Otimização no rank-1 de Chvátal-Gomory para o problema Set Covering

João C. Abreu[1]

[1]*Universidade Federal de Minas Gerais, DCC,*
Avenida Antônio Carlos 6627, Belo Horizonte, Brazil,
joao.junior@dcc.ufmg.br

Abstract Esse trabalho apresenta o problema Set Covering e um algoritmo para gerar todos os cortes de rank-1 de Chvátal-Gomory para esse problema.

Keywords: Set Covering, Chvátal-Gomory, Desigualdades válidas

1. Introdução

Dados $M = \{1, ..., m\}$ e $N = \{1, ..., n\}$ dois conjuntos. Seja M_1, M_2, ..., M_n uma coleção de subconjuntos de M com um custo c_j associado a cada um desses subconjuntos. Uma cobertura de M é um subconjunto $F \subset N$ tal que $\cup_{j \in F} M_j = M$. O problema que consiste em encontrar esse subconjunto $F \subset N$ de custo mínimo é denominado o problema Set Covering(SCP). Segundo Balas[1] esse problema é *np-dificil* e não se conhece uma boa caracterização do politopo desse problema.

Em [1], Balas caracteriza a classe de inequações válidas para o SCP com coeficientes iguais a 0,1 ou 2 e apresenta condições necessárias e suficientes para essas restrições serem não dominadas por outras. Esse paper também mostra que todas as inequações desse tipo são às únicas a formarem o feixo 1 de Chvátal-Gomory.

Em [2] Saxena extende o trabalho de Balas[1] e apresenta uma caracterização da classe de inequações válidas para o SCP com coeficientes iguais a 0, 1, 2 ou 3 e propõe um problema *np-dificil* para gerar todas essas inequações a partir de soluções fracionárias do SCP.

Em [5], Beasley propõe um algoritmo para resolver o SCP baseado nos resultados teóricos propostos em [1].

O Objetivo desse trabalho é otimizar o problema SCP através do rank-1 de Chvátal-Gomory.

A seção 2 apresenta uma modelagem matemática para esse problema, a seção 3 apresenta

um algoritmo para gerar as restrições Chvátal-Gomory de rank1 e resolver o SCP com essas restrições, a seção 4 apresenta um pequeno exemplo da utilização do algoritmo proposto na seção 3, a seção 5 apresenta os experimentos computacionais obtidos aplicando o algoritmo da seção 3 nas instâncias presentes na literatura e a seção 6 apresenta as conclusões.

2. Modelagem Matematica

Essa seção apresenta a formulação para o SCP proposta em [7]. Para formular o SCP como um problema de otimização inteira, é introduzida uma matriz de incidência A de tamanho mxn para a coleção de subconjuntos $M_j, \forall j \in N$, com as entradas dadas por:

$$a_{ij} = \begin{cases} 1; & \text{se } i \in M_j, \\ 0; & \text{caso contrário} \end{cases} \tag{1}$$

Nessa formulação, a função objetivo (5) minimiza o custo da cobertura procurada. Os valores constantes $c_j, \forall j \in N$ são os custos de cada subconjunto M_j. A variável de decisão x_j é igual a 1 quando $j \in F$ e 0, caso contrário, onde $F \subset N$ é a cobertura procurada.

$$\min \sum_{j \in N} c_j x_j \tag{2}$$

Sujeito à:

$$Ax \geq 1 \tag{3}$$

$$x \in \{0,1\}^n \tag{4}$$

3. Algoritmo

Essa seção apresenta um algoritmo para gerar todos os cortes de Chvátal-Gomory de rank-1 e foi retirado de [6]. A figura 1 apresenta esse algoritmo. Na linha 2 o modelo da seção 2 é relaxado, isto é, as variáveis x agora pertencem ao intervalo $[0,1]$, e então é resolvido. Os laços das linhas 3 a 10 são executados até que uma solução inteira seja encontrada ou que o tempo máximo ocorra. A linha 3 verifica se a atual solução x é inteira, caso positivo, nada mais precisa ser feito, caso negativo entra-se no loop. A linha 4 verifica se existe algum corte de Chvátal-

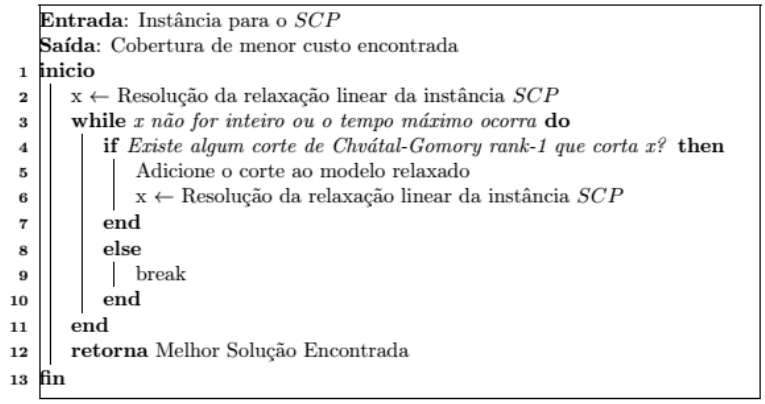

Figure 1: Algoritmo para encontrar cortes de Chvátal-Gomory de rank-1 para o SCP

Gomory de rank-1 que corta x, caso positivo a linha 5 adiciona o corte encontrado no modelo relaxado e a linha 6 resolve o modelo relaxado com esse novo corte. Caso nenhum corte seja encontrado a linha 9 é executada e o algoritmo termina. Para verificar se existe algum corte de Chvátal-Gomory de rank-1 que corta uma solução fracionária x é utilizado um formulação de programação linear inteira mista baseada na formulação do problema de separação *np-dificil* proposta em [6]. A formulação é composta pela função objetivo (5) e pelas restrições (6)-(10).

$$\min \sum_{j \in N} \alpha_j x_j - \alpha_0 \tag{5}$$

Sujeito à:

$$0 \leq \alpha_j - u^T A_j \leq 1 - \delta, \forall j \in N \tag{6}$$

$$0 \leq \alpha_0 - u^T b \leq 1 - \delta \tag{7}$$

$$0 \leq u_i \leq 1 - \delta, \forall i = 1, ..., m \tag{8}$$

$$\alpha_0 \leq \sum_{j \in N} \alpha_j, \tag{9}$$

$$\alpha_0, \alpha_j \text{ inteiro}, \forall j \in N \tag{10}$$

Dada uma solução fracionária x para o SCP, se a formulação anterior possui solução negativa, então teremos $\lceil u^T A_j \rceil x_j \leq \lceil u^T b \rceil$, então a restrição $\lceil u^T A_j \rceil x_j \geq \lceil u^T b \rceil$ é um restrição válida para o SCP que corta x. A restrição (6), juntamente com a restrição (10) garantem que α_j é o menor inteiro, maior ou igual a $u^T A_j$, ou seja $\alpha_j = \lceil u^T A_j \rceil$. A restrição (7), juntamente com a restrição (10) garantem que α_0 é o menor inteiro, maior ou igual a $u^T b$, ou seja $\alpha_0 = \lceil u^T b \rceil$. A

restrição (9) garante que os cortes gerados continuaram a manter as variáveis x com um valor máximo 1.

4. Exemplo

Essa seção apresenta uma pequena instância para o problema SCP. Essa instância é então passada como entrada para o algoritmo da seção 3 e também é estudada com o software PORTA. Para essa instância, o conjunto $M = \{1, 2, 3, 4\}$ e o conjunto $N = \{1, 2, 3, 4, 5, 6\}$. A matriz de incidência A para a coleção dos subconjuntos $M_j, \forall j \in N$ é dada abaixo.

$$\begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \end{pmatrix}$$

O custo c_j de cada um dos subconjuntos M_j é dado por $c = (60, 7, 11, 5, 8, 5)$.
Ao submeter essa instância ao software PORTA nós encontramos 38 pontos viáveis. Sendo que desses 38 pontos viáveis 13 satisfazem na igualdade a restrição formada pela linha 1, 14 satisfazem na igualdade a restrição formada pela linha 2, 12 satisfazem na igualdade a restrição composta pela linha 3 e 22 satisfazem na igualdade a restrião formada pela linha 4 da matriz de incidência A. Ao executar o algoritmo da seção 3, a solução encontrada pela relaxação linear é $x = (0.0, 0.5, 0.5, 0.0, 0.5, 0.5)$, com um valor objetivo igual a 15.5, então o algoritmo encontra a restrição $2x_1+1x_2+2x_3+1x_4+1x_5+1x_6 \geq 3$, que pode ser obtida com o arredondamento da soma de um múltiplo das restrições da matriz de incidência A, onde as linhas são respectivamente multiplicadas por $u = (0.99, 0.495, 0.505, 0.505)$. A soma das linhas utilizando esses multiplicadores nos trazem a seguinte inequação: $1.495x_1 + 1x_2 + 2x_3 + 0.495x_4 + 1x_5 + 0.99x_6 \geq 2.495$, fazendo o arredondamento para o inteiro maior ou igual aos coeficientes, teremos a seguinte inequação $2x_1 + 1x_2 + 2x_3 + 1x_4 + 1x_5 + 1x_6 \geq 3$, que é exatamente o corte de Chvátal-Gomory de rank-1 encontrado pelo algoritmo da seção 3. Utilizando o software PORTA é confirmado que essa restrição é válida para essa instância do SCP, pois os mesmos 38 pontos continuam sendo gerados como pontos viáveis. Claramente essa restrição corta o ponto $x = (0.0, 0.5, 0.5, 0.0, 0.5, 0.5)$, pois utilizando esse ponto nessa nova restrição, teríamos $2.5 \geq 3$ que é falso. A segunda iteração do algoritmo vai então encontrar o ponto $x = (0.0, 0.0, 1.0, 1.0, 0.0, 0.0)$, com valor objetivo

igual a 16, que é exatamente a solução encontrada executando o modelo de programação inteira da seção 2 para essa instância. Rodando no software PORTA essa instância, agora com essa nova restrição, novamentes temos 38 pontos viáveis, porém apenas 5 desses pontos satisfazem essa nova inequação na igualdade. Rodando o software PORTA passando esses 38 pontos viáveis e procurando uma caracterização do politopo do SCP, encontramos a seguinte matriz de incidência:

$$\begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

com o vetor $b = (1, 1, 1, 1, 2)$. Observe que a restrição adicional encontrada pelo software PORTA é muito parecida com a restrição encontrada pelo algoritmo da seção 3, porém a restrição encontrada pelo algoritmo é dominada pela restrição encontrada pelo software PORTA.

5. Experimentos Computacionais

Os experimentos computacionais foram executados em uma máquina Intel Dual-Core de 2.81 GHz de clock e 2GB de memória RAM, rodando o sistema operacional Linux. O modelo matemático apresentado em 2 foi implementado no Ilog CPLEX 12.5.1 e o algoritmo da seção 3 foi implementado em Python 2.7, sendo que o otimizador utilizado para resolver os modelos lineares presente nesse algoritmo foi o Ilog CPLEX 12.5.1. Foram utilizados quatro conjuntos de instâncias de testes nos experimentos computacionais e essas instâncias foram retiradas de [9]. Nessas instâncias de testes cada linha da matriz de incidência A é coberta por pelo menos duas colunas e cada coluna cobre pelo menos uma linha. O custo c_j de cada coluna j está entre $[1, 100]$. A tabela 1 resume esses conjuntos de instâncias. A coluna 1 dessa tabela representa o identificador do conjunto da instância de teste, as colunas 2 e 3 mostram, respectivamente, o número m de linhas e n de colunas da matriz de incidência A. A coluna 4 representa a densidade da matriz A que é calculado pelo quantidade de 1's dessa matriz dividido pela quantidade total de elementos de A que é igual a mn e a coluna 5 mostra a quantidade de problemas em cada conjunto. Os dez problemas do conjunto 4 são nomeados como scp41-scp410, os cinco problemas

do conjunto 6 são nomeados como scp61-scp65, e os problemas do conjunto A e B são nomeados respectivamente como scpa1-scpa5 e scpb1-scpb5.

Conjunto	Linhas	Colunas	Densidade	Problemas
4	200	1000	2	10
6	200	1000	5	5
A	300	3000	2	5
B	300	3000	5	5

Table 1: Detalhes das instâncias de testes utilizadas

No experimento desse trabalho foi comparado a performance do modelo proposto na seção 2, que será chamado aqui de IP, com o algoritmo proposto na seção 3, que será chamado $ARank1$. O modelo IP foi executado através do CPLEX com todos os parâmetros default. O modelo presente no algoritmo $ARank1$ foi executado pelo CPLEX com um tempo máximo de 120 segundos e foi setado para que o CPLEX encontra-se no máximo cinco soluções inteiras, explorando no máximo 50000 nós na árvore de Branch-and-Bound, encontra-se soluções com um valor objetivo sempre inferior a -0.05 e a ênfase na busca de soluções foi setado 4. O modelo IP e o algoritmo $ARank1$ foram executados com um tempo de execução máximo de 7200 segundos.

A tabela 2 apresenta os resultados obtidos para o conjuntos de instância 4 e 6 e a tabela 3 apresenta os resultados para o conjuntos de instância A e B. Nessas tabelas a coluna 1 mostra o nome da instância de teste, as colunas 2 e 3 são resultados referentes ao modelo IP e as colunas 4,5,6 e 7 são resultados referentes ao algoritmo $ARank1$. A coluna 2 apresenta o custo da solução obtido pela modelo IP e a coluna 3 apresenta o tempo consumido para encontrar essa solução. A coluna 4 mostra o valor da relaxação linear obtida para o SCP no início do algoritmo $ARank1$, a coluna 5 apresenta o custo da solução obtido após procurar e inserir os cortes de Chvátal-Gomory de rank 1, a coluna 6 traz a quantidade de cortes que o algoritmo $ARank1$ conseguiu adicionar e a coluna 7 mostra o tempo consumido pelo algoritmo $ARank1$. Para todas as instâncias do conjunto 4 e 6 o CPLEX conseguiu encontrar soluções ótimas em um tempo muito pequeno, conforme pode ser observado pelas colunas 2 e 3 da tabela 2. Para as instâncias scp41, scp42, scp43, scp45 e scp47 nenhum corte é possível de ser adicionado, pois a relaxação linear já provém uma solução inteira ótima para o SCP, conforme pode ser observado na coluna 4, linhas 1,2,3,5 e 7 da tabela 2. Para todas as outras instâncias desse conjunto o algoritmo $ARank1$ conseguiu adicionar cortes de Chvátal-Gomory de rank-1, como pode ser

observado pelas linhas 4,6,8,9,10,11,12,13,14 e 15 na coluna 6 da tabela 2. Para esse conjunto, a única instância que o algoritmo $ARank1$ conseguiu resolver na otimalidade foi a instância scp410, conforme pode ser observado pela coluna 5, linha 10 da tabela 2. Nessa mesma tabela, retirando-se as instâncias que possuem uma relaxação linear inteira, pode-se observar que o tempo gasto pelo algoritmo $ARank1$ foi bastane alto, conforme a coluna 7.

Instância	IP		$ARank1$			
	Custo Solução	Tempo(s)	Relaxação Linear	Custo Solução	#Cortes	Tempo(s)
scp41	429	0.84	429.00	429.00	0	0.00
scp42	512	0.85	512.00	512.00	0	0.00
scp43	516	0.86	516.00	516.00	0	0.00
scp44	494	0.86	494.00	494.00	6	863.71
scp45	512	0.85	512.00	512.00	0	0.00
scp46	560	0.89	557.25	558.94	32	1038.07
scp47	430	0.83	430.00	430.00	0	0.00
scp48	492	0.97	488.67	490.67	65	6118.55
scp49	641	0.90	638.54	639.87	75	8054.13
scp410	514	0.90	513.50	514.00	14	1349.04
scp61	138	1.23	133.14	133.53	52	7267.44
scp62	146	2.06	140.46	141.15	50	7284.06
scp63	145	1.26	140.13	141.46	72	7289.33
scp64	131	0.96	129.00	130.08	77	7251.63
scp65	161	1.94	153.35	154.13	50	7370.46

Table 2: Comparação entre os custos da solução e tempos obtidos entre o modelo IP e o algoritmo $ARank1$ para as instâncias do conjunto 4 e 6.

Para todas as instâncias do conjunto A e B o CPLEX conseguiu encontrar soluções ótimas em um tempo pequeno, conforme pode ser observado pelas colunas 2 e 3 da tabela 3. Para essas instâncias o algoritmo $ARank1$ não conseguiu encontrar a solução ótima para nenhuma delas, conforme pode ser observado pela coluna 5 da tabela 3. O algoritmo $ARank1$ só conseguiu encontrar cortes de Chvátal-Gomory de rank-1 para as instâncias scpa3, scpa5 e scpb5, conforme pode ser observado pela coluna 7 e linhas 3,5 e 10 da tabela 3. O tempo máximo de 120 segundos para o modelo de separação no algoritmo $ARank1$ pode ter sido a causa de não encontrar cortes válidos para as demais instâncias. Para a instância scpa3, mesmo adicionando cortes válidos a solução encontrada possui o mesmo custo da solução na relaxação linear, conforme pode ser observado, comparando-se a coluna 4 e 5 da linha 3, indicando degenerações nas soluções.

6. Considerações finais

Esse trabalhou apresentou o problema Set Covering(SCP) e um algoritmo para gerar as restrições de Chvátal-Gomory de rank-1 para esse problema. O algoritmo apresentando foi com-

Instância	IP		$ARank1$			
	Custo Solução	Tempo(s)	Relaxação Linear	Custo Solução	#Cortes	Tempo(s)
scpa1	253	9.95	246.84	246.84	0	251.75
scpa2	252	9.87	247.50	247.50	0	252.95
scpa3	232	9.48	228.00	228.00	6	1685.40
scpa4	234	8.76	231.40	231.40	0	248.60
scpa5	236	8.64	234.89	235.02	25	5826.96
scpb1	69	10.08	64.54	64.54	0	246.67
scpb2	76	10.87	69.30	69.30	0	256.28
scpb3	80	9.67	74.16	74.16	0	252.03
scpb4	79	11.52	71.22	71.22	0	250.54
scpb5	72	9.86	67.67	67.67	2	741.67

Table 3: Comparação entre os custos da solução e tempos obtidos entre o modelo IP e o algoritmo $ARank1$ para as instâncias do conjunto A e B.

parado com a resolução da modelagem matemática do SCP pelo CPLEX. O CPLEX mostrou-se muito eficiente para resolver as instâncias de testes selecionadas, enquanto o algoritmo não obteve bons resultados, conforme Balas[1], o fato desse algoritmo não apresentar uma boa perfomance na prática já era esperado. Todo código fonte produzido por esse trabalho pode ser obtido no endereço eletrônico: $https://github.com/joaojunior/feixo1\_2scp$

References

[1] Balas, Egon and Ng, ShuMing. On the Set Covering Polytope: I. All the Facets with coefficients in {0,1,2}. Mathematical Programming, 43:57–69, 1989.

[2] Anureet Saxena. On the Set Covering Polytope: All the Facets with coefficients in {0,1,2,3}. GSIA Working Paper, 2004.

[3] Balas, Egon. Cutting planes from conditional bounds: a new approach to set covering. Combinatorial Optimization, volume 12, pages 19–36, 1980.

[4] Balas, Egon and Ho, Andrew. Set Covering algorithms using cutting planes, heuristics, and subgradient optimization: a computational study. Combinatorial Optimization, volume 12, pages 37–60, 1980.

[5] Beasley, J. E. An algorithm for set covering problem. European Journal of Operational Research, 31:85–93, 1987.

[6] Fischetti, Matteo and Lodi, Andrea Optimizing over the First Chvàtal Closure. Mathematical Programming, 110:3–20, 2007.

[7] Bertsimas, Dimitris and Weismantel, Robert. Formulations. Optimization over Integers, 1:5–6, 2005.

[8] Beasley, J. E. OR-Library: distributing test problems by electronic mail. Journal of the Operational Research Society, 41:1069–1072, 1990.

[9] Fischetti, Matteo and Lodi, Andrea Optimizing over the first Chvátal closure. Mathematical Programming, 41:1069–1072, 1990.