Đại Học Quốc Gia Thành Phố Hồ Chí Minh
Trường Đại Học Công Nghệ Thông Tin

CẤU TRÚC DỮ LIỆU VÀ GIẢI THUẬT

LỚP: IT003.M21.ANTN

Giảng viên: Nguyễn Thanh Sơn

Sinh viên: Nguyễn Đức Vương MSSV: 21522809

- Ý tưởng: sử dụng phương pháp Meet in the middle , bảng băm.

```
string mang;
string r;
string sumbit;
string S="abcdefghijklmnopqrstuvwxyz012345";
int firstBit,secondBit;
int length;
int AmountCharacter;

queue<int> index_int;
unordered_map<string,string> Hash[60];
char Cout(string x);
int find(char x);
int makeEqualLength(string &str1, string &str2);
string addBitStrings( string first, string second );
string Bit(int n);
string CreateBit(string base);
string SumBit(string res);
string ReceateBit(string sum);
queue<int> StringToIndex(string index);
string CipherToPlain(queue<int> q);

void Print_Permutation(vector<vector<int> > v);
class Solution {
public:
   vector < vector <int> > res;
   void solve(int n, int k, vector <int> temp, int start = 1){
      if(temp.size() == k){
         res.push_back(temp);  return;
      }
      for(int i = start; i <= n; i++){
         temp.push_back(i);
         solve(n, k, temp, i + 1);
         temp.pop_back();
```

- Tóm tắt:

- Đầu vào gồm N=5*C dòng, mỗi dòng chứa một chuỗi ký tự có độ dài tối đa là 12 dùng để tạo ra bảng mã hóa. Ngoài ra còn một chuỗi ký tự có độ dài C là mật khẩu sau khi đã được mã hóa.
- 40 chuỗi ký tự (lấy N=8) được nhập vào sau đó được chuyển về dạng bit rồi lưu vào mảng arr, chuỗi ký tự đầu tiên sẽ được lưu ở vị trí thứ 0(arr[0]), chuỗi ký tự cuối cùng sẽ được lưu ở vị trí thứ 39 (arr[39]).
- Lưu $\sum_{k=1}^{40} C_n^k$ (N = 5*C) tổ hợp của các vị trí bit được bật trong mật khẩu ban đầu được tạo ra từ bảng nhập, tính tổng và sau đó được lưu vào Hash[40] , giá trị sẽ là tổng của các chuỗi bit còn khóa sẽ là vị trí của các chuỗi bit ở mảng arr được cộng vào. Vị trí của các các chuỗi bit ở mảng arr sẽ được chuyển về dạng string để làm khóa.
- Hash[0] sẽ lưu giá trị của tổ hợp chập 1 của 40, Hash[39] sẽ lưu giá trị của tổ hợp chập 40 của 40 , mật khẩu được nhập vào sẽ chuyển về dạng bit và tìm kiếm trong mảng Hash[40].
- Hàng đợi index_int (dòng 21) được dùng để lưu vị trí của các chuỗi bit ở vị trí thứ i ở trong mảng arr được cộng vào.
- Hàm find (dòng 24) được dùng để tìm kiếm vị trí của kí tự trong chuỗi S, sau đó vị trí ở dạng thập phân sẽ được chuyển về dạng nhị phân.
- Hàm AddBitsString (dòng 25) được dùng để cộng hai chuỗi bit vào với nhau (chuỗi có độ dài 40 bit) .
- Hàm Bit (dòng 27) được dùng để chuyển kí tự được nhập vào thành chuỗi bit con có độ dài là 5 sau đó được hợp lại thành chuỗi bit có độ dài là 40 và lưu vào mảng arr[40].
- Hàm RecreateBit được dùng để chuyển chuỗi bit có độ dài 40 thành chuỗi ký tự có độ dài là 8.
- Hàm StringToIndex (dòng 31) được dùng để chuyển vị trí từ dạng string sang dạng int và được lưu vào hàng đợi.
- Hàm CipherToPlain (dòng 32) được dùng để tạo ra một chuỗi bit với các bit được bật là vị trí của các arr được lưu ở hàng đợi.
- Hàm Print_Permutation (dòng 34) được dùng để tạo ra các tổ hợp chập k của n với (k từ 1 đến 40, n bằng 40).

- Đánh giá độ phức tạp: O(n) với n = $\sum_{k=1}^{40} C_n^k$ (N = 5*C), thuật toán phải tạo ra các tổ hợp có thể có sau đó mới tìm kiếm tổng bit trùng với mật khẩu đã được mã hóa.
- Nhận xét:

- thuật toán hoạt động tốt đối với các mật khẩu có số lượng kí tự ít, nếu mật khẩu có độ dài lớn hơn 5 thì sẽ không hoạt động được bởi unordered_map sẽ bị tràn.
- Hạn chế: chưa tìm ra cách tối ưu để lưu giá trị và khóa của chuỗi bit.