Universidade Federal do Rio Grande do Sul - Instituto de Informática

Busca Gulosa Iterada aplicada ao Problema da Cobertura por Conjuntos

Tiago Mazzarollo de Paula - 00210271

tmpaula@inf.ufrgs.br

Porto Alegre, 06 de Junho de 2016

Índice

1. Introdução

O objetivo deste relatório é mostrar e avaliar os resultados obtidos na aplicação do método da Busca Gulosa Iterada (ILS do inglês *Iterated Local Search*) ao Problema da Cobertura por Conjuntos (PCC). Será feita a comparação entre os soluções obtidas com a heurística, as soluções obtidas com o *software GNU Linear Programming Kit* (GLPK) e as melhores soluções conhecidas para as instâncias dos problemas utilizadas.

2. Problema da Cobertura por Conjuntos

2.1 Descrição intuitiva

Dado um conjunto de elementos U e diversos subconjuntos compostos por elementos de U, busca-se encontrar um família de subconjuntos que unidos contenham todos os elementos de U. Cada subconjunto possui um custo associado, deseja-se encontrar a família de subconjuntos que minimize o custo total da cobertura e respeite as restrições do problema. Definição formal em [1]

2.2 Programação inteira

Sendo U a lista de elementos e F uma família de subconjuntos, tendo $m = |U|$ e $n = |F|$. $R_{ij} = 1$ se o subconjunto i contém o elemento j e $R_{ij} = 0$ caso contrário, sendo que $i \in [n]$ e $j \in [m]$. V_i representa o custo/valor do subconjunto i e S_i define se o subconjunto i faz parte da solução, com $i \in [n]$ e S_i e $R_{ij} \in \{0,1\}$. Sendo assim deseja-se:

minimar $\sum_{i \in [n]} V_i S_i$

sujeito a $\sum_{i \in [n]} R_{ij} S_i \geq 1, \quad \forall j \in [m]$

3. Busca Gulosa Iterada

3.1 Definição intuitiva

O algoritmo de Busca Gulosa Iterada utiliza busca local para caminhar no espaço de soluções. Após a busca local convergir para um mínimo/máximo local, são aplicadas perturbações na solução corrente buscando-se atingir novos pontos no espaço de soluções que façam com que a próxima aplicação da busca local convirja para um novo mínimo/máximo local ou, no caso ideal, para uma solução ótima. Definição formal da heurística em [2].

4. Características da implementação

4.1 Solução inicial

A solução inicial não possui nenhum subconjunto. Conforme definição em 2.2: a solução inicial utilizada é $S_i = 0, \ \forall i \in [n]$.

4.2 Valor da solução

O valor da solução corresponde à soma dos custo dos subconjuntos contidos na solução mais uma penalização para cada elemento de U que não está sendo coberto. O valor da penalização por elemento não coberto é $max(V) + 1$. Desta forma garante-se que deixar elementos descobertos é sempre mais custoso do que incluir um novo subconjunto, evitando assim que a busca convirja para uma solução inválida.

Um vetor C de tamanho m é utilizado para manter uma contagem de quantos subconjuntos contêm cada um dos elementos. Sempre que $C[j]$ é 0 uma penalização é somada ao valor da solução.

4.3 Definição e estratégia de escolha da vizinhança

Foram considerados soluções vizinhas aquelas que estão a um *flip* de distância da solução atual. Então para cada iteração da busca local são considerados n vizinhos.

Para escolha de vizinho foi utilizado *best improvement*. Portanto o vizinho com a solução de menor valor é selecionado para a próxima iteração. A busca para quando não houver nenhum vizinho melhor.

4.4 Perturbação

A perturbação seleciona aleatoriamente k índices de S e realiza um *flip*. O parâmetro k foi testado com três valores diferentes, conforme demostrado nos resultados.

4.5 Condição de parada

Duas condições de parada foram aplicadas. Foi definido um número máximo de iterações sem melhora, para os testes realizados este valor foi fixado em 1000 iterações. Além disso, foi aplicado o controle do tempo total de execução, limitado a 1800 segundos.

5. Resultados

5.1 Ambiente utilizado

Foi utilizado uma máquina com um processador Intel(R) Core(TM) i5-2310 CPU 2.90GHz com 4GB de memória disponível.

5.2 Resultados no GLPK

Os testes para todas as instâncias foram limitados a 1800 segundos. Em nenhuma das instâncias o processamento chegou ao final. Porém soluções muito próximas à ótima foram encontradas em alguns casos.

Junto com o tempo de execução, em segundos, é mostrado o *gap* apresentado pelo GLPK e quantidade de memória utilizada, em MB. Em algumas instâncias não houve memória suficiente para inciar o busca pela solução. As soluções obtidas são mostradas abaixo. A coluna DSO representa o desvio em relação à solução ótima.

	Solução encontrada	DSO	Gap	Memória	Tempo
scpcyc06	71	15,49%	11,70%	265,9	1800
scpcyc07	176	18,18%	26,10%	132,7	1800
scpcyc08	394	13,20%	33,40%	285	1800
scpcyc09	928	16,59%	36,80%	1492	1800
scpcyc10	-	-	-	insuficiente	-
scpcyc11	-	-	-	insuficiente	-
scpclr10	30	16,67%	8,00%	145,9	1800
scpclr11	33	30,30%	10%	93,9	1800
scpclr12	31	25,81%	45.2%	188,4	1800
scpclr13	37	37,84%	59,50%	489,1	1800

Nota-se que somente a instância scpclr12 teve um resultado próximo ao melhor conhecido. Nos demais casos o desvio é sempre maior que 10% chegando a quase 40% para a instância scpclr13. No geral, com o limite de tempo em 1800 segundos o GLPK não teve bons resultados. Em testes realizados com a menor instância(scpcyc06) que possui 192 subconjuntos, a solução ótima conhecida(60) foi obtida após aproximadamente três horas, porém até provar que essa era a solução ótima decorreram mais 4 horas.

O uso de memória também foi grande, comparado à implementação da Busca Gulosa Iterada como será mostrado abaixo. As instâncias scpcyc10 e scpcyc11 sequer finalizaram a etapa de relaxação linear devido à falta de memória disponível.

5.3 Resultados da heurística

5.3.1 Valores de parâmetros utilizados

O único parâmetro modificado foi a quantidade de *flips* realizados a cada perturbação, nomeado k. Foram feitos testes com $k = 4, k = 16$ e $k = 64$. Para cada valor de k foram realizados 10 testes com sementes diferentes para geração da perturbação. Sendo assim, foram feitos 30 casos de teste para cada instância do problema.

5.3.2 Resultados

Os resultados detalhados, com o valor obtido e o tempo de execução para cada um dos casos de teste, junto com as sementes utilizadas, são mostrados no Anexo I. Abaixo segue uma tabela com a média do valor obtido e do tempo de execução para cada instância e k avaliado, junto com o valor da solução inicial e da melhor conhecida.

Os valores das soluções inicias apresentados acabam sendo o pior valor possível, um vez que a solução inicial utilizada não possui nenhum subconjunto, portanto todos os elementos de U estão descoberto e m penalizações são somadas ao valor da solução.

Na tabela as colunas SI, SF e SO representam, respectivamente, a solução: inicial, final e ótima conhecida.

	k	SI	SF	Desvio SI-SF	Tempo	SO	Desvio SF-SO
scpcyc06	4	480	60	87,50%	1,0	60	0,00%
	16	480	60	87,50%	4,5	60	0,00%
	64	480	60	87,50%	11,0	60	0,00%
scpcyc07	4	1344	144	89,29%	16,0	144	0,00%
	16	1344	144	89,29%	81,0	144	0,00%
	64	1344	149	88,91%	136,0	144	3,36%
scpcyc08	4	3584	346	90,35%	275,5	342	1,16%
	16	3584	354,5	90,11%	890,5	342	3,53%
	64	3584	365	89,82%	1596,5	342	6,30%
scpcyc09	*						
scpcyc10	*						
scpcyc11	*						
scpclr10	4	1022	29	97,16%	2,0	25	13,79%
	16	1022	29	97,16%	5,5	25	13,79%
	64	1022	28	97,26%	13,5	25	10,71%
scpclr11	4	2046	29	98,58%	6,0	23	20,69%
	16	2046	29	98,58%	11,5	23	20,69%
	64	2046	27,5	98,66%	41,0	23	16,36%
scpclr12	4	4094	27	99,34%	14,0	23	14,81%
	16	4094	27	99,34%	27,0	23	14,81%
	64	4094	27	99,34%	41,0	23	14,81%
scpclr13	4	8190	32	99,61%	54,5	23	28,13%
	16	8190	31,5	99,62%	83,5	23	26,98%
	64	8190	30	99,63%	238,5	23	23,33%

O uso de memória não foi mostrado, pois este permaneceu muito abaixo dos 4GB disponíveis.

Em todos os casos de teste é possível notar que o tempo médio de execução aumenta junto com o valor de k. Este comportamento e justificável para a Busca Gulosa Iterada, porque após a aplicação de uma perturbação duas situações podem ocorrer: a solução irá para um ponto que convergirá para uma solução melhor do que a atual ou a irá para um ponto que retornará à solução anterior à perturbação. Caso ocorra a primeira situação, a quantidade de iterações que levarão para a nova solução é desconhecida. Caso a segunda

situação ocorra, serão necessárias k iterações da busca local para retornar a solução anterior à aplicação da perturbação. Portanto, quanto maior o k maior o tempo de execução.

Nos resultados das instâncias scpcyc06, scpcyc07 e scpcyc08 nota-se que em pouco tempo de execução, a solução ótima foi obtida para scpcyc06 e scpcyc07 e uma solução muito próxima da ótima para scpcyc08. Os melhores resultados obtidos com a heurística para cada uma das instâncias possuem desvios: $0\%, 0\%$ e 1.16%, obtidos em poucos segundos e são muito melhores do que os $15.46\%, 18.18\%$ e 13.2% do GLPK obtidos após atingir o limite 1800 segundos.

As instâncias scpcyc09, scpcyc10 e scpcyc11 tiveram o resultados omitidos porque a primeira busca local não foi concluída dentro do tempo limite definido. Removendo-se a restrição de tempo de execução estas instâncias finalizaram a primeira busca local com resultados 812, 1927 e 4303, respectivamente. Estes resultados são a média de um teste(somente uma semente) das três variações de k e o tempo de execução de cada teste foi, aproximadamente, 10.000 segundos. Os desvios dessas soluções em relação à solução ótima são: $4.68\%, 5.55\%$ e 5.00%. Com um tempo maior de execução a heurísticas chegou em valores muito próximos aos melhores conhecidos pata estas instâncias.

Os resultados das instâncias scpclr10, scpclr11, scpclr12 e scpclr13 não foram tão bons quanto os das instâncias citadas acima, mas em comparação com os resultados obtidos pelo GLPK e o tempo necessário para obtê-los podem ser considerados excelentes. Com melhores desvios: $10.71\%, 16.36\%, 14.81\%$ e 23.33%, comparados aos $16.67\%, 30.30\%, 45.2\%$ e 37.84% do GLPK.

6. Conclusões

Avaliando os dados apresentados é fácil perceber que os resultados obtidos pela heurística são bastante satisfatórios. A aplicação da Busca Gulosa Iterada é uma boa alternativa para a resolução do problema da Cobertura por Conjuntos, quando não é necessário obter a solução ótima ou ter a prova de otimalidade. As principais vantagens para o uso da heurística são a menor quantidade de tempo necessário para encontrar boas soluções, muitas vezes até ótimas soluções, e o baixo uso de memória.

Apesar dos bons resultados mostrados, melhorias podem ser feita. Um comportamento notado durante a realização dos teste é que a primeira solução encontrada, muitas vezes acabou sendo utilizada como solução final. Ou seja, após a aplicação da primeira busca local não ocorria a troca para uma solução melhor. Por exemplo, para a instância scpcyc07 com $k = 64$, a primeira aplicação da busca local encontrou o valor 149, depois ocorreram 1000 iterações e todas permaneceram no mínimo local encontrado inicialmente. Sendo assim, uma alteração que poderá trazer melhorias seria uma outra função de perturbação. Ao invés da realização de *flips* em posições aleatórias, utilizar uma função que visite outros pontos no espaço de soluções que levem a mínimos locais melhores.

De forma geral, a implementação e a realização dos teste foi bem interessante para trazer um conhecimento prático da dificuldade na resolução de problemas NP. Dando uma noção da importância das heurísticas como ferramentas para resolvê-los.

7. Bibliografia

[1] Bar-Yehuda, Reuven, and Shimon Even. "A linear-time approximation algorithm for the weighted vertex cover problem." *Journal of Algorithms* 2.2 (1981): 198-203.

[2] Lourenço, Helena R., Olivier C. Martin, and Thomas Stützle. *Iterated local search*. Springer US, 2003.

[3] Luenberger, David G., and Yinyu Ye. *Linear and nonlinear programming*. Vol. 2. Reading, MA: Addison-wesley, 1984.

Anexo I - Resultados dos casos de teste

São apresentados os valores obtidos e os tempos decorridos para cada caso de teste, com variação de k em $\{4, 16, 64\}$ para 10 sementes diferentes. A tabela abaixo mostra os valores e os tempos separados por vírgula para cada instância e para cada k utilizado. As sementes utilizadas foram: 123456, 578459, 543210, 567891, 987654, 987515, 412875, 165873, 985128, 247551. Os resultados são apresentados na mesma ordem da listagem acima.

	k	Valores	Tempos
scpcyc06	4	60,60,60,60,60,60,60,60,60,60	1,1,1,1,1,1,1,1,1,1
	16	60,60,61,60,61,61,61,60,60,60	5,4,4,6,4,4,4,8,5,6
	64	60,61,61,61,61,61,61,61,61,61	7,8,11,15,8,8,11,15,14,11
scpcyc07	4	144,144,144,145,144,144,144,144,144,144	11,16,19,16,30,12,16,24,16,13
	16	148,144,145,143,144,144,144,144,145,146	38,78,41,84,43,109,90,109,46,91
	64	149,149,149,149,149,149,149,149,149,149	125,136,147,129,136,134,141,142,142
scpcyc08	4	344,350,344,348,346,346,347,350,346,346	200,172,352,175,284,314,352,287,192,267
	16	353,355,355,349,355,359,358,353,354,352	1130,726,1121,1518,524,885,896,1804,749,858
	64	365,365,365,365,365,365,365,365,365,365	1629,1590,1475,1649,1628,1642,1603,986,849,847
scpcyc09	*		
scpcyc10	*		
scpcyc11	*		
scpclr10	4	29,29,28,28,29,27,29,29,29,29	3,2,2,4,2,3,3,2,2,2
	16	28,28,25,28,27,28,28,28,28,26	5,6,9,5,8,5,4,5,6,9
	64	28,28,28,26,26,28,28,27,26,28	14,11,23,13,18,11,13,12,22,15
scpclr11	4	29,29,29,29,29,29,29,29,29,29	4,7,7,5,7,5,6,6,6,8
	16	29,29,29,29,29,29,29,29,29,29	11,10,11,12,15,13,13,11,12,11
	64	27,28,28,28,23,28,28,27,24,27	29,30,31,38,75,40,43,42,83,48
scpclr12	4	29,26,26,26,25,28,26,28,28,30	13,14,18,22,29,13,14,15,13,13
	16	30,29,30,27,26,26,26,29,27,26	22,21,22,39,35,24,35,29,26,28
	64	27,27,26,26,25,27,29,29,27,24	62,50,79,70,110,54,51,50,48,83
scpclr13	4	32,32,32,32,32,32,32,32,32,32	48,55,69,42,54,55,55,55,44,47
	16	30,32,32,31,32,31,31,32,31,32	176,70,77,141,66,80,130,72,114,87
	64	31,30,30,30,30,30,30,30,30,31	242,210,283,239,364,211,238,200,213,309
* Limite de tempo execedido			