Project Sudoku

Manual do Usuário

Bruno Netto

July 21, 2018

Contents

1 Introductory Chapter **1**
1.1 Guia dos Parâmetros 1
1.1.1 Sudoku Reader 1
1.1.2 Solver Funcs 2
1.1.3 Solver 3
1.1.4 Generator 3

1

Introductory Chapter

1.1 Guia dos Parâmetros

Cada Seção desse capítulo estará focada em um arquivo .py diferente, e comentará sobre os diferentes parâmetros e seus esperados formatos;

1.1.1 Sudoku Reader

Nesse arquivo estão localizadas as funçoes responsáveis por ler um arquivo txt e extrair os sudokus contidos nele. O esperado é que ele esteja na mesma forma que o do problema 96 do Project Euler, que pode ser encontrardo nesse link : `https://projecteuler.net/problem=96`.

- A primeira função, "sudoku_reader",recebe o nome de um arquivo txt em string, que esteja contido na mesma pasta que o arquivo Sudoku Reader.py. E é responsável por extrair a informação contida no arquivo e transformar em uma string, removendo parte do texto que não é importante, como "GRID XX".

- Já a segunda, "sudoku_format" ,recebe uma string e altera o tipo do numeros de string para lista de listas, e remove a ultima palavra de cada linha (incluindo da ultima!), pois são os '\n'. O recomendado é colocar um '\n' no final do arquivo.

- A terceira e última funçao, "puzzles", aplica a segunda na primeira e retorna uma lista contendo uma lista pra cada sudoku no arquivo txt.

1.1.2 Solver Funcs

Aqui estão as funções que fazem parte do processamento dos dados de cada sudoku; como por exemplo contar o numero de movimentos, determinar qual movimento é válido e até mesmo escolher qual o proximo numero a se chutar.

- A primeira função, "Num_Moves", recebe uma das listas contidas no retorno da função "puzzles" e conta o numero de zeros, visto que, isso determina o numero de movimentos corretos até se chegar na solução do sudoku.

- A segunda, "All_moves", é uma função que gera uma lista com 81 listas, representando cada casa do sudoku, e em em cada uma dessas 81 assoscia aos numeros de 1 a 9. A partir desta lista é que vamos marcando quais numeros não podem ser colocados em cada lugar, de maneira que se um lugar já está preenchido, entao a sua casa, de index determinado da esquerda para a direita, de cima para baixo, terá uma lista vazia.

- A terceira função, "available_moves", é a que verifica os numeros já preenchidos e os remove das listas dos quadrados da mesma linha, coluna e sub-quadrado. Esta é uma das funções mais pesadas, e por isso o foco era só rodar ela no começo do sudoku, no pré-processamento.

- A proxima função, "next_move", recebe essa lista com os movimentos possiveis retornada pela "available_moves" e procura pelas listas com menos possibilidades, com a finalidade de determinar o proximo movimento. Retornando o valor do proximo movimento, se houver, e sua coordenada.

- A quinta função, "next_branch", recebe o parâmetro branch, que é responsável por nos localizar na árvore das possibilidades até a solução do sudoku. A função utiliza esse parâmetro como uma bússola que aponta para o próximo chute, retornando-o, e evitando circulos viciosos.

- A sexta função, "refresh_guess_list" é uma "available_moves" que altera as linhas, colunas e sub_quadrados apenas dos afetados por um dado movimento. Evitando recalcular tudo nas casas que não altera nada.

- A sétima função é uma espécie de copy para listas de listas. Útil para evitar o uso de um deepcopy sobre a guess_list. Que é necessário para o backtracking.

- As proximas 3 funções verificam sobre a guess_list a n-ésima linha, coluna ou sub-quadrado, retornando tambem cada um de seus indexes. E são utilizadas pela ultima função do arquivo.

- A função "check_singles" é responsável por verificar cada linha, coluna e sub-quadrado e retornar as aparencias de numeros que só tem uma alternativa de se jogar. Ela pode ser alterada para implementar técnicas como x-wing.

1.1.3 Solver

Aqui estão as duas funções que são responsáveis por resolver os sudokus. Lembrando que o formato do tabuleiro é uma lista de listas, onde cada uma dessas segundas são as linhas do tabuleiro; Contendo numeors de 0 a 9, com os 0 representando os espaços vazios e os numeros de 1 a 9 são eles mesmos.

- A primeira função, "sudoku_starter",recebe como parâmetro um dos tabuleiros, e é responsável por extrair as informações deste, como por exemplo o numero de zeros, oo proximo movimento. Bem como criar a variável utilizadas para backup do tabuleiro, com o nome de backup, e a variável para salvar o galho da árvore, branch. Nessa função tambem são feitos os movimentos sugeridos pela check_single, alem dos movimentos triviais onde só tem uma alternativa no quadrado, sugeridos pela next_move, até eles acabarem. Com isso temos 2 possibilidades, ou o sudoku está resolvido, e o numero de 0 é 0, onde retornamos o tabuleiro completo. Ou então, utilizamos da memoria para salvar o tabuleiro e chutar um dos valores no quadrado com menos alternativas, passando para a proxima função.

- Nesta segunda função, "sudoku_solver", é onde é feito o backup, e começa o backtracking toda vez que chegarmos em um sudoku sem movimentos validos mas que ainda possua zeros. Ela não possui a checagem da função "check_singles" pois gera muita perda de tempo devido a intensa recursividade.

1.1.4 Generator

Nesse arquivo estão salvos 3 exemplos de sudoku e suas soluções, alem de funções que pegam essas soluções e removem, aleatoriamente, numeros de

maneira que a solução final seja a mesma e única.

- A primeira função, "unique_sudoku",recebe um tabuleiro qualquer de sudoku, e o resolve percorrendo dois caminhos opostos, e se a resposta for a mesma então ela é única.
- Já a segunda, "create_from_solution", recebe como parâmetro um numero n, que é a quantidade de numeros removidos do tabuleiro, e um tabuleiro resolvido. Deste, ela 'escolhe' n numeros e os transforma em zeros.
- A terceira função, "sudoku_creator", aplica a segunda função em algum tabuleiro e testa sua unicidade até antes do tabuleiro perder a unicidade. Note que ela recebe de parametro um numero de 0 a 2, que escolhe dentre os 3 tabuleiros, já salvos, para um ponto de partida.
- A ultima função, alem do numero de 0 a 2, para determinar a seed, tambem recebe um numero de iterações. Esta função repete a "sudoku_creator" guardando os minimos locais até atingir 17 dicas, ou até acabarem as iterações.