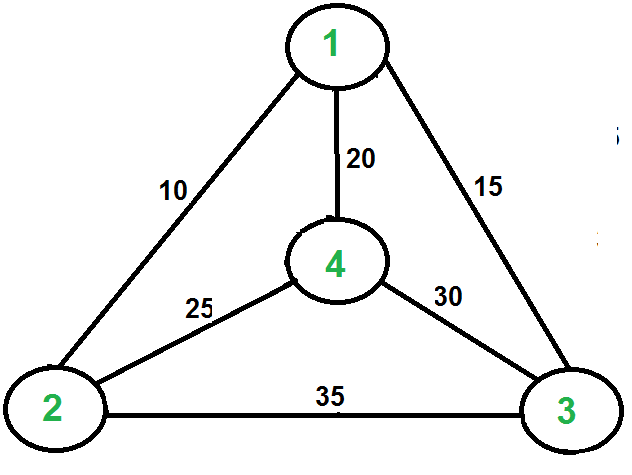
TÌM HIỂU GIẢI THUẬT TÌM ĐƯỜNG ĐI NGẮN NHẤT

Bài toán:

Một người du khách đi tham quan tất cả các địa điểm dụ lịch (mỗi địa điểm một lần) với đường đi ngắn nhất



Cách 1: Sử dụng hoán vị

1) Giả sử bắt đầu tại thành phố số 1

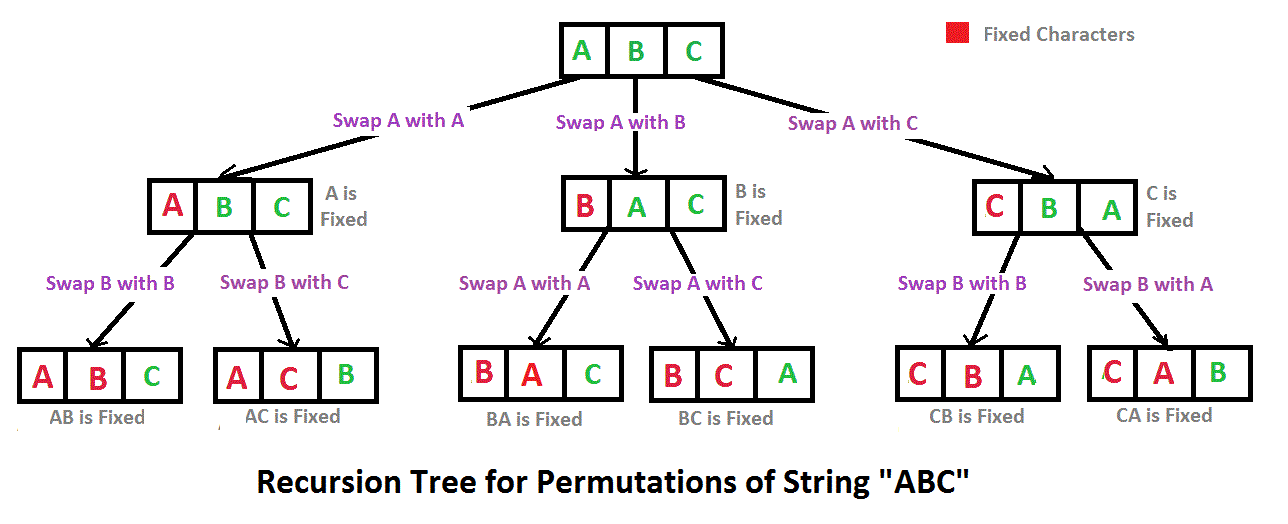
2) Thực hiện hoán vị (n-1)! lần thứ tự các thành phố cần đi tiếp theo

3) Tính toán quãng đường cần đi của mỗi hoán vị vào tìm quãng đường ngắn nhất

4) Trả về kết quả của lần hoán vị có quãng đường ngắn nhất.

Đây là cách đơn giản, độ phức tạp lớn O(n!) nên chỉ được dùng khi số thành phố nhỏ.

Để giải quyết bài toán này ta cần phải tìm hiểu được cách làm sao tìm được tất cả các hoán vị đường đi một cách tối ưu nhất. Do đó ta sẽ bắt đầu với bài toán tím hoán vị một chuỗi.



Một chuỗi có độ dài n sẽ có n! các hoán vị. Ví dụ trên là ví dụ cho hoán vị của chuỗi ABC.

Sử dụng đệ quy để tìm tất cả các hoán vị

1) Bát đầu từ phần tử đầu tiên, hoán vị chính bản thân nó

2) Thực hiện hoán vị với chuỗi bắt đầu từ phần tử tiếp theo

3) Quá trình gọi hàm hoán vị lặp đi lặp lại đến khi chuỗi chỉ còn 1 phần tử.

/\*\*

\*

\* @brief - calculate all permutation of string a begin from l to r

\* @param *a* - String want to permute

\* @param *l* - Starting index of the string

\* @param *r* - Ending index of the string

\* @retval - None

\*

\*/

void permute(char \**a*, int *l*, int *r*)

{

int i;

if (*l* == *r*)

{

//Code here

/\* DBG("permutation city:");

for (int j = 0; j <= r; j++)

{

printf("%d\t", \*(a + j));

}

printf("\n"); \*/

/\* printf("%s\n",a); \*/

mincost(*a*, 0, *r*);

}

else

{

for (i = *l*; i <= *r*; i++)

{

swap((*a* + *l*), (*a* + i));

permute(*a*, *l* + 1, *r*);

swap((*a* + *l*), (*a* + i));

}

}

}