 1 \\ &= C(n/4) + 2 \\ &= (C(n/8) + 1) + 2 \\ &= C(n/8) + 3 \\ &= C(n/2^k) + k \\ &= C(1) + \mathrm{k} \\ &= 1 + \log n. \end{aligned}$$

Vậy $C(n) = O(\log n)$.

5.3. Các bài toán ứng dụng

5.3.1. Nhân hai số nguyên lớn

Trong Chương 1, chúng ta đã giới thiệu bài toán nhân hai số nguyên (xem mục 1.5.2). Có nhiều giải thuật khác nhau để nhân hai số nguyên. Giải thuật nhân thông thường có độ phức tạp $O(n^2)$.

Giải thuật chia để trị

Để áp dụng kỹ thuật chia để trị, số chữ số của hai số nguyên cần nhân phải bằng nhau và phải bằng lũy thừa của 2. Nếu hai điều kiện trên không thoả mãn, thì chúng ta thêm vào bên trái hai số các chữ số 0.

Giả sử hai số cần nhân là a và b, mỗi số đều có n chữ số và n là lũy thừa của 2. Gọi a_L và a_R là hai nửa trái và phải của số a, tương tự b_L và b_R là hai nửa trái và phải của b. Khi đó, phép nhân a và b được thực hiện như sau:

$$\begin{aligned} a \times b &= (a_L \times 10^{n/2} + a_R) \times (b_L \times 10^{n/2} + b_R) \\ &= a_L \times b_L \times 10^n + a_L \times b_R \times 10^{n/2} + a_R \times b_L \times 10^{n/2} + a_R \times b_R \\ &= a_L \times b_L \times 10^n + (a_L \times b_R + a_R \times b_L) \times 10^{n/2} + a_R \times b_R \end{aligned}$$

Vậy ba bước chia để trị được thực hiện như sau:

Chia: Chia mỗi số nguyên có *n* chữ số thành hai số nguyên có *n*/2 chữ số.

Trị: Thực hiện các phép nhân hai số nguyên có *n*/2 chữ số bởi các lời gọi đệ quy.

Kết hợp: Thực hiện các phép cộng hai số nguyên có *n*/2 chữ số.

Vậy giải thuật chia để trị được xây dựng như sau:

```
algorithm NhanSoNguyen(a, b, n)
input: hai số nguyên a và b có n chữ số
output: tích a × b
begin
   if (n = 1) then
      return a × b
   else
      Chia a thành hai nửa aL và aR, b thành hai nửa
      bL và bR
      p1 ← NhanSoNguyen(aL, bL, n/2)
      p2 ← NhanSoNguyen(aL, bR, n/2)
      p3 ← NhanSoNguyen(aR, bL, n/2)
      p4 ← NhanSoNguyen(aR, bR, n/2)
      return (p1 × 10^n + (p2 + p3) × 10^(n/2) + p4)
   endif
end
```

Như vậy, để nhân hai số *a* và *b* có *n* chữ số, cần sử dụng bốn phép nhân và ba phép cộng các số có *n*/2 chữ số. Phép cộng hai số cần O(*n*) thời gian. Từ đó, chúng ta có hệ thức truy hồi tính thời gian thực thi của giải thuật như sau:

$$C(n) = \begin{cases} c & \text{if } n = 1 \\ 4C(n/2) + dn & \text{if } n > 1 \end{cases}$$

trong đó, *c* và *d* là các hằng số. Dễ dàng giải hệ thức truy hồi:

$$\begin{aligned} C(n) &= 4C(n/2) + dn \\ &= 4^2 C(n/4) + 4d(n/2) + dn \\ &= 4^3 C(n/8) + 2^2 dn + 2dn + dn \end{aligned}$$

$$= 4^i C(n/2^i) + dn\sum_{k=0}^{i-1} 2^k$$

$$= 4^{\log n} C(1) + dn\sum_{k=0}^{\log n - 1} 2^k$$

$$= cn^2 + dn\frac{1-2^{\log n-1+1}}{1-2}$$

$$= cn^2 + dn^2 - dn.$$

Vậy độ phức tạp giải thuật chia để trị cũng như độ phức tạp giải thuật nhân thông thường O(n^2).

Giải thuật Karatsuba

Giải thuật Karatsuba cải tiến giải thuật chia để trị bằng cách giảm số phép nhân các nửa a và b bằng các biến đổi dưới đây.

Đặt: $p = a_L \times b_L$

$q = a_R \times b_R$

$r = (a_L + a_R) \times (b_L + b_R)$

Khi đó, ta có:

$$a \times b = a_L \times b_L \times 10^n + (a_L \times b_R + a_R \times b_L) \times 10^{n/2} + a_R \times b_R$$

$$= p \times 10^n + (r - p - q) \times 10^{n/2} + q.$$

Thay vì thực hiện bốn phép nhân và ba phép cộng các nửa của a và b, chỉ cần thực hiện ba phép nhân và sáu phép cộng (trừ). Như vậy, giảm được số phép nhân, nhưng tăng số phép cộng. Tuy nhiên, độ phức tạp của giải thuật chủ yếu phụ thuộc vào phép nhân, vì vậy giảm được đáng kể độ phức tạp của giải thuật. Giải thuật chia để trị được cải tiến như sau:

```
algorithm Karatsuba(a, b, n)
input: hai số nguyên a và b có n chữ số
output: tích a × b
begin
   if (n = 1) then
      return a × b
   else
      Chia a thành hai nửa aL và aR, b thành hai nửa bL và bR
      p ← Karatsuba(aL, bL, n/2)
      q ← Karatsuba(aR, bR, n/2)
```