UNIVERSIDADE DE SÃO PAULO

INSTITUTO DE CIÊNCIAS MATEMÁTICAS E DE COMPUTAÇÃO

SCC0218 – Algoritmos Avançados e Aplicações

Coloração de Mapas (Backtracking)

Elias Italiano Rodrigues – 7987251
Rodolfo Megiato de Lima - 7987286

1. Resumo

O objetivo desse trabalho é a utilização do algoritmo de Backtracking para colorir mapas, levando em consideração que duas regiões vizinhas não podem estar pintadas com a mesma cor. Para isso, foi utilizado um grafo não direcionado que tem seus vértices representando as regiões e as arestas as fronteiras. O uso do Backtracking tradicional, em certos grafos de tamanho maior, pode-se tornar inviável, por isso uma variação do algoritmo foi implementada, com a utilização de algumas heurísticas, como a Verificação Adiante, os Mínimos Valores Remanescentes e levando em conta o grau do vértice (região) no grafo.

2. Detalhes da Implementação

A implementação do problema foi realizada com um TAD ***color_map*** escrito em linguagem C. O TAD contém uma estrutura de dados interna ***map_t*** que armazena todos os dados necessários e possui as seguintes operações:

- Inicialização da estrutura de dados;
- Leitura da entrada dos dados;
- Saída dos dados para a coloração do mapa;
- Execução do algoritmo de Backtracking;
- Impressão de “debug”;
- Liberação da memória ocupada.

A execução do algoritmo de Backtracking é realizada pela função:

int color_map_make(int method);

que recebe como parâmetro o método que pode ser:

0 – Backtracking simples sem poda;
1 – Backtracking com Verificação Adiante;
2 – Backtracking com Verificação Adiante e Mínimos Valores Remanescentes;
3 – Backtracking com Verificação Adiante e Mínimos Valores Remanescentes com desempate por grau da região.

3. Resultados Obtidos

Executando o algoritmo com o mapa dos Estados Unidos da América, obtivemos os seguintes resultados comparativos da quantidade de atribuições de cor realizadas:

Mapa	Método 0	Método 1	Método 2	Método 3
EUA	4.088.658	1.663.292	51	51

4. Conclusão

O algoritmo de Backtracking para ser utilizado com grandes entradas de dados é inviável, mas para pequenos conjuntos funciona de maneira aceitável, podendo ser aprimorado com algumas heurísticas que realmente diminuem muito o número de iterações realizadas, nesse caso, só fazendo o ideal para completar seu objetivo.