Propostas Eficientes para o Problema de Inundação em Grafos

Bruno José da Silva Barros
Unidade Acadêmica de Garanhuns – Universidade Federal Rural de Pernambuco
Garanhuns – PE
brrsbruno@gmail.com

Rian Gabriel S. Pinheiro
Unidade Acadêmica de Garanhuns – Universidade Federal Rural de Pernambuco
Garanhuns – PE
rian.gabriel@gmail.com

André Renato Villela da Silva
Instituto de Ciência e Tecnologia – Universidade Federal Fluminense
Rio das Ostras – RJ
andre.renato@gmail.com

RESUMO

O Problema de Inundação em Grafos deriva de um popular jogo de tabuleiro com dimensão $n \times m$ de casas coloridas. O objetivo do jogo é aplicar mudanças de cores, denominadas inundações, a alguma casa até que todo o tabuleiro seja monocromático. O jogo tem duas versões: a versão fixa e a livre, onde ambas têm o mesmo objetivo. Para resolver o problema são propostos neste trabalho três algoritmos heurísticos: *Iterated Local Search* (ILS), *Greedy Randomized Adaptive Search Procedure* (GRASP) e um Algoritmo Evolutivo (EA). Os resultados obtidos foram comparados com formulações matemáticas encontradas na literatura e obtiveram a mesma qualidade das soluções dos algoritmos exatos. Porém, foram encontradas em tempo computacional muito inferior. O aluno (primeiro autor) desenvolveu o trabalho, como parte de seu projeto de iniciação científica e trabalho de conclusão de curso em Ciência da Computação, sempre supervisionado pelos outros dois autores (orientadores).
PALAVRAS CHAVE: Problema de Inundação em Grafos. Metaheurísticas. ILS. GRASP. Algoritmo Evolutivo.
Área Principal: OC - Otimização Combinatória.

ABSTRACT

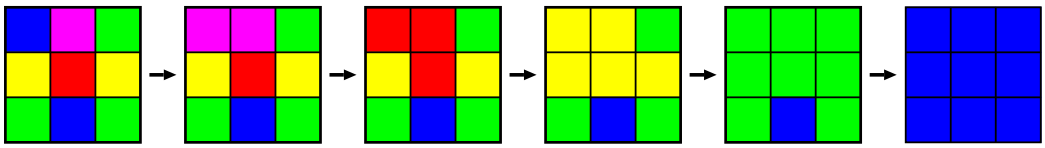
The Graph Flooding Problem is derived from a popular board game with dimension n x m of colorful cells. The goal of the game is to apply color changes, called floods, to some cells until the entire board is monochromatic. The game has two versions: the fixed version and the free version. However, the goal of the game is the same in both versions. In order to solve the problem, three heuristic algorithms are proposed: Iterated Local Search (ILS), Greedy Randomized Adaptive Search Procedures (GRASP), and one Evolutionary Algorithm. The results obtained were compared with mathematical formulations found in the literature and obtained the same quality of the solutions of the exact algorithms. However, they were found in much lower computational time. The student (first author) developed the work as part of his scientific initiation project and undergraduate thesis in Computer Science, always supervised by the other two authors (advisors).
KEYWORDS: Flooding Problem. Metaheuristics. ILS. GRASP. Evolutionary Algorithms.
Main area: OC - Combinatorial Optimization.

https://proceedings.science/p/85560?lang=pt-br

1. O problema de Inundação em Grafos

Este trabalho trata do problema de INUNDAÇÃO EM GRAFOS. Este problema é derivado de um jogo de tabuleiro com casas coloridas, cujo o objetivo é tornar o tabuleiro monocromático mediante inundações. Dada uma casa escolhida para sofrer uma mudança de cor, uma inundação é definida como uma mudança de cor com o intuito de aumentar a região inundada. Uma região inundada é uma região que já sofreu uma mudança de cor, se nenhuma alteração de cor tiver sido efetuada, então, a região inundada será somente a casa que sofrerá a mudança de cor (pivô). Em outras palavras, o objetivo do jogo é fazer com que todo o tabuleiro fique de uma única cor aplicando mudanças de cores a uma casa do tabuleiro denominada pivô. Para descrever mais detalhadamente como funciona o jogo de inundação em tabuleiros, na Figura 1 a seguir é mostrada uma sequência de movimentos que inunda um tabuleiro simples de tamanho 3×3. No exemplo em questão, o pivô é a casa na posição $(1, 1)$ (primeira linha e primeira coluna).

Figura 1: Sequência de movimento que inunda um tabuleiro 3×3

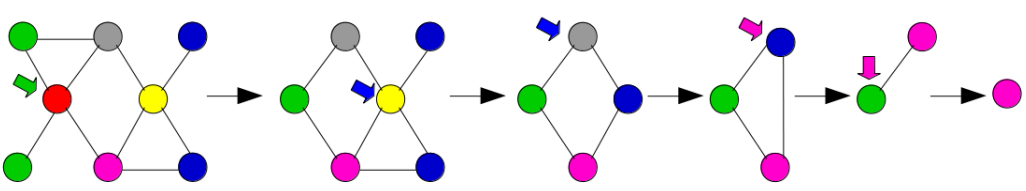


Note que, ao jogar uma cor no pivô, também mudam de cor as casas conectadas a ele. Além disso, vale ressaltar que um movimento de inundação só faz sentido se inundar pelo menos uma casa a mais do que as que já estão inundadas, por exemplo, na configuração inicial do exemplo, Figura 1, não faz sentido jogar a cor vermelha no pivô, pois a área inundada continuaria sendo a mesma. Além disso, neste exemplo, foram feitas 5 inundações para completar todo o tabuleiro.

O problema de INUNDAÇÃO EM GRAFOS é $\mathcal{NP}$-difícil e é conhecido em duas versões: INUNDAÇÃO FIXA e INUNDAÇÃO LIVRE. Esta ultima pode ser entendida como uma generalização da primeira. Na INUNDAÇÃO FIXA, em todas as rodadas do jogo, o pivô já é predefinido e não muda de uma jogada para outra. Geralmente, o pivô é a casa da posição $(1, 1)$ do tabuleiro. Na versão livre, em cada jogada o jogador define o pivô. A INUNDAÇÃO mostrada na Figura 1 é um exemplo de INUNDAÇÃO FIXA. Nesse caso, a região que começa sendo inundada sempre será aumentada a cada movimento. Já na inundação livre, como o pivô pode variar de um movimento para outro, é possível que antes da inundação ser completada existam vários componentes monocromáticos que foram inundados mas que não estão conectados entre si.

O problema de INUNDAÇÃO do ponto de vista computacional é modelado como um problema de grafos com vértices coloridos. Inicialmente, pode-se pensar que o grafo do jogo sempre será uma grade, no entanto, isso nem sempre é verdade. O grafo será uma grade se nenhum par de casas com um lado em comum tiver a mesma cor. Isso porque se duas casas com um lado em comum têm a mesma cor, elas se comportam como um único vértice. Isso implica que uma área do tabuleiro que já foi inundada também se comporta como um único vértice. A Figura 2 a seguir apresentada, mostra a modelagem de um tabuleiro em forma de grafo, além de uma sequência de inundações que contemplam todo o grafo.

Figura 2: Exemplo de Inundação Livre



https://proceedings.science/p/85560?lang=pt-br

No conhecimento dos autores, existem pouco trabalhos sobre o problema, e a maioria deles é de caráter teórico. Por exemplo, Meeks e Scott (2012) mostraram que INUNDAÇÃO é um problema $\mathcal{NP}$-difícil em grades $3 \times n$ com pelo menos quatro cores. Para uma caso particular, Clifford et al. (2011) descrevam um algoritmo de tempo polinomial para INUNDAÇÃO em grades $2 \times n$. Já Lagoutte et al. (2014) mostraram que INUNDAÇÃO é $\mathcal{NP}$-difícil em árvores. Nos trabalhos de Souza et al. (2013) e Fellows et al. (2015), foi mostrado que INUNDAÇÃO em árvores é análogo a um importante sub-caso do problema SUPER-SEQUÊNCIA COMUM MAIS CURTA.

O primeiro trabalho algorítmico sobre INUNDAÇÃO que foi encontrado com esse tema foi de Barros et al. (2015), onde foram propostos alguns algoritmos heurísticos construtivos e uma formulação matemática para resolver o problema de INUNDAÇÃO FIXA EM GRAFOS.

Considerando aplicações praticas para o problema, podemos citar algumas nas quais os jogos de inundação modelam muito bem, como por exemplo, já foi mostrado por Aschwanden (2004) que alguns modelos de propagação de doenças se espalham de forma similar a jogos de inundação. Alguns problemas de bioinformática, como mostrado por Fellows et al. (2015), podem ser modelados por inundação em grafos do tipo árvore, como é o caso de comparação de sequências de DNA. Além disso, jogos de inundação podem ser aplicados no escalonamento da linha de produção de processos de manufatura (Fellows et al., 2015).

2. Propostas

Como este trabalho aborda as duas versões do problema de INUNDAÇÃO EM GRAFOS, então cada versão foi tratada de forma independente, sendo assim, iremos explicar o funcionamento dos algoritmos e tentar explicitar as diferenças entre as duas versões.

2.1. Estrutura de Vizinhança e Busca Local

Dada uma solução S, a vizinhança $N(S)$ é o conjunto de soluções nas quais um passo da inundação foi retirado de S como definido na Equação 1.

$$N(S) = \{S' : S' = S - \{S_k\},\ 1 \leq k \leq |S|\}. \tag{1}$$

Por outro lado, a busca local consiste em varrer todo o vetor de solução tentando retirar cada elemento; se após a retirada do elemento S_k o vetor restante ainda for uma solução, então esta é adotada e o processo é repetido no índice seguinte.

2.2. Função de Perturbação

A função de pertubação consiste em selecionar um número inteiro r aleatório de 1 a k, onde k é o número de cores do grafo, e inserir essa quantidade de movimento na solução já pronta. Para a versão livre do problema, a única diferença nessa perturbação é que tem que ser selecionado um vértice pivô para cada movimento inserido. Esses movimentos extras podem ser inseridos em qualquer posição escolhidas aleatoriamente. Vale lembrar que qualquer operação sobre uma solução tem a diferença de tratar os pivôs caso seja a respeito de uma inundação livre.

2.3. GRASP

A meta-heurística GRASP consiste em criar uma solução gulosa aleatória e aplicar a busca local; se a solução resultante for melhor do que a solução salva, então esta passa a ser a nova solução. O GRASP gera uma solução a cada iteração do algoritmo, depois de melhora-la até que não seja mais possível, ele cria uma nova. Cada solução é criada de forma gulosa aleatorizada. O funcionamento do GRASP segue o *framework* descrito por Feo e Resende (1995).

Na versão livre do problema, o GRASP envolve o mesmo principio de funcionamento, porém, como tem que ser feita a escolha dos vértices pivôs, na fase de inicialização da solução, os vértices com a menor excentricidade tem mais chances de serem escolhidos como pivô do que vértices periféricos.

https://proceedings.science/p/85560?lang=pt-br

2.3.1. ILS

O ILS utiliza a mesma busca local do GRASP, porém a solução inicial é gerada com a heurística `flooding-I`, proposta por Barros et al. (2015). Feito isso, o algoritmo entra em *loop* e trabalha apenas com essa solução. Esse comportamento do ILS segue o padrão descrito por Lourenço et al. (2010).

Na versão livre, o vértice central do grafo c, é escolhido para ser o pivô inicial, depois disso a inicialização é feita com a inundação fixa tendo o vértice c como pivô. Observe que durante a faze de inicialização o pivô não varia, isso só vai acontecer na faze de perturbação.

2.3.2. Algoritmo Evolutivo (EA)

O EA proposto neste trabalho é uma modificação do EA proposto por Silva (2016). No trabalho citado, o autor não considerou utilizar uma busca local para melhorar os indivíduos. Fazendo isso, tem-se um novo algoritmo que seleciona de forma mais eficiente os indivíduos, podendo convergir mais rapidamente para a melhor solução.

Mais uma vez, na versão livre, o EA foi apenas adaptado para tratar os pivôs, e sua lógica de funcionamento continuou a mesma.

2.4. Inundação Livre

Na versão livre a maioria dos algoritmos segue a mesma lógica e foram desenvolvidos seguindo os mesmos princípios de funcionamento. A principal diferença entre os algoritmos de cada versão do problema é que, na versão livre todos eles tem que tratar os pivôs de cada rodada e na versão fixa isso não era necessário.

3. Ferramentas e Experimentos Computacionais

Esta seção informa a respeito das ferramentas utilizadas nos experimentos, e também como foram geradas as instâncias utilizadas e em seguida os experimentos realizados.

3.1. Ferramentas

Com relação às ferramentas utilizadas nos experimentos, os algoritmos foram executados em um computador com processador Intel I7-3630QM, 8GB de memória RAM, sistema operacional Windows 8.0. Os códigos foram escritos nas linguagens C e C++ e compilados com o GNU GCC 4.7.1. A formulação matemática foi executada pelo *solver* CPLEX 12.6.1.

3.2. Geração de Instâncias

Para realizar os experimentos foram geradas instâncias aleatórias de duas formas diferentes: instâncias em `grades` e instâncias `genéricas`. As instâncias do tipo `grade` simulam um tabuleiro, o que faz de todas elas instâncias planares. Isto pode ser um problema pois instâncias não planares podem ser ainda mais difíceis de serem resolvidas.

As instâncias do tipo `grade` são instâncias aleatórias, onde cada cor tem a mesma chance de figurar em uma casa do tabuleiro. Essas instâncias possuem dimensão 12×12, 15×15 e 20×20, com 6, 7 e 8 cores.

As instâncias `genérias` foram geradas da seguinte forma: primeiro consideramos que as colorações são partições, já que dois vértices da mesma cor não possuem uma aresta em comum. Tentamos centrar a probabilidade de uma aresta existir com a mesma probabilidade de cada vértice ter um vizinho de cada cor. Então, seja c o número de cores (partições), n o número total de vértices e k a quantidade de vértices de cada cor, a probabilidade será de $p' = 1/k$.

Este valor não garante que os grafos gerados sejam difíceis de serem resolvidos. Para testar isso foi usado um fator f para multiplicar essa probabilidade, podendo aumentá-la ou diminuir. Dessa forma, a probabilidade final será $p = fp' = f/k$. Neste trabalho, as instâncias genéricas geradas para os testes contêm entre $6, 7, 8, 10$ e 15 cores, com $20, 30$ e 40 vértices de cada cor e o fator f foi testado com os valores $2/3, 1$ e 2.

https://proceedings.science/p/85560?lang=pt-br

3.3. Resultados das Instâncias Planares

Para fazer a comparação entre os métodos desenvolvidos e também os métodos da literatura, foram utilizadas as instâncias de Silva (2016) e a formulação matemática (FM) também de Silva (2016).

A FM foi executada com limite de tempo igual a quatro horas (14400 segundos) e as meta-heurísticas foram executadas 30 vezes com limite de tempo igual a $|V|/10$, onde $|V|$ é a quantidade de vértices de cada grafo.

A Tabela 1 exibe o resultado das execuções nas instâncias planares. Na última linha da tabela a seguir, nomeada de "Resumo", as colunas referentes a "Melhor Solução", mostram a quantidade de instâncias que cada algoritmo não chegou na melhor solução encontrada por qualquer um dos métodos considerados. As instâncias marcadas com "*", indicam em quais instâncias a FM não encontrou o ótimo. Nas colunas em "Frequência" o resumo contém a quantidade de vezes que a melhor solução foi encontrada, considerando as 30 execuções de cada instância e sem considerar a melhor solução de um método no qual, esse encontrou uma solução melhor. Por último, o resumo de "Média das Soluções", apresenta a média das médias.

Tabela 1: Resultado das Instâncias Planares - Inundação Fixa

Instância				Melhor Solução				Frequência			Média das Soluções		
n	C	#	$\|V\|$	FM	EA	ILS	GR	EA	ILS	GR	EA	ILS	GR
		1	99	**18**	**18**	**18**	**18**	29	30	30	18.0	18.0	18.0
		2	105	**20**	**20**	**20**	**20**	16	24	26	20.5	20.2	20.1
12	6	3	97	**18**	**18**	**18**	**18**	30	30	30	18.0	18.0	18.0
		4	101	**18**	**18**	**18**	**18**	30	30	30	18.0	18.0	18.0
		5	92	**16**	**16**	**16**	**16**	30	30	30	16.0	16.0	16.0
		1	146	**19**	**19**	**19**	**19**	29	30	30	19.0	19.0	19.0
		2	167	**24**	**24**	**24**	**24**	22	29	29	24.3	24.0	24.0
15	6	3	142	**20**	**20**	**20**	**20**	28	30	24	20.1	20.0	20.2
		4	146	**19**	**19**	**19**	**19**	27	29	26	19.1	19.0	19.1
		5	164	**23**	**23**	**23**	**23**	9	29	3	23.8	23.0	24.4
		1	266	**25**	**25**	**25**	**25**	19	13	13	25.4	25.7	25.6
		2	251	**22**	**22**	**22**	**22**	23	15	23	22.3	22.5	22.2
20	6	3	266	**27**	**27**	28	28	1	8	9	28.3	28.7	28.7
		4	269	**25**	**25**	**25**	**25**	29	26	14	25.0	25.2	25.6
		5	253	**27**	**27**	28	28	21	10	7	27.4	28.7	28.8
		1	251	**25**	**25**	**25**	**25**	6	2	2	25.8	26.0	25.9
20	7	2	243	**24**	**24**	**24**	25	8	5	1	25.1	25.7	26.4
		3	265	**25**	**25**	**25**	**25**	10	30	13	25.7	25.0	25.6
		1	267	**27**	**27**	**27**	**27**	12	24	5	27.7	27.2	28.1
20	8	2*	258	24	**23**	**23**	24	2	1	12	23.9	24.0	24.6
		3	298	**28**	**28**	**28**	**28**	5	16	8	29.5	28.5	28.9
Resumo				1	0	2	4	386	423	336	22.99	22.97	23.2

Os resultados da Tabela 1 mostraram que a formulação matemática encontrou a maioria dos ótimos mas executou por muito mais tempo, e o EA e o ILS apresentaram resultados muito próximos, mas o EA aparenta ser um pouco melhor que o ILS se consideramos as médias e a frequência das melhores soluções. O GRASP não ficou muito atrás mas mesmo assim obteve um desempenho um pouco inferior.

3.4. Resultados das Instâncias Genéricas

Na Tabela 2 é apresentado a execução das instâncias genéricas com $f = 2/3$. Os resultados são similares para os parâmetros: $f = 1$ e $f = 2$. A variação do f acaba construindo grafos mais ou menos densos, e isso influencia na dificuldade de resolver as instâncias. A tabelas a seguir mostra ainda os limites duais (LD) e primais (LP) obtidos pela FM, na coluna "Limites". A tabela também mostra o tempo de execução do exato.

https://proceedings.science/p/85560?lang=pt-br

Tabela 2: Resultados das Instâncias Genéricas com $f = 2/3$

Instância		Melhor Solução			Frequência			Média das Soluções			Limites		
$\|C\|$	k	EA	ILS	GR	EA	ILS	GR	EA	ILS	GR	LP	LD	Tempo(s)
	20	13	13	13	30	30	30	13.0	13.0	13.0	13	13	68.1
6	30	14	14	14	17	12	13	14.4	14.6	14.6	14	14	468.0
	40	15	15	15	30	28	30	15.0	15.1	15.0	15	15	873.4
	20	13	13	13	21	30	26	13.3	13.0	13.1	13	13	261.9
7	30	14	14	14	5	9	4	14.8	14.7	14.9	14	14	994.7
	40*	16	16	16	27	22	11	16.1	16.3	16.6	16	14	14400.0
	20	16	16	17	10	9	30	16.7	16.7	17.0	16	16	1602.9
8	30	16	16	16	14	5	1	16.5	16.9	17.0	16	16	13481.3
	40*	17	17	17	29	14	16	17.0	17.5	17.5	17	14	14400.0
	20*	17	17	17	27	24	5	17.1	17.2	17.8	17	15	14400.0
10	30*	18	18	18	29	20	12	18.0	18.3	18.6	18	15	14400.0
	40*	18	19	18	3	2	1	18.9	19.9	19.9	21	15	14400.0
	20*	21	21	21	28	4	3	21.1	21.8	21.9	39	18	14400.0
15	30*	22	23	23	10	9	4	22.7	23.7	23.9	24	19	14400.0
	40*	22	23	23	3	6	1	22.9	23.9	24.3	29	17	14400.0
Resumo		0	3	3	283	207	152	17.16	17.51	17.67		8	

A Tabela 2 também segue o mesmo padrão da Tabela 1, apenas um detalhe a mais foi adicionado, na coluna "Limites", o resumo indica a quantidade de instâncias que a FM não encontrou o ótimo. Como é possível observar, não foi possível encontrar muitos ótimos com a FM por causa da ordem dos grafos. Nem mesmo o tempo de 4 horas de execução foi suficiente para encontrar soluções próximas das meta-heurísticas em muitas instâncias. Foi possível constatar também que a medida que f cresce (resultado não exposto neste trabalho devido ao número de páginas), as instâncias ficaram mais fáceis de serem resolvidas, enquanto para o f menor as instâncias mostraram-se mais difíceis. Isso pode ser atestado verificando a quantidade de ótimos que não foi encontrado para cada valo de f, quando este cresce, a quantidade de ótimos não encontrados diminui, ou seja, grafos densos são mais fáceis de serem resolvidos.

3.5. Resultados da Inundação Livre

As instâncias utilizadas para realizar os testes forma extraídas de Silva et al. (2017) e são de pequeno porte pois do ponto de vista prático, essa versão do problema é bem mais difícil de resolver. Estas instâncias são do tipo grade, e não foram feitos testes em instâncias genéricas para o problema de inundação livre. Além disso, os algoritmos exatos tem um comportamento ainda pior, dessa forma, não faz sentido usar instâncias de grande porte sem ter algum resultado dos algoritmos exatos para comparar.

Para fazer a comparação dos métodos propostos neste trabalho para a versão livre de inundação, foi utilizado o melhor dos três algoritmos *Backtracking* propostos em da Silva et al. (2017) e executados com tempo limite de duas horas (7200 segundos). Os demais algoritmos foram executados seguindo as mesmas configurações dos algoritmos voltados para a inundação fixa.

A Tabela 3 a seguir, mostra o resultado das execuções dos algoritmos de inundação livre. Sua configuração segue o mesmo padrão utilizado para as instâncias de inundação fixa.

https://proceedings.science/p/85560?lang=pt-br

Tabela 3: Resultado da INUNDAÇÃO LIVRE

Instância				Melhor Solução				Frequência			Média		
N	C	#	$\|V\|$	BT	ILS	GR	EA	ILS	GR	EA	ILS	GR	EA
		1	11	**6**	**6**	**6**	**6**	30	30	30	6,0	6,0	6,0
		2	13	**6**	**6**	**6**	**6**	30	30	30	6,0	6,0	6,0
4	6	3	12	**6**	**6**	**6**	**6**	30	30	30	6,0	6,0	6,0
		4	15	**6**	**6**	**6**	**6**	30	30	21	6,0	6,0	6,3
		5	14	**6**	**6**	**6**	**6**	30	30	24	6,0	6,0	6,2
		1	16	**6**	**6**	**6**	**6**	30	30	30	6,0	6,0	6,0
		2	18	**6**	**6**	**6**	**6**	30	30	30	6,0	6,0	6,0
5	6	3	20	**7**	**7**	**7**	**7**	30	30	19	7,0	7,0	7,4
		4	16	**6**	**6**	**6**	**6**	30	30	30	6,0	6,0	6,0
		5	18	**7**	**7**	**7**	**7**	30	30	30	7,0	7,0	7,0
		1*	26	9	**8**	**8**	**8**	30	30	25	8,0	8,0	8,2
		2*	27	10	**8**	**8**	**8**	30	30	21	8,0	8,0	8,3
6	6	3*	27	9	**8**	**8**	**8**	30	30	30	8,0	8,0	8,0
		4*	31	9	**8**	**8**	**8**	15	30	1	8,6	8,0	9,0
		5	29	**9**	**9**	**9**	**9**	30	30	30	9,0	9,0	9,0
		1*	34	9	**8**	**8**	**8**	30	30	30	8,0	8,0	8,0
		2*	34	10	**8**	**8**	**8**	29	28	8	8,0	8,1	8,7
7	6	3*	33	10	**9**	**9**	**9**	30	28	26	9,0	9,1	9,1
		4*	29	8	**7**	**7**	**7**	29	30	29	7,0	7,0	7,0
		5*	35	9	**8**	**8**	**8**	24	23	6	8,2	8,2	8,8
		1	46	**10**	**10**	**10**	**10**	28	11	24	10,1	10,6	10,2
		2	44	**9**	**9**	**9**	**9**	13	12	19	9,6	9,6	9,4
8	6	3*	48	11	**10**	**10**	**10**	30	29	30	10,0	10,1	10,0
		4*	48	13	**9**	**9**	**9**	11	5	1	9,6	9.9	10,0
		5*	44	15	**10**	**10**	**10**	30	29	30	10,0	10.1	10,0
		1*	53	15	**10**	**10**	**10**	18	3	2	10,4	11,1	10,9
		2*	49	12	**10**	**10**	**10**	19	7	17	10,4	10,7	10,4
9	6	3*	62	13	**10**	11	**10**	13	14	18	10,6	11,5	10,4
		4*	66	15	**12**	**12**	**12**	19	2	30	12,4	12,9	12,0
		5*	55	11	**9**	**9**	**9**	28	7	30	9,1	9,8	9,0
		1*	65	14	**11**	12	**11**	21	25	26	11,3	12,2	11,1
		2*	77	13	**12**	13	**12**	25	10	14	12,2	13,7	12,5
10	6	3*	63	16	**11**	**11**	**11**	17	3	2	11,4	11,9	12,2
		4*	71	13	**12**	**12**	**12**	27	1	18	12,1	13,4	12,1
		5*	70	15	**12**	**12**	**12**	26	10	10	12,1	12,7	12,7
Resumo				22	0	3	0	902	708	751	8,71	8.96	8,85

4. Conclusões

Os métodos desenvolvidos neste trabalho apresentam os melhores resultados da literatura conhecida. Na versão INUNDAÇÃO FIXA o ILS, GRASP e EA foram capazes de encontrar a maioria dos ótimos para quase todas as instâncias planares testadas. O ILS e o EA obtiveram os melhores desempenhos em muito pouco tempo de execução, se comparado com o exato, o EA foi ainda levemente mais robusto que o o ILS.

Nas instâncias genéricas, as heurísticas de INUNDAÇÃO FIXA obtiveram resultados muito melhores que o exato, chegando a apresentar vários movimentos de diferença. Porém, para instâncias mais densas o exato conseguiu encontrar uma quantidade relevante de ótimos, mas para as instâncias mais esparsas, isso não se repetiu. Isso faz dos algoritmos heurísticos ainda mais robustos, pois formam capazes de ficar muito acima do exato. Portanto, as instâncias mais esparsas são mais difíceis de serem resolvidas.

https://proceedings.science/p/85560?lang=pt-br

No que diz respeito à INUNDAÇÃO LIVRE, as instâncias escolhidas para o teste possuem bem menos vértices, mas devido à alta dificuldade de resolver o problema, elas ainda se mostraram bastante difíceis, pois o *backtracking* não foi capaz de encontrar muitos ótimos para essas instâncias. As meta-heurísticas porém, foram muito superiores e apresentaram resultados muito melhores. Ainda é possível destacar o ILS que obteve os melhores resultados para o problema na versão livre.

5. Agradecimentos

Os autores deste trabalho gostariam de agradecer a Universidade Federal Rural de Pernambuco - UFRPE, pelo apoio que foi essencial para a realização desta pesquisa.

Referências

Aschwanden, C. (2004). Spatial simulation model for infectious viral disease with focus on sars and the common flu. *37th Annual Hawaii International Conference on System Sciences, IEEE Computer Society*.

Barros, B. J. S., Pinheiro, R. G., e Souza, U. (2015). Métodos heurísticos e exatos para o problema de inundação em grafos. *XLVII Simpósio Brasileiro de Pesquisa Operacional*.

Clifford, R., Jalsenius, M., Montanaro, A., e Sach, B. (2011). The complexity of flood filling games. *arXiv.1001.4420v3 [cs.DS]*.

da Silva, A. R. V., Ochi, L. S., Souza, U., da Silva Barros, B. J., e Pinheiro, R. G. S. (2017). Algoritmos heurısticos e exatos para o problema de inundaç ao livre em grafos.

Fellows, M. R., Souza, U. S., Protti, F., e da Silva, M. D. (2015). Tractability and hardness of flood-filling games on trees. *Theoretical Computer Science 576*, p. 102–116.

Feo, T. A. e Resende, M. G. (1995). Greedy randomized adaptive search procedures. *Journal of global optimization*, 6(2):109–133.

Lagoutte, A., Noual, M., e Thierry, E. (2014). Flooding games on graphs. *Discrete Applied Mathematics*, p. 532–538.

Lourenço, H. R., Martin, O. C., e Stutzle, T. (2010). Iterated local search: Framework and applications. In Glover, F. e Kochenberger, G. A., editors, *Handbook of Metaheuristics, 2nd Edition*, volume 146, p. 363–397. Kluwer Academic Publishers.

Meeks, K. e Scott, A. (2012). The complexity of flood-filling games on graphs. *Discrete Applied Mathematics 160*, p. 959–969.

Silva, A. R. V. (2016). Uma meta-heurística eficiente para o problema de inundação em grafos. *XLVIII Simpósio Brasileiro de Pesquisa Operacional*.

Silva, A. R. V., Machado, R. V., e da Silva, M. D. (2017). Heurísticas construtivas para um problema de inundação em matrizes. *Proceeding Series of the Brazilian Society of Computational and Applied Mathematics*, 5(1).

Souza, U. S., Protti, F., e da Silva, M. D. (2013). Parameterized complexity of flood-filling games on trees. *Lecture Notes in Computer Science 7936*, p. 531–542.

https://proceedings.science/p/85560?lang=pt-br