Técnicas de Algoritmos

- Força Bruta
- Pesquisa Exaustiva
- Dividir e Conquistar
- Gulosos
- Backtracking (retrocesso)
- Programação Dinâmica

Backtracking

https://sites.google.com/site/seminariobacktracking/material-de-apoio

Backtracking é um algoritmo baseado em estrutura de dados, tem como meta resolver o problema no menor intervalo de tempo, não levando em consideração o esforço para alcançar a solução, usando recursividade.

Backtracking é um algoritmo genérico que busca por força bruta, soluções possíveis para problemas computacionais (tipicamente problemas de satisfações à restrições).

De maneira incremental, busca por candidatos à soluções e abandona cada candidato parcial C quando C não pode resultar em uma solução válida.

Backtracking é um tipo de algoritmo que representa um refinamento da busca por força bruta, em que múltiplas soluções podem ser eliminadas sem serem explicitamente examinadas. O termo foi cunhado pelo matemático estado-unidense D. H. Lehmer na década de 1950.

Quando sua busca chega a uma extremidade da estrutura de dados, como um nó terminal de uma árvore, o algoritmo realiza um retrocesso tipicamente implementado através de uma recursão.

Backtracking é aplicável na solução de vários problemas conhecidos, dentre os quais podem-se destacar:

Exemplos de Problemas:

- N-Rainhas - https://www.youtube.com/watch?v=ckC2hFdLff0
- Passeio do cavalo
- Labirinto
- Caixeiro Viajante
- Entrevista com o usuário (revisar os eventos)
- Sala de aula (passos)
- Sudoku (uma grade de 9 x 9 (matriz de 81 posições) https://www.youtube.com/watch?v=X_8LNpuJbrU
- Resta 1
- Cubo Mágico
- 08 Damas - https://pt.wikipedia.org/wiki/Problema_das_oito_damas
- Outros - https://www.youtube.com/watch?v=xI2VP5sAx3c

Exemplo de Algoritmo

```
bool acabou = FALSE;
backtrack(int a[], int k, int n) {
   int c[MAXCANDIDATOS];  /* Candidatos para a próxima posição */
   int ncandidatos;       /* Número de candidatos para a próxima posição */
   int i;                 /* Contador */
   if (e_uma_solucao(a, k, n)) {
      processar_solucao(a, k, n);
   } else {
      k = k + 1;
      construir_candidatos(a, k, n, c, &ncandidatos);
      for (i=0; i<ncandidatos; i++) {
         a[k] = c[i];
         backtrack(a, k, n);
         if (acabou) return;
      }
   }
}
```

Recursividade

Recursão é um método de programação no qual uma função pode chamar a si mesma

// Multiplicar dois números 7 x 5 - Solução Recursividade

```
#include <stdio.h>
int multiplica(int a, int b){
  int resp = 0;
  if ( b == 0 ) return resp;
  resp = a + multiplica(a, b-1);
}

int main(void){
  int resposta, a = 7, b =5;
  resposta = multiplica(7,5);
  printf("Resultado %d x %d = %d ", resposta);
}
/* PILHA
    retorno função multiplica(7,0) = 0; resp = 0;
    retorno função multiplica(7,1) => 7 + multiplica(7,0) => 7 +  0 = 7;  resp = 6;
    retorno função multiplica(7,2) => 7 + multiplica(7,1) => 7 +  7 = 14; resp = 14;
    retorno função multiplica(7,3) => 7 + multiplica(7,2 )=> 7 + 14 = 21; resp = 21;
    retorno função multiplica(7,4) => 7 + multiplica(7,3 )=> 7 + 21 = 28; resp = 28;
    retorno função multiplica(7,5) => 7 + multiplica(7,4 )=> 7 + 28 = 35; resp = 35;


    https://www.youtube.com/watch?v=ny0LDwUXbIg
    https://www.youtube.com/watch?v=ztYmMIZ2x_k
https://stock.adobe.com/mx/search?k=recursivo&load_type=tagged%20keyword&prev_url=detail
    https://giphy.com/explore/fibonacci
*/
```

// Torre de Hanoi – Solução Recursividade

```
#include <stdio.h>
#include <stdlib.h>
void hanoi(int n, char a, char b, char c)
{
 /* mova n discos do pino a para o pino b usando
     o pino c como intermediario                    */
     if (n == 1)
       printf("mova disco %d de %c para %c\n", n, a, b);
  else
  {
      hanoi(n - 1, a, c, b);                        // H1
      printf("mova disco %d de %c para %c\n", n, a, b);
      hanoi(n - 1, c, b, a);                        // H2
    }
  }
  int main(void)
  {
    int numDiscos;
    scanf("%d", &numDiscos);
    hanoi(numDiscos, 'A', 'B', 'C');
    return 0;
  }
```

https://sites.google.com/a/liesenberg.biz/cjogos/home/materiais-de-apoio/topicos-relativos-a-c/recursao/torre-de-hanoi