Trabalho 1 - Programação Linear e Inteira

João Lucas Duim
Raphael Felberg Levy

11 de Abril de 2022

1 Introdução

Descrição do Problema

Sudoku é um quebra-cabeça lógico-combinatório que compreende uma tabela de números 9×9 (caso clássico) que deve ser preenchida de acordo com algumas regras:

- Dentro de qualquer uma das 9 caixas 3×3 individuais, cada um dos números de 1 a 9 deve ser encontrado.
- Dentro de qualquer coluna da grade 9×9, cada um dos números de 1 a 9 deve ser encontrado.
- Dentro de qualquer linha da grade 9×9, cada um dos números de 1 a 9 deve ser encontrado.

O gerador de quebra-cabeça fornece uma grade parcialmente completa, que para um quebra-cabeça bem colocado tem uma única solução.

Os jornais franceses apresentavam variações dos quebra-cabeças de Sudoku no século 19, e o quebra-cabeça apareceu desde 1979 em livros de quebra-cabeças sob o nome de Number Place. No entanto, o Sudoku moderno só começou a ganhar popularidade em 1986, quando foi publicado pela empresa japonesa de quebra-cabeças Nikoli sob o nome Sudoku, que significa "número único". Ele apareceu pela primeira vez em um jornal dos EUA, e depois no The Times (Londres), em 2004, graças aos esforços de Wayne Gould, que desenvolveu um programa de computador para produzir rapidamente quebra-cabeças únicos.

2 Modelagem do Sudoku Clássico

Modelaremos um Sudoku 9×9 utilizando programação linear.

Primeiramente, sejam as variáveis de decisão binárias:

$$x_{ijk} = \begin{cases} 1, \text{ se o elemento } (i, j) \text{ da matriz do Sudoku é o número inteiro } k \\ 0, \text{ caso contrário.} \end{cases}$$

Agora, desejamos que a solução satisfaça as regras do jogo.

Primeiramente, para que se tenha um único elemento $k \in \{1, 2, ..., 9\}$ em cada coluna, devemos ter:

$$\sum_{i=1}^{9} x_{ijk} = 1 \ , \ j, k \in \{1, 2, ..., 9\}$$

Para que se tenha um único elemento $k \in \{1, 2, ..., 9\}$ em cada linha, devemos ter:

$$\sum_{j=1}^{9} x_{ijk} = 1 \ , \ i, k \in \{1, 2, ..., 9\}$$

Para que cada um dos 9 quadrados 3×3 tenha todos os números de 1 a 9, fazemos:

$$\sum_{j=3q-2}^{3q} \sum_{i=3p-2}^{3p} x_{ijk} = 1 \ , \ k \in \{1, 2, ..., 9\} \ , \ p, q \in \{1, 2, 3\}$$

Apenas para garantir que toda a matriz do quebra-cabeça seja preenchida, temos:

$$\sum_{k=1}^{9} x_{ijk} = 1 \ , \ i, j \in \{1, 2, ..., 9\}$$

Finalmente, como no Sudoku não há intenção de minimizar ou maximizar algo, apenas encontrar a solução factível, podemos definir como objetivo minimizar $\mathbf{0}^T \boldsymbol{x}$.

Portanto, temos a seguinte modelagem do Sudoku clássico:

Minimize $\mathbf{0}^T \boldsymbol{x}$

sujeito a:

$$\sum_{i=1}^{9} x_{ijk} = 1 \ , \ j, k \in \{1, 2, ..., 9\}$$
$$\sum_{j=1}^{9} x_{ijk} = 1 \ , \ i, k \in \{1, 2, ..., 9\}$$
$$\sum_{j=3q-2}^{3q} \sum_{i=3p-2}^{3p} x_{ijk} = 1 \ , \ k \in \{1, 2, ..., 9\} \ , \ p, q \in \{1, 2, 3\}$$
$$\sum_{k=1}^{9} x_{ijk} = 1 \ , \ i, j \in \{1, 2, ..., 9\}$$
$$x_{ijk} \in \{0, 1\} \ , \ \forall i, j, k \in \{1, 2, ..., 9\}$$

3 Modelagem do Sudoku Geral

Modelaremos um Sudoku $n \times n$, com $n = m^2$ quadrado perfeito, utilizando programação linear.

Primeiramente, sejam as variáveis de decisão binárias:

$$x_{ijk} = \begin{cases} 1, \text{ se o elemento } (i, j) \text{ da matriz do Sudoku é o número inteiro } k \\ 0, \text{ caso contrário.} \end{cases}$$

Agora, desejamos que a solução satisfaça as regras do jogo.

Primeiramente, para que se tenha um único elemento $k \in \{1, 2, ..., n\}$ em cada coluna, devemos ter:

$$\sum_{i=1}^{n} x_{ijk} = 1 \ , \ j, k \in \{1, 2, ..., n\}$$

Para que se tenha um único elemento $k \in \{1, 2, ..., n\}$ em cada linha, devemos ter:

$$\sum_{j=1}^{n} x_{ijk} = 1 \ , \ i, k \in \{1, 2, ..., n\}$$

Para que cada um dos n quadrados $m \times m$ tenha todos os números de 1 a n, fazemos:

$$\sum_{j=(m-1)q+1}^{mq} \sum_{i=(m-1)p+1}^{mp} x_{ijk} = 1 \ , \ k \in \{1, 2, ..., n\} \ , \ p, q \in \{1, 2, ..., m\}$$

Apenas para garantir que toda a matriz do quebra-cabeça seja preenchida, temos:

$$\sum_{k=1}^{n} x_{ijk} = 1 \ , \ i, j \in \{1, 2, ..., n\}$$

Finalmente, como no Sudoku não há intenção de minimizar ou maximizar algo, apenas encontrar a solução factível, podemos definir como objetivo minimizar $\mathbf{0}^T \boldsymbol{x}$.

Portanto, temos a seguinte modelagem do Sudoku clássico:

Minimize $\mathbf{0}^T \boldsymbol{x}$

sujeito a:

$$\sum_{i=1}^{n} x_{ijk} = 1 \ , \ j,k \in \{1,2,...,n\}$$

$$\sum_{j=1}^{n} x_{ijk} = 1 \ , \ i,k \in \{1,2,...,n\}$$

$$\sum_{j=(m-1)q+1}^{mq} \sum_{i=(m-1)p+1}^{mp} x_{ijk} = 1 \ , \ k \in \{1,2,...,n\} \ , \ p,q \in \{1,2,...,m\}$$

$$\sum_{k=1}^{n} x_{ijk} = 1 \ , \ i,j \in \{1,2,...,n\}$$

$$x_{ijk} \in \{0,1\} \ , \ \forall i,j,k \in \{1,2,...,n\}$$

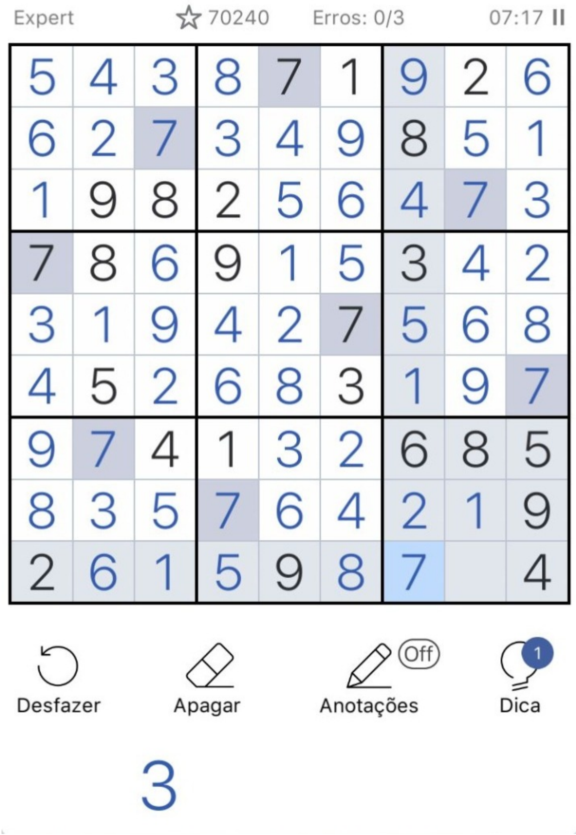
4 Implementação

Veja em detalhes a implementação das modelagens acima descritas na linguagem R, através do solver *GLPK*, de forma devidamente comentada no arquivo *Sudoku_PLI.R* anexado.

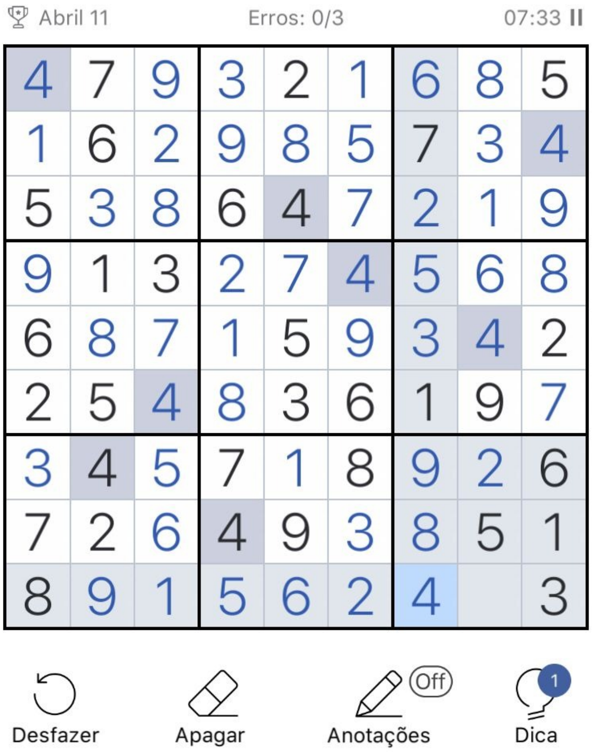
5 Resultados

Veja nas imagens a seguir, exemplos variados de sudokus resolvidos (números iniciais em preto e números preenchidos em azul para os jogos 9×9, e números iniciais em roxo e preenchidos em preto para a 16×16) e os resultados obtidos pela implementação em R:

Observação: No caso 16×16, as letras de A a G substituem os números de 10 a 16, nessa ordem.



	1	2	3	4	5	6	7	8	9
1	5	4	3	8	7	1	9	2	6
2	6	2	7	3	4	9	8	5	1
3	1	9	8	2	5	6	4	7	3
4	7	8	6	9	1	5	3	4	2
5	3	1	9	4	2	7	5	6	8
6	4	5	2	6	8	3	1	9	7
7	9	7	4	1	3	2	6	8	5
8	8	3	5	7	6	4	2	1	9
9	2	6	1	5	9	8	7	3	4



7

	1	2	3	4	5	6	7	8	9
1	4	7	9	3	2	1	6	8	5
2	1	6	2	9	8	5	7	3	4
3	5	3	8	6	4	7	2	1	9
4	9	1	3	2	7	4	5	6	8
5	6	8	7	1	5	9	3	4	2
6	2	5	4	8	3	6	1	9	7
7	3	4	5	7	1	8	9	2	6
8	7	2	6	4	9	3	8	5	1
9	8	9	1	5	6	2	4	7	3

8

	1	2	3	4	5	6	7	8	9
1	1	7	8	9	2	6	3	4	5
2	9	4	3	8	5	1	7	2	6
3	6	5	2	4	7	3	8	9	1
4	7	2	4	5	6	8	9	1	3
5	5	1	9	3	4	2	6	8	7
6	3	8	6	1	9	7	2	5	4
7	2	3	1	6	8	5	4	7	9
8	8	9	5	7	3	4	1	6	2
9	4	6	7	2	1	9	5	3	8

5	6	A	C	B	F	G	3	D	8	4	1	9	2	7	E
2	4	B	7	9	6	E	C	G	A	5	F	3	1	8	D
G	8	F	9	5	A	1	D	3	2	E	7	6	C	B	4
3	1	E	D	7	8	4	2	6	B	9	C	G	F	A	5
E	A	9	B	1	5	D	G	F	3	2	6	8	4	C	7
C	F	3	G	2	7	B	4	9	1	D	8	E	A	5	6
8	5	D	6	3	9	C	F	4	E	7	A	2	G	1	B
7	2	4	1	8	E	A	6	5	C	G	B	D	3	9	F
6	3	8	2	D	G	9	B	A	7	C	4	5	E	F	1
1	C	7	A	6	3	2	8	B	5	F	E	4	9	D	G
F	D	G	4	C	1	5	E	8	9	3	2	7	B	6	A
B	9	5	E	F	4	7	A	1	D	6	G	C	8	2	3
9	E	2	3	4	C	6	1	7	F	B	D	A	5	G	8
A	7	1	F	E	D	3	5	2	G	8	9	B	6	4	C
D	G	6	5	A	B	8	9	C	4	1	3	F	7	E	2
4	B	C	8	G	2	F	7	E	6	A	5	1	D	3	9

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	5	6	10	12	11	15	16	3	13	8	4	1	9	2	7	14
2	2	4	11	7	9	6	14	12	16	10	5	15	3	1	8	13
3	16	8	15	9	5	10	1	13	3	2	14	7	6	12	11	4
4	3	1	14	13	7	8	4	2	6	11	9	12	16	15	10	5
5	14	10	9	11	1	5	13	16	15	3	2	6	8	4	12	7
6	12	15	3	16	2	7	11	4	9	1	13	8	14	10	5	6
7	8	5	13	6	3	9	12	15	4	14	7	10	2	16	1	11
8	7	2	4	1	8	14	10	6	5	12	16	11	13	3	9	15
9	6	3	8	2	13	16	9	11	10	7	12	4	5	14	15	1
10	1	12	7	10	6	3	2	8	11	5	15	14	4	9	13	16
11	15	13	16	4	12	1	5	14	8	9	3	2	7	11	6	10
12	11	9	5	14	15	4	7	10	1	13	6	16	12	8	2	3
13	9	14	2	3	4	12	6	1	7	15	11	13	10	5	16	8
14	10	7	1	15	14	13	3	5	2	16	8	9	11	6	4	12
15	13	16	6	5	10	11	8	9	12	4	1	3	15	7	14	2
16	4	11	12	8	16	2	15	7	14	6	10	5	1	13	3	9

6 Conclusão

Foi apresentada a classe de quebra-cabeças conhecidos como Sudoku, os quais estão fortemente presentes no cotidiano. Uma abordagem matemática envolvendo programação linear nos permitiu modelar o Sudoku clássico 9×9 e qualquer Sudoku geral $n \times n$, com n quadrado perfeito.

Uma grande vantagem de se ter um algoritmo de resolução de Sudokus é tornar a sua resolução mais rápida, o que pode ser interessante para gerar aplicações. Um exemplo disso, é a criptografia: dados alguns valores iniciais e algumas regras de preenchimento, pode-se decodificar toda a matriz de dados, pois a solução é única, além de permitir uma trasferência mais ágil dos dados de forma mais enxuta.

Uma desvantagem evidente dos algoritmos baseados nas modelagens aqui descritas é a complexidade dos mesmos. Valores pequenos de n já geram uma quantidade enorme de restrições para o problema a ser resolvido. Assim, para Sudokus cada vez maiores, perde-se a eficiência do método.

Portanto, foi possível descrever de forma detalhada o problema em linguagem natural, modelar matematicamente o problema, implementar algoritmos de solução para o problema e algumas de suas variações de forma devidamente documentada, além de validar exemplos resolvidos.

7 Referências

https://en.wikipedia.org/wiki/Sudoku

https://coin-or.github.io/pulp/CaseStudies/a_sudoku_problem.html

https://www.researchgate.net/profile/Amy-Langville/publication/228615106_An_integer_programming_model_for_the_sudoku_problem/links/5c0883e7a6fdcc494fdca89d/An-integer-programming-mo
pdf