TÉCNICAS DE COLORAÇÃO DE GRAFOS APLICADAS À RESOLUÇÃO DE QUEBRA-CABEÇAS DO TIPO SUDOKU

Article · November 2013

CITATIONS
0

READS
325

3 authors, including:

Thiago Gouveia
Instituto Federal de Educação, Ciência e Tec…
4 PUBLICATIONS **5** CITATIONS
SEE PROFILE

Paulo Ditarso Maciel Jr.
Instituto Federal de Educação, Ciência e Tec…
11 PUBLICATIONS **79** CITATIONS
SEE PROFILE

VIII CONNEPI
Congresso Norte Nordeste de Pesquisa e Inovação
Salvador - Bahia - 2013

TÉCNICAS DE COLORAÇÃO DE GRAFOS APLICADAS À RESOLUÇÃO DE QUEBRA-CABEÇAS DO TIPO SUDOKU

Nailson dos Santos Cunha[1], Thiago Gouveia da Silva[1] e Paulo Ditarso Maciel Jr[1]
[1] Unidade Acadêmica de Informática - Instituto Federal de Educação, Ciência e Tecnologia da Paraíba (IFPB) – João Pessoa, PB – Brasil. E-mails: nailsoncunha@gmail.com, {thiago.gouveia, paulo.maciel}@ifpb.edu.br;

Artigo submetido em julho/2013 e aceito em novembro/2013

RESUMO

Neste trabalho são apresentadas duas heurísticas não determinísticas desenvolvidas para rápida resolução de quebra-cabeças do tipo *Sudoku*: a **Coloração em Largura** e a **Coloração em Profundidade.** Estas heurísticas se basearam, respectivamente, nos conhecidos algoritmos de **Busca em Largura** e **Busca em Profundidade** para pesquisa em grafos. Para que conhecidas técnicas da Teoria dos Grafos pudessem ser aplicadas ao *Sudoku*, este foi tratado como um problema de coloração de grafos. As duas heurísticas foram testadas em *Sudokus* de diferentes níveis de dificuldade e se mostraram (em suas melhores configurações) viáveis para utilização na resolução deste tipo de problema, apresentando bom desempenho para resolução de *Sudokus* de nível fácil (100% de eficiência), médio (55% de eficiência) e difícil (60% de eficiência).

PALAVRAS-CHAVE: *sudoku*, coloração de grafos, busca primeiro em largura, busca primeiro em profundidade.

GRAPH COLORING TECHNIQUES APPLIED TO SOLVE SUDOKU PUZZLES

ABSTRACT

In this work, we present two non-deterministic heuristics developed for quick resolution of *Sudoku* puzzles: the **Breadth First Coloring** and the **Depth First Coloring**. Both heuristics were based, respectively, on the well known graph searching algorithms: the **Breadth First Search** and the **Depth First Search**. In order to apply these Graph Theory techniques in the context of *Sudoku* problems, we addressed these problems as a graph coloring one. The two proposed heuristics were tested in *Sudokus* with different levels of difficult and both proved to be viable, in their best configurations, for use in the resolution this kind of problems, showing up good performance for solving easy-level *Sudokus* (efficiency of 100%), medium-level *Sudokus* (efficiency of 55%) and hard-level *Sudokus* (efficiency of 60%).

KEY-WORDS: *sudoku*, graph coloring, breadth first search, depth first search.

TÉCNICAS DE COLORAÇÃO DE GRAFOS APLICADAS À RESOLUÇÃO DE QUEBRA-CABEÇAS DO TIPO SUDOKU

1. INTRODUÇÃO

Este trabalho apresenta duas heurísticas não determinísticas desenvolvidas para rápida resolução de quebra-cabeças do tipo *Sudoku*. A ideia básica consiste em transformar o tabuleiro do *Sudoku* em um problema de coloração de grafos e, desta forma, tentar colorir o grafo gerado utilizando apenas nove cores.

Como os quebra-cabeças *Sudoku* podem ter vários níveis de dificuldade, as heurísticas foram avaliadas em alguns desses níveis, conseguindo um alto percentual de resolução para as instâncias de nível mais fácil e um percentual de resolução aceitável para instâncias de nível intermediário.

O trabalho está organizado da seguinte maneira: além desta seção introdutória, a Seção 2 trata de uma rápida explicação sobre os quebra-cabeças *Sudoku*, além de alguns trabalhos relacionados ao tema; a Seção 3 revisa o problema da coloração de grafos e como este pode aplicado aos quebra-cabeças *Sudoku*; a Seção 4 trata das heurísticas desenvolvidas e dos algoritmos gerados; a Seção 5 relata os testes efetuados sobre instâncias do *Sudoku* de diferentes níveis; a Seção 6 apresenta as informações sobre os resultados obtidos; e, por último, a Seção 7 traz as considerações finais e perspectivas de trabalhos futuros.

2. SOBRE O *SUDOKU*

O *Sudoku* é um jogo de desafio lógico (ou *puzzle*) que, em sua versão mais popular, é disposto em um tabuleiro de oitenta e um quadrados, sendo nove linhas, nove colunas e nove regiões 3x3, conforme ilustrado na Figura 1a.

Uma vez que alguns espaços do tabuleiro já iniciam preenchidos com dígitos de um a nove, o objetivo do jogo é preencher cada quadrado restante com um número (também de um a nove), de forma que este mesmo número não se repita na mesma coluna, linha ou região do tabuleiro. A Figura 1b realça o primeiro quadrado do tabuleiro (amarelo), tal como a região (azul), a linha (vermelho) e a coluna (verde) as quais ele pertence.

O *Sudoku* pode ser generalizado de forma que o tabuleiro possua N^2 regiões *NxN*, tendo que ser preenchido com dígitos de *1* .. N^2. No entanto, este trabalho aborda apenas a versão tradicional do jogo, na qual *N=3*.

Vários trabalhos tem sido publicados recentemente utilizando diversas abordagens para resolução do *Sudoku*: Mantere e Koljonen (2006) propõem a utilização de Algoritmos Genéticos; Lewis (2007) utiliza *Simulated Annealing*; Bartlett et al. (2008) descreve um modelo matemático

para o *Sudoku*; e Mandal e Sadhu (2011) resolvem o problema através da metaheurística *Harmony Search*.

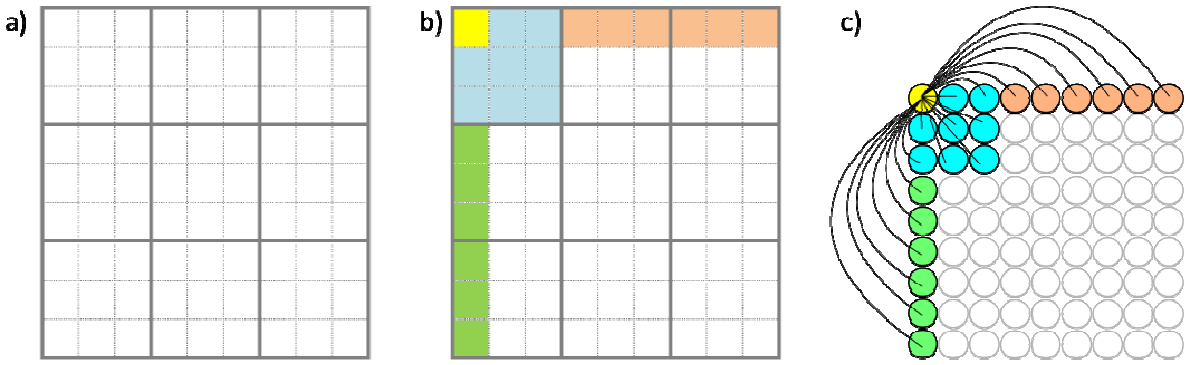

Figura 1a - Tabuleiro de *Sudoku* vazio; 1b - Região, linha e coluna adjacentes ao primeiro quadrado do tabuleiro; 1c - Representação das arestas do primeiro vértice do grafo.

3. O PROBLEMA DA COLORAÇÃO DE GRAFOS

Na área de estudo de Teoria dos Grafos, o problema da Coloração de Grafos, na sua forma mais simples, consiste em rotular com valores (denominados cores) os vértices de um grafo de tal forma que vértices adjacentes não possuam a mesma cor.

O problema surgiu quando Francis Guthrie, ao tentar colorir o mapa da Inglaterra de modo que regiões com uma fronteira comum não recebessem a mesma cor, percebeu que seriam necessárias apenas quatro cores. Então, ele estendeu essa questão para saber se isso era possível a qualquer mapa ou grafo planar (MITCHEM, 1981).

O estudo da coloração de grafos possui várias aplicações como, por exemplo, nos problemas de atribuição de frequências a antenas de rádio ou telefonia, coloração de mapas, alocação de produtos químicos em armazéns e alocação de registradores em compiladores (GALINIER, et. al., 2012).

Formalmente, o problema pode ser descrito como segue. Dado um grafo não direcionado $G = (V,E)$, com um conjunto V de vértices e um conjunto E de arestas que incidem sobre os vértices de G, o problema de coloração de grafos consiste em atribuir cores para os vértices de G, de modo que dois vértices adjacentes não possuam a mesma cor.

A representação de um tabuleiro de *Sudoku* em forma de grafo pode ser feita de maneira que cada quadrado do tabuleiro represente um vértice, e seus adjacentes sejam os quadrados que se encontram na mesma coluna, na mesma linha e no mesmo quadrante. Desta forma, o grafo resultante possui oitenta e um vértices, cada um destes possuindo vinte vértices adjacentes. A Figura 1c representa o vértice referente ao primeiro quadrado do tabuleiro, tal como seus vértices adjacentes.

Assumindo que os números de 1 a 9 do *Sudoku* representem as cores do problema de coloração de grafos, a verificação de que um determinado tabuleiro de *Sudoku* é solucionável reduz-se a

checar se o grafo gerado é 9-colorível, ou seja, se é possível colorir o grafo com o uso de apenas nove cores obedecendo-se a regra de que vértices adjacentes não podem possuir a mesma cor. Este problema de decisão é NP-Difícil (GAREY; JOHNSON, 1979), estimulando, assim, o desenvolvimento de métodos heurísticos para sua resolução.

4. ABORDAGENS INVESTIGADAS

Esta seção descreve duas heurísticas desenvolvidas para resolução rápida do *Sudoku* a partir da transformação deste no problema da coloração de grafos: **Coloração em Largura (CL)** e **Coloração em Profundidade (CP)**. Estas heurísticas são baseadas, respectivamente, em dois algoritmos de comum utilização para percorrer vértices de grafos: a Busca em Largura (*Breadth-First Search*) e a Busca em Profundidade (*Depth-First Search*).

Por conta do comportamento não determinístico das heurísticas desenvolvidas, estas foram envoltas em um framework Multi-Start (MARTÍ; MORENO-VEGA; DUARTE, 2010). Conforme detalhado na Figura 2, o procedimento Multi-Start executa a heurística selecionada (até) max_iter vezes, aumentando a chance de sucesso na resolução do sudoku.

Os tópicos seguintes desta seção descrevem o funcionamento das heurísticas Coloração em Largura e Coloração em Profundidade, assim como a rotina de atribuição de cores aos vértices, que é usada por ambas abordagens.

```
Multi-Start ( G, s, Heuristica, max_iter )
1   PARA iter = 1 ATÉ max_iter FAÇA
2       EXECUTE( Heuristica( G, s ) )
3       SE G é 9-colorível ENTÃO
4           RETORNE SUCESSO
5   RETORNE FALHA
```

Figura 2 – *Framework Multi-Start*

4.1. COLORAÇÃO EM LARGURA

A heurística de Coloração em Largura (CL) baseia-se no conhecido algoritmo de Busca em Largura para pesquisa em grafos (CORMEN et al., 2009). Na CL a coloração tem início em um vértice pré-definido, dito origem, sendo coloridos primeiramente os vértices adjacentes a este. Após esta iteração, um dos vértices adjacentes torna-se a nova origem e o processo é repetido até que todos os vértices tenham sido coloridos.

Como apresentado na Figura 3, a CL foi implementada com a utilização de uma fila auxiliar de forma a manter a sequência de coloração dos vértices do grafo. O caráter não determinístico da heurística é proveniente da rotina *Colorir_Vértice* (detalhada na Subseção 4.3), invocada sempre que se deseja atribuir uma cor a um vértice.

```
Coloração_Largura( G, s )
1   Colorir_Vértice( s )
2   ENFILEIRA( Fila, s )
3   ENQUANTO Fila ≠ { } FAÇA
4       vi ← DESENFILEIRA( Fila )
5       PARA cada vértice va ∈ adj( vi ) FAÇA
6           SE va não está colorido ENTÃO
7               Colorir_Vértice( va )
8               ENFILEIRA( Fila, va )
```

Figura 3 – Heurística de Coloração em Largura (CL)

4.2. COLORAÇÃO EM PROFUNDIDADE

A estratégia seguida pela heurística de coloração em profundidade (CP), baseada no algoritmo de Busca em Profundidade para pesquisa em grafos (CORMEN et al., 2009), consiste em iniciar a coloração a partir de uma origem no grafo e, sempre que possível, mudar a origem para um dos seus vértices adjacentes. Deste modo, quando não houver mais como aprofundar (ou seja, todos os vértices adjacentes à origem já estão coloridos), a busca retorna para o vértice anterior ao último encontrado e reinicia o processo a partir deste.

Como apresentado na Figura 4, a CP foi implementada com a utilização de uma pilha auxiliar com o intuito de manter a sequência de coloração dos vértices do grafo. Assim como na heurística CL, o caráter não determinístico da abordagem é proveniente da rotina *Colorir_Vértice.*

```
Coloração_Profundidade( G, s )
1   Colorir_Vértice( s )
2   EMPILHA( Pilha, s )
3   ENQUANTO Pilha ≠ { } FAÇA
4       vi ← DESEMPILHA( Pilha )
5       PARA cada vértice va ∈ adj( vi ) FAÇA
6           SE va não está colorido ENTÃO
7               Colorir_Vértice(va )
8               EMPILHA( Pilha, vi )
9               EMPILHA( Pilha, va )
10              QUEBRA o laço PARA
```

Figura 4 – Heurística de Coloração em Profundidade (CP)

4.3. ROTINA DE ATRIBUIÇÃO DE CORES AOS VÉRTICES

Como visto nos tópicos 4.1 e 4.2, tanto a heurística CL quanto a CP delegam a tarefa de escolha de uma cor para um determinado vértice à sub-rotina *Colorir_Vértice*. Uma abordagem intuitiva e gulosa seria atribuir a menor cor possível ao vértice em questão, na tentativa de minimizar o número total de cores utilizadas. No entanto, esta abordagem apresentou resultados muito ruins, não sendo capaz de resolver nem mesmo instâncias simples do problema.

A alternativa encontrada foi a utilização de uma abordagem entre o guloso e o aleatório. Para o vértice do grafo recebido pela função é criada a lista ordenada de cores possíveis de serem utilizadas (entre 1 e 9). Uma cor é sorteada aleatoriamente desta lista e atribuída ao vértice em questão. Se nenhuma cor entre 1 e 9 puder ser atribuída, então esta iteração não conseguiu resolver o desafio e a heurística (CP ou CL) deve ser reiniciada.

A sub-rotina *Colorir_Vértice* recebe dois parâmetros: o vértice que se deseja colorir e tamanho máximo da lista de cores possíveis que deve ser utilizada para o sorteio da cor atribuída ao vértice. Este segundo parâmetro controla o nível de gulosidade/aleatoriedade das abordagens, uma vez que uma lista restrita ao tamanho 1 remete a uma heurística puramente gulosa, enquanto uma lista de tamanho 9 produz um comportamento totalmente aleatório.

Neste trabalho foi estudado o comportamento das heurísticas CL e CP com tamanhos de lista 9, 7, 5 e 3. Como apresentado na Figura 5, dado um vértice *v* e um tamanho máximo da lista de candidatos *max_list*, inicialmente, o procedimento *Colorir_Vértice* constrói uma lista (dita lista restrita) com as cores que podem ser atribuídas a *v.* E, por fim, colore *v* com uma cor escolhida aleatoriamente dentre as *max_list* melhores cores da lista restrita.

```
Colorir_Vértice( v, max_list )
1   Cores = { }
2   PARA cor = 1 até 9 FAÇA
3       SE pode_usar_cor(v, cor) ENTÃO
4           Cores ← Cores + { cor }
5       SE |Cores| = max_list ENTÃO
6           quebre o laço PARA
7   v ← escolha_cor_aleatória( Cores )
```

Figura 5 – Rotina de Coloração de Vértices

5. EXPERIMENTOS REALIZADOS

Com objetivo de mensurar a qualidade das abordagens desenvolvidas, foram selecionadas oito instâncias do quebra-cabeça *Sudoku* a partir de sites de jogos da Internet, classificadas quanto ao nível de dificuldade. A Figura 6 exibe as oito instâncias utilizadas como *benchmark*: nível fácil: F0 e F1; nível médio: M0 e M1; nível difícil: D0 e D1; e nível experiente: E0 e E1.

Para melhor avaliação das heurísticas, foram estudados diferentes parâmetros relacionados ao vértice de origem e ao tamanho máximo da lista restrita utilizada pelo procedimento *Colorir_Vértice*.

Para todos os testes, foi utilizado o valor 10.000 (dez mil) para o parâmetro *max_iter (*passado para o *framework Multi-Start).* Em cada execução, caso a heurística encontrasse uma solução para o *Sudoku* testado, retornava sucesso e parava imediatamente. Caso contrário, o retorno seria falha. Este processo foi repetido dez vezes para cada uma das oito instâncias utilizadas como *benchmark*.

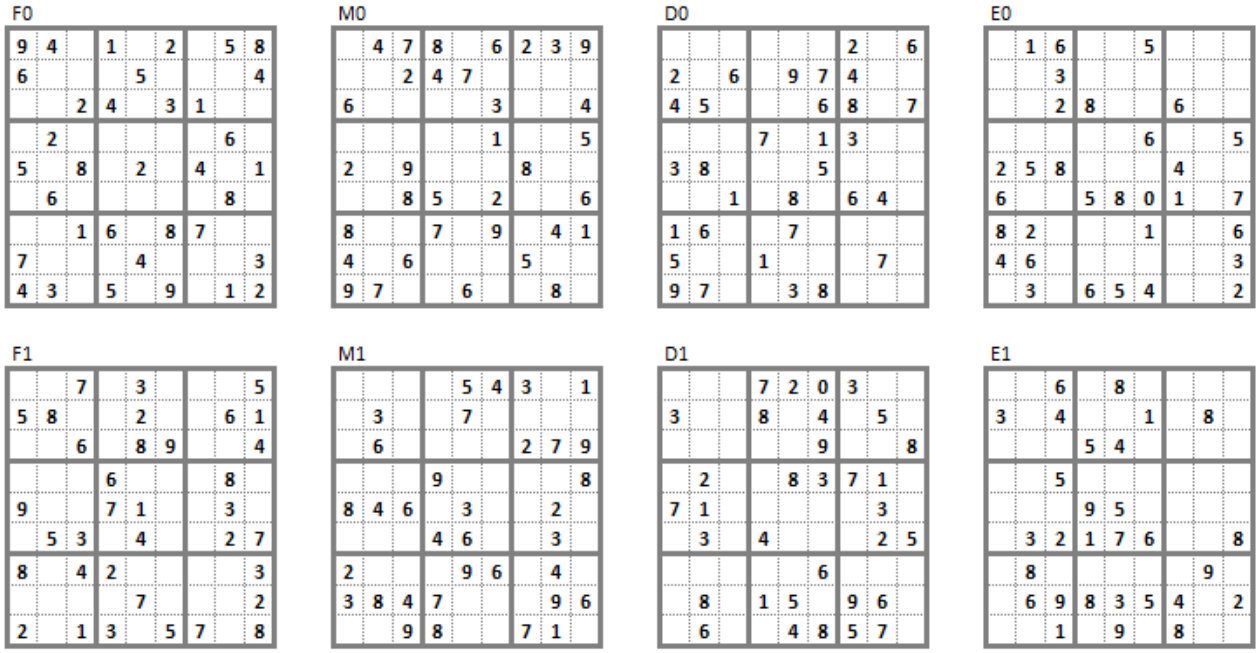

Figura 6 – *Sudokus* fáceis (F0 e F1), médios (M0 e M1), difíceis (D0 e D1) e experientes (E0 e E1)

A implementação se deu em linguagem Python 2.7 e os testes foram realizados em uma máquina virtual criada no programa Oracle Virtual Box versão 4.2 executando o Linux Mint 13. Como configuração, a máquina virtual compartilhava um núcleo lógico do processador do computador (Intel Core I5 3.33 GHZ) e 1,5GB de memória RAM.

6. RESULTADOS OBTIDOS

Esta seção apresenta as tabelas que detalham os resultados obtidos pelas heurísticas desenvolvidas, assim como uma breve discussão acerca da qualidade das técnicas propostas e de seu tempo de execução. Os resultados estão organizados de acordo com a abordagem de inicialização adotada (colunas) e também em relação ao tamanho máximo da lista de cores utilizada (linhas).

As colunas CL-1(M), CL-C(M), CL-A(M) e CL-MV(M) significam a heurística de Coloração em Largura tendo M como tamanho máximo da lista de cores, partindo, respectivamente, do(e): primeiro quadrado do tabuleiro; centro do tabuleiro; um vértice sorteado aleatoriamente; e vértice que possui maior número de adjacentes já coloridos. O mesmo se aplica às colunas CP-1(M), CP-C(M), CP-A(M) e CP-MV(M), que se referem à heurística de Coloração em Profundidade.

Para cada conjunto *{ heurística, instância, ponto de partida, tamanho máximo da lista de cores }* é apresentada a porcentagem de vezes em que o teste retornou sucesso, tendo assim conseguido resolver o quebra-cabeças *Sudoku* em questão, além de linhas e colunas com médias que sumarizam estes percentuais.

Tabela 1 – Resultados obtidos para *Sudokus* de nível fácil.

C_List	CL-1	CL-C	CL-A	CL-MV	Méd.	CP-1	CP-C	CP-A	CP-MV	Méd.
9	30%	70%	**80%**	**80%**	65%	**100%**	**100%**	**100%**	85%	96%
7	60%	60%	70%	**80%**	68%	95%	95%	90%	90%	93%
5	30%	50%	75%	**80%**	59%	**100%**	95%	90%	**100%**	96%
3	10%	25%	60%	60%	39%	**100%**	90%	95%	90%	94%
Méd.	33%	51%	71%	75%	**58%**	99%	95%	94%	91%	**95%**

Tabela 2 – Resultados obtidos para *Sudokus* de nível médio.

C_List	CL-1	CL-C	CL-A	CL-MV	Méd.	CP-1	CP-C	CP-A	CP-MV	Méd.
9	40%	10%	30%	25%	26%	15%	15%	15%	5%	13%
7	30%	45%	**55%**	35%	41%	5%	15%	5%	20%	11%
5	30%	30%	45%	20%	31%	10%	5%	15%	10%	10%
3	25%	20%	45%	30%	30%	**25%**	10%	10%	15%	15%
Méd.	31%	26%	44%	28%	**32%**	14%	11%	11%	13%	**12%**

Tabela 3 – Resultados obtidos para *Sudokus* de nível difícil.

C_List	CL-1	CL-C	CL-A	CL-MV	Méd.	CP-1	CP-C	CP-A	CP-MV	Méd.
9	35%	35%	40%	30%	35%	15%	5%	0%	10%	8%
7	30%	50%	40%	35%	39%	15%	15%	20%	10%	15%
5	30%	35%	45%	**60%**	43%	**30%**	10%	10%	20%	18%
3	25%	25%	35%	40%	31%	10%	20%	10%	0%	10%
Méd.	30%	36%	40%	41%	**37%**	18%	13%	10%	10%	**13%**

Tabela 4 – Resultados obtidos para *Sudokus* de nível experiente.

C_List	CL-1	CL-C	CL-A	CL-MV	Méd.	CP-1	CP-C	CP-A	CP-MV	Méd.
9	5%	0%	0%	5%	3%	0%	**5%**	0%	**5%**	3%
7	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
5	0%	5%	**10%**	0%	4%	0%	0%	0%	0%	0%
3	0%	0%	5%	0%	1%	0%	0%	0%	0%	0%
Méd.	1%	1%	4%	1%	**2%**	0%	1%	0%	1%	**1%**

Como apresentado na Tabela 1, para as instâncias de nível fácil (F0 e F1), a heurística de Coloração em Profundidade se mostrou mais eficiente que o de Coloração em Largura, conseguindo encontrar solução em 100% das tentativas para CP-1(9), CP-C(9), CP-A(9), CP-1(5), CP-MV(5) e CP-1(3). Por sua vez, a Coloração em Largura conseguiu um percentual máximo de 80% com CL-A(9) , CL-MV(9), CL-MV(7) e CL-MV(5). O tempo máximo gasto em um teste, no pior caso (de efetuar as dez mil repetições e não encontrar uma solução) foi 2,2 segundos.

Como pode ser observado na Tabela 2, para as instâncias de nível médio (M0 e M1), a heurística de Coloração em Largura supera a de Coloração em Profundidade, alcançando um valor de 55% de encontro de soluções com CL-A (7). A heurística CP chegou a 25% com CP-1(3). O tempo máximo para instâncias de nível médio, no pior caso, foi 2,9 segundos.

Para as instâncias de nível difícil (D0 e D1) a heurística CL alcançou seu melhor rendimento com CL-MV(5), obtendo soluções em 60% das tentativas. A heurística CP alcançou 30% com CP-1(5), conforme mostrado na Tabela 3. Vale notar que os melhores valores e as médias para as instâncias de nível difícil ficaram bem próximos dos valores para as instâncias de nível médio, sendo até um pouco maiores. Deste modo, pode-se afirmar que para as heurísticas apresentadas não existe uma diferença grande entre estes níveis de dificuldade. Quanto ao tempo máximo para um teste, no pior caso, foi de 3 segundos para o CP e de 2,9 segundos para o CL.

Para as instâncias de nível experiente (E0 e E1), apesar de ter obtido um baixo percentual de solução, mais uma vez a heurística CL se saiu melhor que a CP. A CL chegou a 10% de encontro de soluções com CL-A(5), enquanto a CP conseguiu um máximo de 5% nos casos CP-C(9) e CP-MV(9), como exibido na Tabela 4. Para estas instâncias, o tempo máximo dos testes foi de 4,4 segundos.

7. CONSIDERAÇÕES FINAIS E TRABALHOS FUTUROS

Neste trabalho são apresentadas duas heurísticas não determinísticas desenvolvidas para rápida resolução de quebra-cabeças do tipo *Sudoku:* a **Coloração e Largura** e a **Coloração em Profundidade**. As duas heurísticas foram testadas em *Sudokus* de diferentes níveis de dificuldade e se mostraram viáveis para serem utilizadas na resolução desse tipo de problema, com um bom desempenho na resolução de *Sudokus* de nível fácil (100% de eficiência), médio (55% de eficiência) e difícil (60% de eficiência).

Para trabalhos futuros, as duas heurísticas podem ser usadas como métodos construtivos para metaheurísticas, principalmente devido ao desempenho apresentado quanto à velocidade de execução e à possibilidade de geração de soluções que, por conta do caráter não determinístico, exploram diversas partes do universo de soluções.

REFERÊNCIAS

1. BARLETT, A., CHARTIER, T. P., LANGVILLE, A. N., RANKIN, T. D. An Integer Programming Model for the Sudoku Problem. The Journal of Online Mathematics and Its Applications: Volume 8, 2008.
2. CORMEN, T.H., LEISERSON, C.E., RIVEST, R.L., STEIN, C. Introduction to algorithms, 3rd ed., MIT Press & McGraw-Hill, 2009.
3. GALINIER, P., HAMIEZ, J., HAO, J., PORUMBEL, D., Recent advances in graph vertex coloring, In I. ZELINKA, V. SNASEL & A. ABRAHAM (eds.), Handbook of Optimization, Springer (Intelligent Systems Series, Vol. 38), pp. 505-528. 2012.
4. GAREY, M.R. JOHNSON, D.S, Computers and Intractability: A Guide to the Theory of NP-Completeness. DIMACS Series in Discrete Mathematics and Theoretical Computer Science. Freedman, New York, 1979.
5. LEWIS, R. Metaheuristics can Solve Sudoku Puzzles. Journal of Heuristics, Volume 13, 2007
6. MANDAL, S. N., SADHU, A. An Efficient Approach to Solve Sudoku Problem by Harmony Search Algorithm. Journal Anu Books, 2011.

7. MANTERE, T., KOLJONEN, J. Solving and Rating Sudoku Puzzles with Genetic Algorithms. New Developments in Artificial Intelligence and the Semantic Web, Proceedings of the 12th Finnish Artificial Intelligence Conference, 2006.

8. MARTÍ, R., MORENO-VEGA, J. M., DUARTE, A. Advanced Multi-start Methods. Handbook of Metaheuristics. International Series in Operations Research & Management Science Volume 146, 2010.

9. MITCHEM, J., On the history and solution of the four color map problem, The College Math. Journal, 12, 1981.