UNIVERSIDADE FEDERAL DO ESPÍRITO SANTO
TEORIA DOS GRAFOS
INF-09348 e PINF-6037
2^o Trabalho computacional – 2016-2 – Profa Claudia Boeres

O problema de coloração de grafos é um dos problemas clássicos da computação e possui muitas aplicações em problemas reais. Alocação de registradores e problemas de *scheduling*, problemas de tabela horário, resolução do jogo sudoku[1], entre outros, são alguns exemplos de aplicações.

O problema de coloração de vértices pode ser definido em como colorir (atribuir nomes para os vértices) de modo que nenhum vértice adjacente possui a mesma cor. Normalmente a coloração de grafos tem o objetivo de colorir todos os vértices minimizando a quantidade de cores. A quantidade mínima de cores necessárias para colorir um grafo G é denominada número cromático e é representado por $\chi(G)$. Achar $\chi(G)$ é um problema de otimização extremamente difícil.

Nesse trabalho abordaremos a versão de decisão do problema de coloração chamado k-coloração. Nessa versão devemos descobrir se é possível colorir o grafo usando apenas k cores, se for, a solução deve ser apresentada. A prova de que é possível colorir é um exemplo de coloração. Existem alguns casos triviais: Se o $k \geq |V|$ o problema é trivial pois é possível dar uma cor diferente para cada vértice; se $k = 2$ basta descobrir se o grafo é bipartido, que pode ser feito com uma busca em largura. Para os demais casos k-coloração é $\mathcal{NP}$-Completo.

O segundo trabalho computacional consiste na implementação de um resolvedor de Sudoku[2]. No Sudoku temos um tabuleiro de $n^2 \times n^2$ que é dividido em blocos de $n \times n$, na figura 1 podemos ver um exemplo quando $n = 3$. O objetivo é colocar um número $i \in \{1, \ldots, n^2\}$ em cada célula de modo que todos os demais números da mesma linha, da coluna, e do bloco sejam diferentes de i.

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

Figura 1: Exemplo do tabuleiro de Sudoku ($n = 3$).

[1]Sim, Sudoku é NP-Completo. [Artigo]
[2]Por que é a aplicação mais legal de k-coloração.

Modelagem como k-coloração

A modelagem do Sudoku como k-coloração é bem fácil de ser realizada.

- Considere cada célula do tabuleiro como um vértice do grafo.
- Se dois vértices estão na mesma coluna, linha, ou bloco $n \times n$ eles são adjacentes no grafo.
- Os números que já estão preenchidos inicialmente no tabuleiro devem ser a cor do vértice e essa cor não deve ser alterada (constante).

Nessa modelagem é fácil de perceber que o grafo terá $\chi(G) = n^2$ e que se uma cor $i \in \{1, \ldots, n^2\}$ ocorrer em qualquer célula, não poderá ocorrer em nenhum vértice vizinho correspondente, que representa no caso em questão, uma linha, coluna ou bloco, exatamente como a restrição do Sudoku.

Método

Como o problema de k-coloração é $\mathcal{NP}$-completo, não existe algoritmo capaz de garantir a solução exata em tempo polinomial, a menos que $\mathcal{P} = \mathcal{NP}$. Portanto, cada dupla deverá implementar uma heurística para resolver o problema n^2-coloração com $n \in \{2, 3, 4, 5\}$. Instâncias em anexo (final do documento).

Sugerimos que a heurística utilizada seja gulosa (cada vértice do grafo deve ser colorido com o menor rótulo de cor possível, isto é, que não tenha sido previamente utilizada em algum vizinho do vértice). No entanto, é possível que em algum momento não exista cor disponível para algum vértice, ou seja, todas as n^2 cores já foram usadas nos seus vizinhos. Neste caso, a heurística deve sortear alguma dentre as n^2 possibilidades para colorir o vértice em questão, gerando consequentemente uma solução inválida para o problema. O trabalho consiste em:

- pesquisar e implementar um algoritmo construtivo guloso para gerar uma solução para o problema.
- propor um mecanismo para tentar diminuir o número de violações da solução.

Execução

Como o método implementado é uma heurística e portanto não garante que a solução correta seja encontrada, cada dupla deverá executar 10 vezes para cada instância e anotar:

- Tempo médio de execução do algoritmo.
- Quantas restrições foram violadas (melhor, pior e média das 10 execuções). Para contar o número de restrições violadas deve-se checar todos os vértices, verificando quantos vizinhos possui a mesma cor que ele.
- Quantas vezes (do total de execuções para cada instância) que o método conseguiu achar a solução correta, isto é, que não viole nenhuma restrição.

Seminários

Cada grupo deve fazer uma apresentação de no máximo 15 minutos contendo uma breve explicação dos algoritmos implementados, detalhes da implementação, estruturas de dados utilizadas, apresentação e análise dos resultados obtidos, com destaque para o algoritmo construtivo utilizado e o mecanismo de redução de violações adotado.

Entrega

O material deve ser enviado para o Diego (e-mail: diego.lucchi@gmail.com, com cópia para boeres@inf.ufes.br), no primeiro dia de apresentação dos seminários (25/11/2016). O material consiste em:

- arquivo dos slides apresentados
- compactado dos códigos implementados
- arquivo compactado dos grafos de entrada gerados

Data dos seminários e entrega do material: 25/11/16 e 30/11/2016.

Anexo: Instâncias

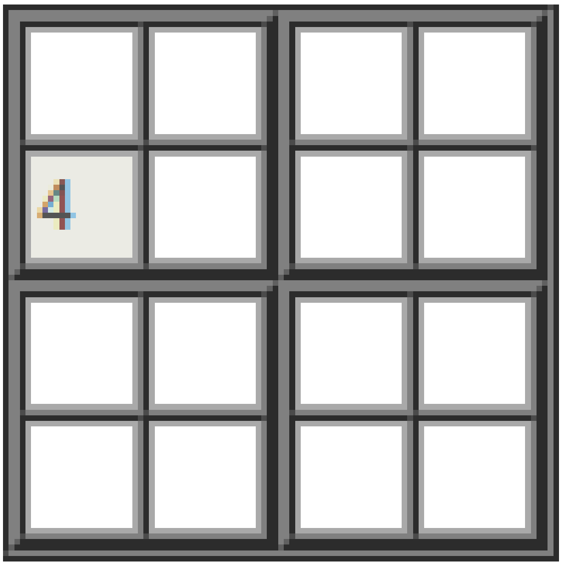

Figura 2: $n = 2$

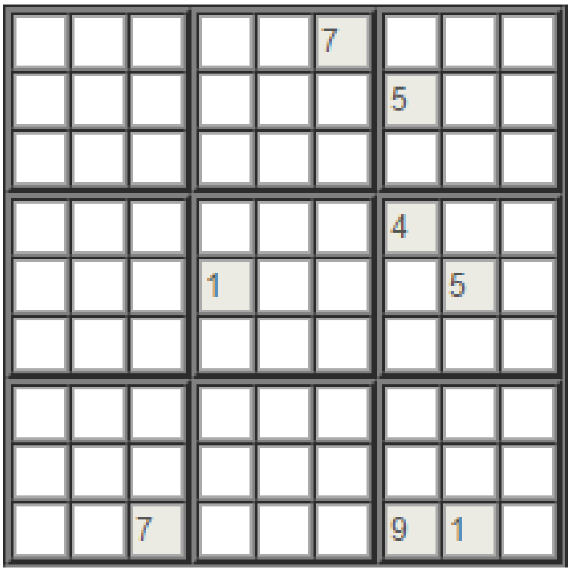

Figura 3: $n = 3$

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

Figura 4: $n = 4$

																						13		9
				20		24					3								1		18			
														22		3		7	2	25		21		
				7												17								
						15									6									
									19											6				
												24												
							4			20													8	
		15		8				12	21															
												25												
		9			19														21					
										19				10									12	
21																								
													25											
		20									7						1							18
		11				13					19											1		
																		16				2		
																	20			10			16	
24	10		22																					
																					23			
			11						9															
											14													
					25										6									

Figura 5: $n = 5$