Universidade Federal de Minas Gerais
Instituto de Ciências Exatas
Departamento de Ciência da Computação
Curso de Bacharelado em Ciência da Computação

Relatório final da matéria Projeto Orientado em computação II

Heurísticas para o problema Max-Sat

Francisco Eli Rodrigues de Lima
Orientador: Thiago Ferreira Noronha

Belo Horizonte
2º semestre de 2019

Resumo - Heurísticas são métodos muito usados para obter soluções aproximadas de problemas em que a solução ótima é difícil de se obter em um tempo razoável e troca-se o custo de se ter uma solução não exata pela velocidade da resposta. Para esse artigo, analisar-se-á metaheurísticas existentes na bibliografia atual para o problema do Max-Sat e propor-se-á novas metaheurísticas para o mesmo. Inicialmente, define-se o problema e explica-se a necessidade do uso de heurísticas, bem como uma revisão de heurísticas já propostas da literatura. Em seguida, propõe-se algumas metaheurísticas que serão comparadas com as existentes na literatura.

INTRODUÇÃO

O Max-Sat é uma generalização do problema de satisfabilidade, onde enquanto no problema de satisfabilidade dado um conjunto de cláusulas em CNF (Forma normal conjuntiva) quer-se uma atribuição das variáveis de forma que a expressão toda é verdadeira caso exista. No Max-Sat quer-se uma atribuição das variáveis de forma a satisfazer a maior quantidade de cláusulas. Claramente, no caso em que todas as cláusulas são satisfeitas, a solução do Max-Sat é solução do Sat, caso contrário, não existe solução para o Sat e a expressão não é satisfazível.

1 CONTEXTUALIZAÇÃO E TRABALHOS RELACIONADOS

Antes do uso de heurísticas para o Max-Sat deve-se explicar porque o seu uso é importante. Uma das principais formas de fazê-lo é mostrar que o problema é NP-Hard, ou seja, é pelo menos tão difícil de resolver quanto qualquer problema em NP e, exceto que P = NP, é um motivo razoável para criar heurísticas para o mesmo. No POC I, foi feita uma redução que prova que o resultado e, em mãos dessa realidade, foram implementadas heurísticas construtivas para o problemas, discutindo-se a limitação e a estrutura das instâncias onde as heurísticas eram mais adequadas. Como prosseguimento desse trabalho, metaheurísticas mais complexas e eficientes serão implementadas e novas metaheurísticas serão feitas e comparadas, novamente abordando as limitações e potencialidades dessas abordagens. O foco dessa segunda fase será o uso de um tipo

especial de heurísticas denominadas metaheurísticas: Uma metaheurística é um arcabouço algorítmico, que serve para uma ampla gama de problemas, que fornece um conjunto de estratégias para o desenvolvimento de algoritmos de otimização (Sörensen and Glover, 2013). Exemplos comuns de metaheurísticas incluem os famosos algoritmos evolucionários, busca tabu, simulated annealing etc, sendo essa uma lista não exaustiva das várias existentes.

2 METODOLOGIA

Inicialmente, foram implementadas e testadas 3 metaheurísticas largamente usadas na literatura para servir como baseline de comparação com as novas metaheurísticas propostas. A implementação das metaheurísticas escolhidas para a comparação foi, basicamente, uma reprodução dos pseudocódigos do seguinte trabalho disponível em http://people.cs.uchicago.edu/~pankratov/addl_exp.html e foi realizada na linguagem C++. Não foi usada nenhuma biblioteca específica, entretanto algumas partes do código tiveram que ser mudadas, pois não eram compatíveis com as versões mais novas do compilador g++. Uma outra mudança implementada foi a de que os códigos originais foram feitos para o problema do MaxSat com pesos, então foram reproduzidos para uma versão sem pesos. As instâncias usadas para os testes dessas metaheurísticas têm em torno de 1000 cláusulas e 100 variáveis e foi o mesmo conjunto usado para testar nossas próprias heurísticas. Para efeitos de comparação e reprodutibilidade, foram usadas instâncias de uma base pública bastante conhecida disponível em http://www.al.cm.is.nagoya-u.ac.jp/~yagiura/msat.html.

2.1.1 Primeira Heurística da literatura

A primeira chama-se NOLS+Tabu que configura-se numa busca tabu com fase de busca local NOLS(non oblivious local search), tal heurística é parecida com a OLS a seguir, mas usa-se de pesos para conjuntos de cláusulas. Cada conjunto é definido pelas cláusulas que estão atualmente satisfeitas por k variáveis.

2.1.2 Segunda Heurística da literatura

A segunda OLS+Tabu que configura-se numa busca tabu com fase de busca local OLS(oblivious local search), tal heurística, simplesmente faz um bit flip na variável que resulta no maior acréscimo após tal operação, ou seja, é uma heurística melhor aprimorante com vizinhança definida por soluções construídas a partir da atual mudando apenas um bit.

2.1.3 Terceira Heurística da literatura

A terceira foi a metaheurística Simulated Anneling**,** que é uma técnica para aproximar o máximo global de uma dada função. É uma metaheurística para aproximar a otimização global em um grande espaço de pesquisa para um problema de otimização. É frequentemente usado quando o espaço de pesquisa é discreto, por exemplo, no nosso caso em específico o problema do Max-Sat. Para problemas em que achar o máximo global aproximado em um tempo fixo, ela é bastante adequada.

2.2 Novas metaheurísticas propostas

Para as metaheurísticas propostas, todas as implementações seguem as implementações mais padrões dos pseudocódigos bem conhecidos para Tabu Search, Grasp e Ils. Todas as heurísticas são primeiro aprimorante, considerado-se a melhor solução global. Embora o pseudocódigo do grasp, ils, vnd e tabu, bem como as explicações acima feitas sejam suficientes para implementar os códigos, ainda é importante citar mais um fator usado para melhorar o tempo de execução do código.

Em todas as metaheurísticas é necessário checar a função objetiva da solução atual, que no caso é a quantidade de cláusulas feitas, em situações gerais é necessário checar toda ou quase toda a expressão lógica do max-sat para isso. Mas, em situações específicas, pequenas alterações podem evitar todo esse retrabalho. Por exemplo, quando um bit flip é feito em uma variável, pode-se calcular o novo valor usando-se o

da solução anterior e as alterações feitas somente nas cláusulas que contém essa variável. Por exemplo, se uma cláusula que contém X1, após um bit flip em X1, não é mais satisfeita, sabe-se que ela só tinha X1 satisfazendo ela, e portanto, basta decrementar a quantidade de cláusulas satisfeitas anterior em 1. Para isso, uma estrutura auxiliar é usada, que guarda para cada variável quais cláusulas ela está presente. Com todas essas considerações, é possível reproduzir os algoritmos feitos e obter os resultados obtidos, a exceção de tempos de execução que podem depender do ambiente de execução.

2.2.1 Metaheur 1

Para a primeira metaheurística, começa-se gerando uma solução aleatória, a partir da mesma faz-se sucessivas iterações de uma busca local, tais iterações são feitas até que passem X iterações sem melhorar a melhor solução obtida. A busca local é definida da seguinte forma, pega-se a melhor solução (não necessariamente aprimorante) das soluções geradas a partir da solução atual da iteração, mudando até 2 variáveis usando de bit flips. Somente considera-se variáveis que não sejam tabu, as variáveis que foram mudadas são colocadas como tabu nas próximas Y iterações

2.2.2 Metaheur 2

Para a segunda metaheurística, faz-se um Grasp. Faz-se repetidas iterações até X iterações sem melhora da melhor solução obtida, onde, em cada iteração, gera-se uma solução aleatória e faz-se uma busca local na mesma. Agora, a busca local é um VND, onde a primeira vizinhança são soluções que mudam 1 variável da solução atual, fazendo um bit flip, e a segunda vizinhança são todas as soluções mudando 2 bits da própria solução(por exemplo, mudando o valor da variável X1 com a X2)

2.2.3 ILSMaxSat

Para a terceira metaheurística, ILSmaxsat, faz-se um Iterated local search, onde novamente começa-se com uma solução aleatória e aplica-se uma busca local nela. Em seguida, fazem-se pequenas perturbações na solução atual, para tentar gerar soluções na região de atração de diferentes ótimos locais e assim, usando novamente da busca local achar novos ótimos locais. As perturbações são de k bit flips de k diferentes variáveis. Onde k é incrementado a cada X iterações sem achar um ótimo local melhor que o atual, quando um é achado, k é resetado para 1. X é um parâmetro que foi definido através de testes. O valor de k é incrementado a cada X iterações, mas tem um valor limite definido também por parâmetro passado ao código. Quando não são geradas soluções melhores, incrementa-se o k até seu valor limite e não consegue-se mais gerar ótimos locais melhor que o atual o ILS para.

3 Resultados

Usando dez instâncias e obtêm-se os seguintes resultados demonstrados nas tabelas abaixo. Para cada instância, faz-se uma média a partir de 10 execuções para obter uma maior segurança quanto ao resultado, uma vez que os algoritmos têm componentes pseudo-aleatórios, então executando mais de uma vez diminui-se os efeitos da seed sobre o resultado, além de que, tomando a média dos tempos obtêm-se maior confiança no tempo medido. Os parâmetros utilizados na execução de nossas heurísticas foram definidos após experimentos que consideraram uma variedade de opções, escolhendo, por fim, uma configuração que levou em conta o tempo de execução e os resultados obtidos, # de iterações = 10 e # de iterações sem melhoria = 2, além do k(# máxima de flips considerados na vizinhança) 9 para o ILS que foi usado o valor 5.

Tempo(s)	TS+OLS	TS+NOLS	SA	Metaheur1	Metaheur2	ILSmaxsat
Inst1	0.317	0.318	0.217	0.34	0.06	0.06
Inst2	0.315	0.303	0.356	0.27	0.07	0.06
Inst3	0.343	0.312	0.411	0.33	0.05	0.06
Inst4	0.338	0.301	0.311	0.31	0.07	0.05
Inst5	0.214	0.215	0.314	0.3	0.06	0.06
Inst6	0.192	2.001	0.193	0.33	0.07	0.07
Inst7	0.142	0.142	0.145	0.29	0.07	0.06
Inst8	0.22	0.302	0.237	0.3	0.05	0.07
Inst9	0.303	0.312	0.321	0.29	0.07	0.05
Inst10	0.375	0.386	0.401	0.34	0.06	0.06

Resultado	TS+OLS	TS+NOLS	SA	Metaheur1	Metaheur2	ILSmaxsat
Inst1	988.2	992.3	987.7	985.7	982.2	985.2
Inst2	986.4	989.4	988.6	983.6	984.1	985.2
Inst3	988.2	985.45	989.3	986.2	982.9	986.1
Inst4	991.1	990.9	991.2	985.7	983.1	983.9
Inst5	988.1	988.2	989.9	985.9	984.1	985.3
Inst6	988.5	988	987.78	984.5	982.2	985.3
Inst7	989.3	987.9	989.12	985.2	983.4	984.4
Inst8	971.4	971.1	984.8	985.5	982.7	986.2
Inst9	976.1	972.5	986.4	983.9	983.1	982.8
Inst10	980.1	976.9	987.9	987.1	984.3	985.9

4 CONCLUSÕES (E TRABALHO FUTUROS)

Percebe-se que, na média, as metaheurísticas propostas obtiveram tempos melhores, sendo que Meta1 e ILSmaxsat tiveram tempos bem melhores. Em relação ao resultado, percebe-se que foram obtidos resultados próximos aos das heurísticas da literatura com um tempo de execução médio menor. Dependendo da aplicação, as heurísticas propostas poderiam ser melhores em termos de custo benefício do que as da literatura, ainda mais se o tempo execução for um fator absolutamente primordial. Para trabalhos futuros, uma linha de trabalho promissora seria tentar analisar de forma mais formal os detalhes de complexidade computacional das metaheurísticas implementadas

e tentar categorizar os tipos de instâncias em que elas poderiam produzir melhores resultados. Além disso, uma proposta interessante também seria o estudo teórico dessas metaheurísticas no sentido de tentar estabelecer limites aproximativos, para alguns tipos de instâncias ou mesmo genéricos.

5 REFERÊNCIAS

[1] JOHNSON, D.S. 1973. Approximation algorithms for combinatorial problems. STOC '73: Proceedings of the fifth annual ACM symposium on Theory of computing, Austin, Texas, United States, ACM Press, New York, NY, USA, 38- 49.

[2]TEIXEIRA, G.F., GARCIA, B.B., 2004. Uma análise meta-heurística sobre o problema max-sat ponderado. XXXVI - SBPO O Impacto da Pesquisa Operacional nas Novas Tendências Multidisciplinares, São João del-Rei, Minas Gerais, Brasil, p. 1248-1259, 2004.

[3]]Max-Sat Project, Additional Experiments http://people.cs.uchicago.edu/~pankratov/addl_exp.html (visitado em 09/11/2018)

[4]Smyth, K., Hoos, H. H. e Stützle, T. (2003). Iterated robust tabu search for max-sat. In Proc. of the 16th Conf. of the Canadian Society for Computational Studies of Intelligence, 2671:129–144. Springer.

[5] Stutzle, T., Hoos, H., & Roli, A. (2002). A review of the literature on local search algorithms for max-sat (Technical Report AIDA-01-02). Technische Universitat Darm-tadt.

[6]Benchmark Instances and Program code for Max-sat http://www.al.cm.is.nagoya-u.ac.jp/~yagiura/msat.html (visitado em 10/10/2019)