UMA FORMA DISTRIBUÍDA DE BUSCA ESTOCÁSTICA PARA OTIMIZAÇÃO COMBINATÓRIA †

Valmir C. Barbosa

Programa de Engenharia de Sistemas e Computação
COPPE/UFRJ
Caixa Postal 68511
21945 Rio de Janeiro - RJ

Sumário

Este trabalho descreve uma aplicação de processamento distribuído à aproximação da solução de problemas NP-completos em Otimização Combinatória. O método básico utilizado é conhecido como "Simulated Annealing", e consiste em uma busca estocástica inspirada nos princípios da Física Estatística.

Para uma vasta classe de problemas, o método pode ser paralelizado, permitindo que a busca possa se realizar sobre um conjunto relativamente grande de variáveis simultaneamente. Este trabalho descreve uma implementação distribuída do método através da associação de um processador dedicado a cada uma das variáveis do problema. Entre os problemas passíveis de paralelização, encontra-se o problema de encontrar-se uma cobertura de vértices de tamanho mínimo em um grafo.

A implementação proposta apresenta um grau de concorrência que depende da forma como as diversas variáveis do problema se interrelacionam.

1. Introdução

Um problema típico em Otimização Combinatória pode ser expresso da seguinte forma. Seja $X = \{X_1, \ldots, X_n\}$ um conjunto de variáveis tomando valores de um conjunto comum D, e FS um conjunto de pontos viáveis em D^n. O problema é encontrar um mínimo global de uma função f da forma

$$f : D^n \rightarrow R$$

† Pesquisa para este trabalho foi realizada com o apoio da CAPES, bolsa número 3638/81-

no conjunto viável FS. Em outras palavras, deseja-se encontrar um ponto $x^* \in FS$ tal que $f(x^*) \leq f(x)$ para todo $x \in FS$.

O problema de encontrar um mínimo global de f em FS e tipicamente difícil, especialmente se f não possui nenhuma característica simplificadora, como por exemplo convexidade. A razão para essa dificuldade é que a maioria dos métodos iterativos para resolver o problema terminam por estancar em mínimos locais, muitas vezes significativamente distantes de um mínimo global.

Além disso, grande parte dos problemas de maior interesse em Otimização Combinatória são NP-completos em suas formulações como problemas de decisão [10]. Com relação a esses problemas, não se conhece nenhum algoritmo que os resolva em um tempo de complexidade polinomial no tamanho da entrada. O mais grave é que bastaria que um algoritmo polinomial fosse encontrado para resolver qualquer problema NP-completo, e esse mesmo algoritmo representaria uma solução polinomial para todos os problemas NP-completos [13]. Anos de pesquisa na área não produziram nenhum algoritmo com essa característica, e como conseqüência existe um sentimento generalizado de que problemas NP-completos não podem ser resolvidos em tempo polinomial. Apesar de também não haver esperança de se encontrar uma prova desse fato a curto prazo, tudo indica que uma boa estratégia é buscar soluções alternativas.

Houve no passado um grande esforço no sentido de desenvolver procedimentos heurísticos para aproximar a solução de vários problemas NP-completos de relevância em outras áreas. Uma outra alternativa é explorada neste trabalho, consistindo essencialmente na realização de uma busca estocástica em FS, o conjunto de pontos viáveis.

As próximas seções estão organizadas da seguinte forma. A versão original (centralizada) de "Simulated Annealing" é descrita na Seção 2. A Seção 3 contém a descrição de uma classe de problemas aos quais a versão distribuída do método pode ser aplicada. A implementação distribuída é então descrita na Seção 4. Exemplos são fornecidos na Seção 5. As Seções 6 e 7 tratam de problemas mais específicos gerados pela implementação distribuída. Estes são o escalonamento dos diversos processadores para operação, e o procedimento para recuperar o resultado ao final da computação. Conclusões seguem na Seção

2. "Simulated Annealing"

A aplicação de uma técnica conhecida como "Simulated Annealing" para aproximar a solução ao problema de encontrar um mínimo global de f em FS foi proposta recentemente por Kirkpatrick *et al.* em [14]. Esse método origina de uma técnica semelhante usada por Metropolis *et al.* [17] para simular o comportamento de sistemas físicos complexos em temperaturas fixas. O nome "Simulated Annealing" tem sua origem no procedimento de laboratório conhecido pelos físicos como "annealing", cujo objetivo é trazer um material a seus estados mais baixos de energia, através de um processo lento de resfriamento a partir de uma temperatura relativamente alta. Trata-se de um processo delicado, uma vez que existe o risco de o processo estancar em estados indesejáveis de energia "localmente" mínima, caso o processo de resfriamento não seja suficientemente vagaroso.

Por analogia com o processo físico, a proposta em [14] consiste essencialmente em uma busca estocástica sobre FS, de forma a imitar o comportamento dos componentes microscópicos do material em resfriamento. A proposta é executar "pulos" probabilísticos entre os pontos de FS, conseqüentemente permitindo pulos ocasionais em que o valor de f cresce. Esses pulos devem ser controlados pela distribuição de Boltzmann, parametrizada por um parâmetro que decresce lentamente com o tempo, como a temperatura no sistema físico.

Uma descrição mais detalhada do método é a seguinte. Seja $x \in FS$ o ponto no qual a simulação se encontra em um determinado instante, e $x' \in FS$ um ponto que pode ser obtido a partir de x através de uma troca no valor de uma das variáveis em X. A simulação passa de x para x' com probabilidade 1 se $f(x') \leq f(x)$ — isto é, se o valor de f decresce — e com probabilidade

$$e^{\frac{1}{T}(f(x)-f(x'))},$$

caso contrário (um pulo em que o valor de f aumenta). T é o parâmetro que deve imitar o comportamento da temperatura — ser lentamente diminuido a partir de um valor inicial alto. Pela descrição do método, vê-se intuitivamente que a granularidade da busca estocástica aumenta à medida que T decresce. Prova-se que, no limite em que $T \to 0$, a simulação se restringe a pontos em FS nos quais f é globalmente mínima. Essa prova encontra-se atualmente em várias versões, [11] sendo um exemplo.

O método tem sido extensivamente utilizado nas diversas áreas em que se encontram

problemas de difícil tratamento. No caso particular de problemas de Otimização Combinatória, vale a pena citar os experimentos descritos em [1], nos quais alguns problemas típicos em Otimização Combinatória foram utilizados para testar o método. As conclusões iniciais são que o método se comporta surpreendentemente bem para alguns problemas, enquanto não consegue superar as melhores heurísticas existentes para outros.

3. Uma Classe de Problemas

O método "Simulated Annealing", como apresentado na seção anterior, pode em muitos casos ser paralelizado. Uma tentativa particularmente bem sucedida nesse sentido encontra-se descrita em [7], onde o Problema do Caixeiro Viajante é abordado. A proposta em [7] utiliza um hipercubo [19] para a obtenção de paralelismo no método.

Neste trabalho, a busca por paralelismo em implementações de "Simulated Annealing" toma uma direção diferente, no sentido em que a intenção é caracterizar funções que se prestem a um método de busca estocástica distribuída. Mais especificamente, pretende-se investigar a possibilidade de associar a cada variável em X um processador, de forma que cada procesador necessite comunicar-se com apenas um subconjunto reduzido e bem delimitado dos processadores restantes, e que em conjunto eles possam aproximar um mínimo global de f em FS.

Para tanto, suponha que f pode ser escrita sob a forma

$$f(x) = \sum_{Y \subseteq X} g_Y(x),$$

para todo $x \in D^n$, onde $g_Y(x)$ não é constante somente se Y não é "muito grande", e depende apenas das coordenadas de x correspondentes a variáveis em Y. A aplicação de "Simulated Annealing" a funções como esta encontra-se descrita em [11].

Duas variáveis são ditas *vizinhas* se ambas pertencem a um subconjunto Y de X tal que $g_Y(x)$ não é constante. A busca estocástica toma a seguinte forma. Para $1 \leq i \leq n$, atribua valor x_i à variável X_i com probabilidade

$$\pi(X_i = x_i | X_j = x_j, j \neq i),$$

onde π é a distribuição de Boltzmann, aqui dada por

$$\pi(x) = \frac{1}{Z} e^{-\frac{1}{T} f(x)},$$

para todo $x \in D^n$, Z sendo uma constante de normalização e T o parâmetro-temperatura que temos considerado. Devido à forma de f, vê-se que

$$\pi(X_i = x_i | X_j = x_j, j \neq i) = \pi(X_i = x_i | X_j = x_j, X_j \text{ vizinha de } X_i).$$

Conforme se nota, a decisão sobre o valor de X_i depende somente dos valores das variáveis X_j que são vizinhas de X_i.

Esse tipo de localidade existente nas decisões sobre o valor das diversas variáveis durante a simulação indica as seguintes linhas gerais para uma implementação distribuída da busca estocástica. Associa-se um processador a cada variável; se duas variáveis são vizinhas, os processadores correspondentes são conectados entre si por um canal bidirecional de comunicações. Como conseqüência da forma da função f sendo otimizada, processadores que não são conectados um ao outro podem operar concorrentemente sobre os valores das variáveis respectivas. O que não é imediatamente claro, porém igualmente verdadeiro, é que variáveis vizinhas não podem ser atualizadas concorrentemente. Isso corresponderia à atualização de uma variável com base em valores obsoletos de suas vizinhas. Como conseqüência, o sistema poderia se encontrar em um estado cuja probabilidade estaria em desacordo com π, a distribuição em que a simulação se baseia.

É importante observar que essa formulação de "Simulated Annealing" é apropriada à minimização de f sem restrições, isto é, à obtenção de um mínimo de f sobre D^n, possivelmente inviável. Todavia, é possível garantir que todos os mínimos globais de f encontram-se em FS através da incorporação em f de funções de penalidade, ou técnicas similares [16].

4. Processamento Distribuído

A variação de "Simulated Annealing" discutida no item anterior pode ser implementada distribuidamente através da associação de um processador a cada variável em X. Processadores são conectados entre si se as respectivas variáveis são vizinhas. Um grafo não-orientado $G = (N, E)$ pode ser associado ao sistema, de forma que cada nó em N corresponde a um processador e cada arco em E a um canal de comunicações.

O procedimento executado por cada processador consiste em atualizar o valor de sua variável com base nos valores das variáveis vizinhas. Pela forma como os processadores são interconectados, esses valores são obtidos diretamente de processadores vizinhos.

Observe que só há ganho com essa paralelização do método se o grafo G é esparso. Serão vistos na próxima seção alguns problemas típicos em que esse é o caso.

A implementação distribuída proposta acima gera dois problemas principais, descritos e analisados em [2,9]. São os seguintes:

1. **Escalonamento 'e processadores vizinhos para operação.** Esse escalonamento torna-se necessário face à restrição de que processadores vizinhos não podem operar concorrentemente. O método adotado é descrito na Seção 6 abaixo, e baseia-se em técnicas semelhantes empregadas anteriormente em [4,8]. Esse método, chamado Escalonamento por Reversão de Arcos, exibe as duas características extremamente desejáveis de ser livre de "deadlock" e "starvation", características essas de fato essenciais à convergência da implementação distribuída de "Simulated Annealing".
2. **Recuperação do mínimo obtido durante a simulação.** "Simulated Annealing" é um método de busca estocástica, e portanto o mínimo encontrado durante a simulação pode em princípio ocorrer em qualquer ponto da simulação, não necessariamente no final. Junte-se a isso a agravante de que o mínimo encontra-se distribuído no sistema, uma vez que cada processador é responsável pelo valor de uma das variáveis. Na Seção 7, encontra-se descrita uma solução que busca o estado global do sistema [15,5] em que o mínimo ocorre. São empregadas técnicas de fluxo máximo [18] em um grafo de precedência. Devido ao método de escalonamento empregado (item 1 acima), e com o aparecimento do recente algoritmo para fluxo máximo em [12], é possível implementar a recuperação do mínimo de forma distribuída e concorrentemente com a própria simulação.

5. Problemas Adequados e Problemas Inadequado.

Um dos problemas NP-completos historicamente mais importantes é o problema de decidir se um conjunto finito de cláusulas disjuntivas com no máximo três literais por cláusula pode ser satisfeito [6,10]. Formulado de forma mais precisa, considere o conjunto X de n variáveis introduzido anteriormente. Neste caso, cada variável toma valores do conjunto comum $D = \{TRUE, FALSE\}$. Sobre essas variáveis, r cláusulas $A_1, \ldots, A_r$ são definidas. Para $1 \leq j \leq r$, A_j é da forma

$$A_j = L_{j_1} \vee L_{j_2} \vee L_{j_3},$$

isto é, A_j é a disjunção dos três literais L_{j_1}, L_{j_2} e L_{j_3}. Cada um desses literais é uma variável X_i, para algum $1 \leq i \leq n$, ou sua negação, ou o valor constante $FALSE$. A

decisão a ser feita é se a fórmula

$$A_1 \wedge \quad \wedge A_r$$

pode ser satisfeita, isto é, tornada $TRUE$ por alguma associação de valores às variáveis de X. A seguinte formulação segue o modelo introduzido anteriormente. Considere a função

$$f(x) = \sum_{Y \subseteq X} g_Y(x),$$

onde $g_Y(x) = c$, $0 \leq c \leq r$, para todo $Y \subseteq X$ tal que c das cláusulas $A_1, \ldots, A_r$ dependem exatamente das variáveis em Y e as mesmas c cláusulas são verdadeiras em x. Vê-se que u cláusulas podem ser simultaneamente satisfeitas se e somente se existe um ponto $x \in \{TRUE, FALSE\}^n$ para o qual $f(x) = u$. Portanto, uma decisão pode ser tomada encontrando-se um ponto x^* no qual $-f$ é mínima, e verificando se $f(x^*) = r$.

A formulação como um problema de otimização sem restrições não causa problemas neste caso, uma vez que $FS = D^n$ Uma outra observação é que o grafo G correspondente ao sistema para otimizar f contém um subgrafo completo para cada uma das cláusulas $A_1, \ldots, A_r$, podendo ser portanto relativamente esparso, uma vez que cada cláusula contém no máximo três literais.

Uma outra vasta classe de problemas NP-completos de interesse é constituída por problemas em grafos. Nos problemas aqui considerados, é dado um grafo $G' = (N', E')$ tal que $|N'| = n$, e com cada nó em N' é associada uma variável de X, no caso tomando valores em $D = \{0, 1\}$. Observe que N' coincide com N, o conjunto de nós do grafo G.

Um problema NP-completo típico sobre G' é o problema de se encontrar uma cobertura de vértices de tamanho mínimo ("Minimum Vertex Cover Problem"). O problema consiste em encontrar um subconjunto $I \subseteq N'$ tal que (a) todo arco $(i, j) \in E'$ possua pelo menos um de seus nós adjacentes i e j como membros de I, e (b) nenhum outro subconjunto de N' para o qual a propriedade (a) vale tenha cardinalidade inferior a $|I|$.

Para esse problema, considere a função f dada pela soma das seguintes funções:

(i) Para todo $i \in N'$:

$$g_{\{i\}}(x) = X_i;$$

(ii) Para todo $(i, j) \in E'$:

$$g_{\{i,j\}}(x) = \begin{cases} 2, & \text{se } X_i = X_j = 0; \\ 0, & \text{caso contrário.} \end{cases}$$

Pode-se notar que, em um mínimo global de f sobre D^n, existe uma correspondência biunívoca entre variáveis com valor 1 e nós em uma cobertura de vértices I de tamanho mínimo em G'. O valor de f nesse ponto ótimo é $|I|$.

Essa formulação pode ser generalizada para aplicar-se a uma classe bem mais ampla de problemas NP-completos em grafos [2]. Uma análise mais detalhada é omitida aqui, mas fica a observação de que problemas como os de encontrar um subconjunto independente máximo de N', um subconjunto dominante mínimo de N', etc., todos NP-completos [10], enquadram-se nessa formulação mais geral.

Duas observações devem ser feitas sobre a formulação apresentada acima. A primeira é que houve a necessidade de incluir uma penalidade em f (item (ii) da formulação), com o objetivo de garantir a viabilidade de todos os mínimos globais, uma vez que $FS \neq D^n$ (isto é, no caso, nem todo subconjunto de N' é uma cobertura de vértices). FS é no caso dado por

$$FS = \{x \in D^n | \text{ se } X_i = X_j = 0, \text{ então } (i,j) \notin E', i,j \in N'\}.$$

A segunda observação refere-se ao fato de que, no caso, $G = G'$. Isso significa, na prática, que o problema pode ser resolvido em um sistema distribuído representado pelo próprio grafo G'. Visto sob outro ângulo, o problema de se determinar uma cobertura mínima de vértices num grafo G' que representa um sistema distribuído pode ser resolvido no próprio sistema, sendo apenas necessário que informação seja trocada entre vizinhos. Esse fato não é verdadeiro para todos os outros problemas aos quais a formulação mais genérica em [2] se aplica. Porém, um requisito importante para uma implementação eficiente é que informação jamais tenha que ser roteada para nós em G' que se encontrem a uma distância superior a uma constante $d > 0$. Note que esses mesmos nós são vizinhos em G, que é então um grafo mais denso que G'. Como exemplo de um problema para o qual esse requisito não pode ser satisfeito, cita-se o problema de determinar o número cromático de G', $\chi(G')$ [3], conforme discutido em [2].

6. Escalonamento por Reversão de Arcos

O método utilizado para escalonar nós de G para operação é baseado na manipulação de orientações acíclicas de G [3]. Inicialmente, G é orientado por uma orientação acíclica, ω por exemplo. Segundo ω, alguns nós são diferenciados, no sentido em que todos os arcos adjacentes a eles são orientados em direção a eles. A esses nós dá-se o direito de operar

inicialmente Quando um nó termina de operar. atualizando o valor de sua variável, ele reverte o sentido de todos os arcos adjacentes a ele e transmite o novo valor de sua variável a seus vizinhos.

Em um sistema síncrono de computação distribuída, esse método de escalonamento, chamado Reversão de Arcos, pode ser visto como a evolução no tempo de orientações acíclicas de G. A cada instante do tempo síncrono, um conjunto diferenciado de nós recebe permissão para operar. Note que a aciclicidade da orientação inicial garante a aciclicidade de todas as orientações subseqüentes, e como conseqüência o método é livre de "deadlock" e "starvation". Essas duas propriedades mantêm-se, naturalmente, no caso mais realista de um sistema distribuído assíncrono.

Vale observar, finalmente, que Escalonamento por Reversão de Arcos exibe um grau de concorrência que depende intimamente da estrutura de G. Como discutido em [2], uma medida de concorrência para o método é dada por um número cujo inverso encontra-se entre os números cromático e multicromático de G [21]. É também provado [2] que o método fornece o maior grau de concorrência possível entre os mecanismos que requerem que nós em G operem alternadamente.

7. Determinando o Estado Global Ótimo

À medida que a computação distribuída para minimizar f progride, o seguinte *grafo de precedência* $H = (V, A)$ é conceitualmente gerado. H é um grafo orientado no qual existe um vértice para cada vez que cada nó em G opera sobre sua variável. Os arcos orientados em A são construídos da seguinte forma. Existe um arco orientado entre dois membros de V se e somente se (a) eles representam ações consecutivas do mesmo nó em G, ou (b) um deles representa uma ação que causou o envio de uma mensagem contendo o valor de uma variável utilizado pela ação representada pelo outro.

Cada estado global do sistema distribuído corresponde a um certo corte no grafo H. Se cada arco em A for rotulado com um número real relativo às mudanças graduais que f sofre ao longo do processo, então o estado global ótimo pode ser recuperado através de técnicas de fluxo máximo [18] aplicadas a ligeiras variações de H, utilizando os rótulos dos arcos como capacidades e/ou limites inferiores para o fluxo naqueles arcos [2].

O método de recuperação do estado global ótimo, como descrito acima, é adequado a uma implementação centralizada, levando um tempo que é em princípio $O(|V|^3)$. Como

o tamanho de V é dependente da duração da simulação, o cálculo do fluxo máximo pode ser bastante dispendioso. Felizmente, dois fatores contribuem para tornar o procedimento mais eficiente. Em primeiro lugar, como H é gerado segundo Escalonamento por Reversão de Arcos, a complexidade da computação do fluxo máximo pode na verdade ser reduzida para $O(|V|n^6)$ [2]. Em segundo lugar, o recente aparecimento do algoritmo distribuído para fluxo máximo em [12] permite que a computação do fluxo máximo em H seja feita de forma distribuída e superposta à própria simulação que gera H [2,9].

8. Conclusões

Uma observação interessante a se fazer é que o conjunto de variáveis X, quando visto como um conjunto de variáveis aleatórias, é conhecido como um *Campo Aleatório de Markov* ("Markov Random Field — MRF") com relação ao grafo G, como conseqüência da positividade de π sobre D^n, e da forma especial das probabilidades condicionais discutidas anteriormente [20]. MRFs representam uma forma de generalizar a dependência Markoviana além do contexto usual de uma única dimensão. A versão distribuída de "Simulated Annealing" apresentada neste trabalho pode ser utilizada como um mecanismo para paralelizar a atualização de MRFs, contanto que T seja mantido constante.

O trabalho está sendo estendido em várias direções, que incluem:

(a) uma avaliação experimental das possibilidades do método;

(b) implementação do método em uma máquina paralela real, possivelmente envolvendo questões de distribuição de carga, entre outras.

Referências

[1] C. R. Aragon, D. S. Johnson, L. A. McGeoch e C. Schevon, "Optimization by Simulated Annealing: an Experimental Evaluation", *Workshop on Statistical Physics in Engineering and Biology* (Abril de 1984).

[2] V. C. Barbosa, "Concurrency in Systems with Neighborhood Constraints", Tese de Doutorado, Computer Science Department, University of California, Los Angeles, CA (1986).

[3] C. Berge, *Graphs and Hypergraphs*, North-Holland, Amsterdam (1976).

[4] K. M. Chandy e J. Misra, "The Drinking Philosophers Problem", *ACM Transactions on Programming Languages and Systems* 6(4), pp. 632-646 (Outubro de 1984).

[5] K. M. Chandy e L. Lamport, "Distributed Snapshots: Determining Global States of Distributed Systems", *ACM Transactions on Computer Systems* **3**(1), pp. 63–75 (Fevereiro de 1985).

[6] S. A. Cook, "The Complexity of Theorem-Proving Procedures", pp. 151–158 in *Proceedings of the Third Annual ACM Symposium on Theory of Computing*, Shaker Heights, OH (Maio de 1971).

[7] E. Felten, S. Karlin e S. W. Otto, "The Traveling Salesman Problem on a Hypercubic, MIMD Computer", pp. 6–10 in *Proceedings of the International Conference on Parallel Processing*, Chicago, IL (Agosto de 1985).

[8] E. M. Gafni e D. P. Bertsekas, "Distributed Algorithms for Generating Loop-Free Routes in Networks with Frequently Changing Topology", *IEEE Transactions on Communications* **COM-29**(1), pp. 11–18 (Janeiro de 1981).

[9] E. M. Gafni e V. C. Barbosa, "Optimal Snapshots and the Maximum Flow in Precedence Graphs", *Proceedings of the Twenty-Fourth Annual Allerton Conference on Communication, Control, and Computing* (Outubro de 1986).

[10] M. R. Garey e D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, Freeman, San Francisco, CA (1979).

[11] S. Geman e D. Geman, "Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images", *IEEE Transactions on Pattern Analysis and Machine Intelligence* **PAMI-6**(6), pp. 721–741 (Novembro de 1984).

[12] A. V. Goldberg e R. E. Tarjan, "A New Approach to the Maximum Flow Problem", pp. 136–146 in *Proceedings of the Eighteenth Annual ACM Symposium on Theory of Computing*, Berkeley, CA (Maio de 1986).

[13] R. M. Karp, "Reducibility among Combinatorial Problems", pp. 85–103 in *Complexity of Computer Computations*, eds. R. E. Miller e J. W. Thatcher, Plenum Press, New York, NY (1972).

[14] S. Kirkpatrick, C. D. Gelatt e M. P. Vecchi, "Optimization by Simulated Annealing", *Science* **220**(4598), pp. 671–680 (Maio de 1983).

[15] L. Lamport, "Time, Clocks, and the Ordering of Events in a Distributed System", *Communications of the ACM* **21**(7), pp. 558–565 (Julho de 1978).

[16] D. G. Luenberger, *Introduction to Linear and Nonlinear Programming*, Addison-Wesley, Reading, MA (1973).

[17] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller e E. Teller, "Equations of State Calculations by Fast Computing Machines", *Journal of Chemical Physics* **21**, pp. 1087–1091 (1953).

[18] C. H. Papadimitriou e K. Steiglitz, *Combinatorial Optimization*, Prentice-Hall, Englewood Cliffs, NJ (1982).

[19] C. L. Seitz, "The Cosmic Cube", *Communications of the ACM* **28**(1), pp. 22–33 (Janeiro de 1985).

[20] F. Spitzer, "Markov Random Fields and Gibbs Ensembles", *American Mathematical Monthly* **78**, pp. 142–154 (1971).

[21] S. Stahl, "n-Tuple Colorings and Associated Graphs", *Journal of Combinatorial Theory, Series B* **20**(2), pp. 185–203 (Abril de 1976).