Relatório sobre o Paradigma de Força Bruta e um Solver de Sudoku em Python

Nome: Daniel Antônio de Sá **Disciplina: PAA**

O paradigma de força bruta é uma técnica de solução de problemas que consiste em examinar sistematicamente todas as possíveis soluções até encontrar a solução correta. É uma abordagem geralmente utilizada quando não há maneira eficiente de reduzir o espaço de busca das soluções. Neste relatório, abordaremos o paradigma de força bruta e apresentaremos um exemplo de aplicação prática em um solver de Sudoku desenvolvido em Python.

O paradigma de força bruta é uma abordagem direta e simples para resolver problemas. Ele envolve a geração sistemática de todas as possíveis soluções e a verificação de cada uma delas até encontrar a solução correta ou a resposta desejada. No entanto, essa abordagem é muitas vezes ineficiente e consome muitos recursos computacionais, especialmente para problemas complexos, devido à sua natureza exponencial.

No contexto do paradigma de força bruta, geralmente ocorre um processo recursivo ou iterativo de geração e teste de soluções. Em cada passo, o algoritmo gera uma escolha possível e a testa para ver se é uma solução válida. Se for, a solução é aceita; caso contrário, o algoritmo volta atrás (backtracking) e tenta outra escolha até encontrar uma solução ou esgotar todas as opções.

O Sudoku é um quebra-cabeça numérico que envolve preencher uma grade 9x9 com números de 1 a 9 de forma que cada linha, coluna e sub grade 3x3 contenham todos os números de 1 a 9 sem repetições. Resolver um Sudoku é um problema que se encaixa perfeitamente no paradigma de força bruta, pois requer a geração e validação sistemática de todas as combinações possíveis.

O sistema é iniciado através do comando python sudoku.py teste.txt, em que o arquivo teste.txt contém a entrada do tabuleiro inicial do jogo. Após a leitura desse arquivo, uma matriz de inteiros é criada para representar o tabuleiro, que será percorrida e manipulada durante a resolução do Sudoku. A leitura do arquivo é realizada utilizando a biblioteca 'argparse'.

Após a leitura do tabuleiro, a função solver é chamada, sendo a função principal do sistema. Ela começa sua execução verificando a posição (0, 0) do tabuleiro e verifica se o valor é igual a zero. Caso contrário, a função verifica se o tabuleiro é válido e encerra o processo, retornando recursivamente o tabuleiro válido, caso seja válido. Se não for válido, a função chama recursivamente o solver para a próxima posição.

Quando o valor da célula é igual a zero, a função verifica se a soma da linha atual é maior do que 45. Se isso ocorrer, significa que a linha está incorreta e a função realiza o processo de desfazer (backtracking). O mesmo procedimento é realizado para verificar a coluna e o quadrante.

O próximo passo é encontrar as possíveis soluções para a posição atual. Para isso, uma função é utilizada para determinar quais valores podem ser atribuídos à célula, levando em consideração os preenchimentos anteriores. Em seguida, o algoritmo testa as combinações em profundidade, ou seja, ele parte do pressuposto de que o primeiro elemento do array de possíveis candidatos é o candidato perfeito e testa essa escolha. Se for uma solução válida, ela é impressa. No entanto, se não for o valor correto, o algoritmo realiza as verificações necessárias, desfaz a escolha (backtracking) retornando o valor para zero, e testa o próximo elemento do array de candidatos.

O principal ponto forte do algoritmo é sua garantia de encontrar uma solução válida para o Sudoku, enquanto seu ponto fraco principal é sua ineficiência devido à necessidade de testar todas as combinações possíveis, tornando-o impraticável para quebra-cabeças maiores ou quando a resolução rápida é necessária.

Em resumo, este sistema de resolução de Sudoku utiliza a técnica de força bruta, testando sistematicamente todas as combinações possíveis para preencher o tabuleiro, verificando a validade de cada escolha e utilizando backtracking quando necessário para encontrar a solução correta.