Correspondência inexata entre grafos

Alexandre S. Freire[1], **Carlos E. Ferreira (orientador)**[1]

[1]Departamento de Ciência da Computacão, IME-USP
Rua do Matão, 1010, CEP 05508-090 São Paulo/SP, Brasil

{afreire,cef}@ime.usp.br

Abstract. *In this master's thesis we have studied the inexact graph correspondence problem which has several variants as we describe latter, with important applications, especially in image processing and computational vision. We obtained theoretical and practical results in the work: we could show that a variant of the problem is NP-hard even when the input graphs are trees. Through formulations using integer linear programming and dynamic programming algorithms we could solve practical instances of an image processing problem with promising results. The results obtained in this study are detailed in two articles [Ferreira and Freire 2009, Freire et al. 2009].*

Resumo. *Neste trabalho de mestrado estudamos o problema de correspondência inexata entre grafos que tem diversas variantes como descrevemos a seguir, com importantes aplicações, especialmente em processamento de imagens e visão computacional. Obtivemos resultados teóricos e práticos no mestrado: pudemos mostrar que uma variante do problema é NP-difícil mesmo quando os grafos da entrada são árvores. Através de formulações usando programação linear inteira e algoritmos de programação dinâmica pudemos resolver instâncias práticas de um problema de processamento de imagens com resultados promissores. Os resultados obtidos neste mestrado estão detalhados em dois artigos [Ferreira and Freire 2009, Freire et al. 2009].*

disponível em: www.teses.usp.br/teses/disponiveis/45/45134/tde-16092008-133830

1. Introdução e motivação

Aplicações do *problema de correspondência inexata entre grafos* (PCIG) em diversas áreas têm sido apresentadas na literatura [Bunke 2000], destacando-se principalmente as áreas de processamento de imagens e visão computacional. Alguns exemplos de aplicações são: reconhecimento de símbolos gráficos [Jiang et al. 1999, Lee et al. 1990], reconhecimento de caracteres [Lu et al. 1991, Rocha and Pavlidis 1994], rastreamento de objetos em vídeos [Graciano 2007] e reconhecimento estrutural de objetos [Noma et al. 2008]. Para uma descrição informal do PCIG, considere dois grafos simples $G_I = (V_I, A_I)$ e $G_M = (V_M, A_M)$. Um mapeamento de G_I para G_M é uma função que associa cada vértice de V_I a um vértice de V_M. Tal função induz o mapeamento de cada aresta de A_I para um par de vértices de V_M da seguinte maneira: se $i \in V_I$ está associado a $k \in V_M$ e $j \in V_I$ está associado a $l \in V_M$, então a aresta ij (caso exista tal aresta) está associada ao par de vértices k, l. A cada possível associação, tanto de vértices como de arestas, é atribuído um custo. O PCIG consiste em encontrar um mapeamento de G_I para G_M, tal que a soma dos custos de suas associações seja mínima.

O PCIG tem sido muito estudado por especialistas em áreas aplicadas, o que naturalmente resulta no surgimento de diversas heurísticas [Cesar-Jr. et al. 2005, Boeres 2002, Duarte 2004] e diferentes variantes do problema. No entanto, não encontramos na literatura nenhum método praticável com garantia de otimalidade e tampouco resultados sobre a complexidade computacional do problema. Esta lacuna na literatura sobre o PCIG nos motivou a estudá-lo do ponto de vista teórico. Como o interesse pelo PCIG tem sido motivado muito mais pelo aspecto prático do que pelo aspecto teórico, decidimos fazer também experimentos computacionais com instâncias reais do problema.

Dada a gama de diferentes aplicações do PCIG, existem variantes em sua formulação, no que diz respeito às restrições do problema, dificultando o seu estudo teórico, uma vez que a presença ou ausência de determinada restrição pode modificar completamente a complexidade computacional do problema. Ao invés de apresentar diversas formulações do problema, optamos por fixar uma única formulação e, a partir dela, fazer um estudo teórico mais aprofundado. Nossa motivação inicial para estudar o PCIG veio da aplicação de reconhecimento estrutural de objetos[1], que consiste em identificar cada parte de um objeto, utilizando um modelo pré-existente da estrutura de tal objeto. Optamos por tratar apenas o *PCIG com conexidade e cobertura de arestas* (PCCA), por ser a variante do PCIG que melhor se adequa à aplicação de reconhecimento estrutural de objetos (na seção seguinte, descreveremos em mais detalhes esta aplicação, bem como as restrições de *conexidade* e *cobertura de arestas*).

A seguir resumimos nosso trabalho de mestrado [Freire 2008]. Na seção 2 explicamos em mais detalhes a aplicação que nos motivou a estudar o PCIG. Na seção 3 listamos os resultados teóricos que obtivemos. Na seção 4 exibimos alguns dos experimentos computacionais que fizemos, tanto com instância geradas automaticamente como com instâncias reais de uma aplicação. Por fim, na seção 5 resumimos as principais contribuições do trabalho e apresentamos algumas possíveis extensões.

2. Reconhecimento estrutural de objetos

Uma das aplicações do PCIG em visão computacional é o reconhecimento estrutural de objetos. São dados:

- um modelo estrutural de um objeto, que descreve as partes em que o mesmo está subdividido e as relações entre essas partes;
- uma imagem digital *segmentada* contendo um objeto, que julgamos ser do mesmo tipo do modelo. A *segmentação* de uma imagem [Graciano 2007, Saarinen 1994, Noma et al. 2008] (ver figura 1) corresponde a um pré-processamento que tem com objetivo agrupar os pontos da imagem em sub-regiões.

O objetivo é atribuir cada ponto da imagem a uma parte do modelo que acreditamos que essa parte da imagem pertença (ver figura 1).

Definimos os grafos $G_M = (V_M, A_M)$ e $G_I = (V_I, A_I)$ da seguinte maneira: V_M corresponde ao conjunto de sub-regiões do modelo; A_M corresponde às relações entre as sub-regiões do modelo (tipicamente existe uma aresta $kl \in A_M$ se e somente se as sub-regiões do modelo correspondentes aos vértices k e l possuem parte de suas bordas

[1]Foi através desta aplicação que o problema chegou ao nosso conhecimento. Agradecemos ao professor Roberto Marcondes Cesar Jr. por compartilhar conosco esta aplicação.

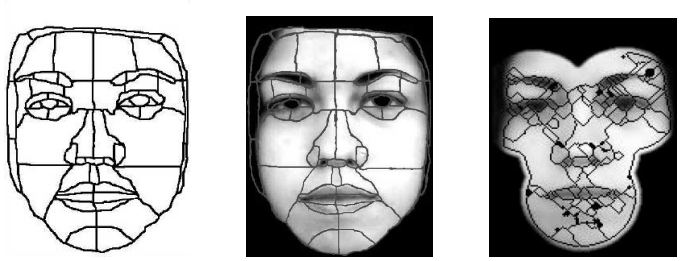

Figura 1. À esquerda, um modelo de uma face humana. À direita, uma imagem segmentada de uma face humana [Cesar-Jr. et al. 2005, Graciano 2007].

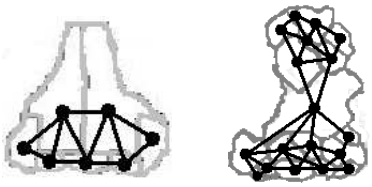

Figura 2. Os subgrafos correspondentes ao nariz contido no modelo (ao lado esquerdo) e na imagem (ao lado direito).

em comum – ver figura 2); V_I corresponde às sub-regiões obtidas na segmentação da imagem; e A_I corresponde às relações entre as sub-regiões da imagem (analogamente a A_M). Portanto, G_M é o grafo que representa o modelo e G_I é o grafo que representa a imagem. O custo da associação entre um vértice $i \in V_I$ e um vértice $k \in V_M$ é definido por um número real não negativo representando a similaridade entre as características das sub-regiões representadas pelos dois vértices. Quanto menor for este número, mais "parecidas" são as duas sub-regiões em questão. O custo da associação entre uma aresta $ij \in A_I$ e um par de vértices $k, l \in V_M$ é definido de forma similar.

Após definidos os custos das possíveis associações, deseja-se encontrar um mapeamento de G_I para G_M que minimize a soma dos custos das associações escolhidas. Na prática, nota-se que somente a minimização desta função objetivo não é suficiente para obter resultados satisfatórios. Há a necessidade de restrições adicionais para explorar as informações estruturais contidas nos grafos do modelo e da imagem. Por isso, consideramos a restrição de *conexidade* e a restrição de *cobertura de arestas*. Dizemos que um mapeamento satisfaz a restrição de *conexidade* se, para todo vértice do modelo os vértices da imagem mapeados a ele formam um subgrafo conexo do grafo da imagem. Além disso, dizemos que um mapeamento satisfaz a restrição de *cobertura de arestas* se, para toda aresta da imagem os dois vértices que são extremos desta aresta estão associados ao mesmo vértice do modelo (neste caso, dizemos que tal aresta da imagem está associada ao respectivo vértice do modelo), ou então os dois vértices que são extremos desta aresta estão associados a dois vértices do modelo para os quais existe uma aresta os ligando no modelo (neste segundo caso, dizemos que tal aresta da imagem está associada à respectiva aresta do modelo). Tais restrições surgem naturalmente na aplicação que consideramos.

3. Contribuições teóricas para o PCCA

Fizemos uma resenha sobre os resultados encontrados na literatura sobre o PCIG. Incluímos em nossa resenha formulações lineares e não-lineares do PCIG [Boeres 2002,

Duarte 2004]. Um ponto de destaque do trabalho de [Duarte 2004] foi o uso de multi-fluxo em redes na formulação linear da restrição de conexidade. Nos trabalhos [Boeres 2002, Cesar-Jr. et al. 2005, Duarte 2004] são relatados ótimos resultados práticos para a aplicação de reconhecimento estrutural de objetos. Não encontramos na literatura resultados teóricos específicos para o PCCA. Nossas contribuições foram focadas no PCCA, pois conforme dito anteriormente, esta variante do PCIG é a que melhor se adequa à aplicação de reconhecimento estrutural de objetos.

Fizemos uma redução do problema de *isomorfismo* para o PCCA. Esta redução mostra que o problema de isomorfismo é um caso particular do PCCA, o que é perceptível intuitivamente. Sabe-se que em alguns casos particulares o problema de isomorfismo pode ser resolvido em tempo polinomial [Hopcroft and Wong 1974, Luks 1982]. É evidente que o problema de isomorfismo está em NP. Porém, não se sabe se ele está em P ou se é NP-completo [Tengan 2002, Arvind and Kurur 2002]. Fizemos também uma redução do problema de *partição* [Chopra and Rao 1993] para o PCCA. Esta segunda redução mostra que o PCCA é NP-difícil, dado que o problema de partição é NP-completo. Desenvolvemos uma formulação linear inteira para o PCCA. Nossa formulação utiliza alguns conceitos de outras formulações já existentes para o PCIG, como por exemplo o tratamento da restrição de conexidade através de multi-fluxo em redes. Além disso, introduzimos uma formulação linear da restrição de cobertura de arestas e mostramos como reduzir a quantidade dessas inequações de forma a obter a mesma descrição do respectivo poliedro.

Obtivemos resultados bastante interessantes sobre o PCCA *para árvores* (caso particular do PCCA no qual os grafos de entrada são árvores). O que nos motivou a focar boa parte do nosso trabalho no PCCA *para árvores* foi o seguinte resultado que obtivemos:

> **Lema 3.1** (RESTRIÇÃO DE CONEXIDADE EM ÁRVORES)
> *Sejam $G_I = (V_I, A_I)$ e $G_M = (V_M, A_M)$ duas árvores. Um mapeamento de G_I para G_M que atende à restrição de cobertura de arestas é viável do PCCA para árvores se e somente se existe no máximo uma aresta de A_I associada a cada aresta de A_M em tal mapeamento.*

Através do lema 3.1 foi possível mostrar que uma vez fixada uma certa associação entre duas arestas, temos duas instâncias menores do problema original, sendo possível combinar as soluções ótimas desses subproblemas para compor uma solução ótima para o problema original (esta propriedade foi utilizada em nosso algoritmo de programação dinâmica para o PCCA *para árvores*). No entanto, ao fixar uma associação de uma aresta de A_I para um vértice de V_M o mesmo não ocorre. Ou seja, a propriedade da subestrutura ótima se verifica apenas em alguns casos.

Um dos resultados mais importantes de nosso trabalho foi a demonstração de que o PCCA *para árvores* é NP-difícil. Fizemos esta demonstração através da redução de um problema NP-completo chamado *emparelhamento tridimensional* [Garey and Johnson 1979]. Este resultado nos motivou a focar o restante do nosso trabalho no estudo do PCCA *para árvores*.

Desenvolvemos uma formulação linear inteira para o PCCA *para árvores* que, em virtude do lema 3.1, é muito mais compacta, quando comparada com a formulação que fizemos para o caso geral do PCCA. Pois, quando os grafos da entrada são árvores a

restrição de conexidade é tratada de forma muito mais simples, sem a necessidade do uso de multi-fluxo em redes.

Utilizando o lema 3.1 chegamos a um algoritmo (com garantia de otimalidade) de programação dinâmica para o PCCA *para árvores* cujo tempo de execução é exponencial no grau máximo dos vértices de V_M. Ou seja, se o grau máximo dos vértices de V_M for limitado por uma constante, o algoritmo torna-se polinomial. Além disso, na prática este algoritmo mostrou-se consideravelmente rápido. Os aspectos principais que fizeram com que nosso algoritmo atingisse uma boa performance foram o uso de programação dinâmica e a utilização de programação linear inteira em um determinado passo do algoritmo.

Nosso último resultado teórico foi o desenvolvimento de um algoritmo polinomial para um caso particular do PCCA *para árvores*. Neste caso particular, exige-se que G_M seja uma árvore qualquer e que G_I seja uma *estrela estendida* (ou *spider graph*). Informalmente, uma estrela estendida é uma árvore que consiste da união de caminhos que têm um único vértice em comum. Para chegar a este resultado fizemos basicamente uma redução deste caso do PCCA para o problema de *emparelhamento perfeito de custo mínimo* [Kuhn 1955, L. Lovász 1986], para o qual existem algoritmos polinomiais.

Vale ressaltar que escrevemos um artigo [Ferreira and Freire 2009] (já submetido) reunindo os principais resultados teóricos que obtivemos em nosso trabalho de mestrado.

4. Contribuições para a aplicação de Processamento de Imagens

O desenvolvimento de um algoritmo com garantia de otimalidade para o PCCA não nos traz garantias de que seu uso na aplicação de reconhecimento estrutural de objetos traga bons resultados. Isso se deve ao fato de que dados o modelo e a imagem segmentada, precisamos gerar as árvores correspondentes (no caso do PCCA *para árvores*) para executar nosso algoritmo. Uma boa escolha destas árvores pode ser fundamental para a eficácia de nossa abordagem. Uma vantagem de tratar uma aplicação como essa é que é possível avaliar a qualidade da solução obtida, uma vez que é fácil verificar se o resultado obtido é o esperado (as partes da imagem são associadas corretamente às partes do modelo). Por isso, nosso primeiro objetivo, nesta etapa do trabalho, foi testar a "eficácia" do nosso algoritmo para o PCCA *para árvores*. Isto é, dada uma instância para a qual espera-se um determinado resultado, nosso objetivo foi testar o quão perto deste resultado está a solução produzida por nosso algoritmo. Para atingir este objetivo, geramos (de forma automatizada) instâncias artificiais do problema, para as quais espera-se um determinado resultado. Cada instância é composta por um par de árvores (G_I e G_M). Utilizamos o valor de $\alpha = |\mathcal{M}^* \cap \mathcal{M}|$ como medida de "eficácia" do algoritmo (quanto maior for o valor de α, mais próximo do esperado está o mapeamento encontrado pelo algoritmo), sendo que $\mathcal{M}$ é o mapeamento produzido por nosso algoritmo e $\mathcal{M}^*$ é o mapeamento esperado. Para cada instância, fizemos diversos testes, aumentando cada vez mais o nível de "ruído" em G_I. Este ruído consiste em alterações feitas nas características dos vértices e das arestas de G_I, de forma a simular o que ocorre na prática. Ou seja, à medida que aumentamos o nível de ruído em G_I, as características encontradas em G_I tornam-se cada vez mais diferentes das que foram modeladas em G_M, o que dificulta a escolha das associações "corretas". Nossos testes mostram que mesmo com um nível relativamente alto de ruído, os mapeamentos que obtivemos estão muito próximos dos esperados.

Avaliamos também o desempenho empírico do nosso algoritmo para o PCCA *para*

árvores. Geramos instâncias de forma automatizada, variando os parâmetros $|V_I|$, $|V_M|$ e $\beta = \max_{k \in V_M} g(k)$, sendo que $g(k)$ corresponde ao grau do vértice k. Nossa análise de pior caso mostra que o algoritmo é exponencial em β. Na prática, verificamos que o algoritmo é praticável mesmo quando β é razoavelmente grande. Resolvemos instâncias com $100 \leq |V_I| \leq 800$, $100 \leq |V_M| \leq 800$ e $\beta \leq 20$ em poucos minutos, e instâncias com $|V_I| = 500$, $|V_M| = 200$ e $\beta \geq 80$ em um pouco mais de uma hora. Estes testes foram feitos em máquina com 512MB de memória RAM e processador Intel Pentium 1.73 GHz 795 MHz. O algoritmo é mais sensível ao aumento do parâmetro $|V_I|$ do que ao aumento do parâmetro $|V_M|$, o que nossa análise da complexidade do algoritmo também reflete.

A última etapa de nossos testes teve como objetivo utilizar o algoritmo para o PCCA *para árvores* como sub-rotina de uma heurística para o problema de reconhecimento estrutural de objetos. O funcionamento desta heurística está fundamentado na ideia de obter uma árvore geradora para cada um dos grafos da entrada, e encontrar um mapeamento ótimo entre estas árvores. Nossa estratégia para obter tais árvores geradoras foi considerar três tipos de árvores: árvore geradora mínima, árvore geradora máxima e árvore geradora obtida aleatoriamente. Testamos nossa heurística com instâncias reais da aplicação de reconhecimento estrutural de objetos. Os resultados obtidos nestes testes foram bastante promissores, considerando que não temos familiaridade suficiente para ajustar detalhes de implementação que dependem de características próprias da aplicação. Na figura 3, mostramos um dos testes realizados com nossa heurística.

Figura 3. À esquerda uma imagem de uma face humana. Ao centro uma representação do resultado esperado. À direita o resultado obtido.

Vale salientar que estamos finalizando um artigo [Freire et al. 2009] no qual reunimos os principais resultados experimentais que obtivemos em nosso trabalho de mestrado.

5. Conclusão e trabalhos futuros

As principais contribuições do nosso trabalho de mestrado são:

- *as reduções de problemas clássicos (isomorfismo e partição) para o PCCA*: com estas reduções pudemos mostrar que o problema de isomorfismo é um caso particular do PCCA e que o caso geral do PCCA é NP-difícil;
- *a demonstração de que o PCCA para árvores é* NP-difícil: através desta redução mostramos que um caso bastante restrito do problema (quando os grafos da entrada são árvores) é NP-difícil, tornando este resultado um dos mais importantes do nosso mestrado;
- *uma formulação linear inteira para o PCCA*: nossa formulação para o PCCA utiliza ideias já existentes para a representação da restrição de conexidade. Introduzimos inequações lineares para representar a restrição de cobertura de arestas;
- *uma formulação linear inteira (mais compacta) para o PCCA para árvores*: através do lema 3.1 derivamos uma representação bastante simples da restrição de

conexidade (que no caso geral é tratada através do uso de multi-fluxo em redes) quando os grafos da entrada são árvores;

- *um algoritmo exato para o PCCA para árvores*: neste algoritmo utilizamos programação dinâmica e programação linear inteira. Seu consumo de tempo é polinomial se o grau máximo dos vértices de V_M for limitado por uma constante. Na prática, mesmo quando o grau máximo dos vértices de V_M é razoavelmente grande (maior ou igual a 80, nos nossos testes), nosso algoritmo ainda atinge um bom desempenho;
- *um algoritmo (exato) polinomial para um caso especial do PCCA para árvores*: mostramos que quando G_I é uma estrela estendida (ou *spider graph*) e G_M é uma árvore arbitrária é possível reduzir o PCCA ao problema de emparelhamento perfeito de custo mínimo, para o qual existem algoritmos eficientes;
- *testes computacionais com nosso algoritmo para o PCCA para árvores*: mostramos através de experimentos computacionais que nosso algoritmo para o PCCA *para árvores* é praticável para instâncias relativamente grandes. Além disso, nossos testes mostraram que nosso algoritmo produz resultados satisfatórios, quando comparado a resultados esperados, mesmo quando há grande discrepância entre os grafos de entrada.
- *uma nova heurística para o problema de reconhecimento estrutural de objetos*: utilizamos nosso algoritmo para o PCCA *para árvores* como base de uma heurística para o problema de reconhecimento estrutural de objetos. Testamos esta heurística com instâncias práticas desta aplicação. Os resultados obtidos nos testes são bastante promissores;

Escrevemos um artigo [Ferreira and Freire 2009] (já submetido) que reúne os principais resultados teóricos do nosso mestrado. Além disso, estamos finalizando um artigo [Freire et al. 2009] com os resultados práticos que obtivemos no mestrado.

Como extensão de nosso trabalho sugerimos: a utilização de nosso algoritmo para o PCCA *para árvores* em conjunto com outras estratégias para extrair árvores geradoras de G_I e G_M, de forma a produzir resultados melhores na aplicação de reconhecimento estrutural de objetos; a utilização de nossas formulações para o PCCA e para o PCCA *para árvores* como ponto de partida para fazer um estudo poliédrico do problema; o desenvolvimento de algoritmos de aproximação para o PCCA (não encontramos nada na literatura com esta abordagem).

Referências

Arvind, V. and Kurur, P. P. (2002). Graph isomorphism is in spp. *Proceedings of the 43rd Symposium on Foundations of Computer Science*, pages 743–750.

Boeres, M. C. S. (2002). *Heurísticas para reconhecimento de cenas por correspondência de grafos*. PhD thesis, Universidade Federal do Rio de Janeiro.

Bunke, H. (2000). Graph matching: Theoretical foundations, algorithms, and applications. In *Vision Interface*, pages 82–88.

Cesar-Jr., R. M., Bengoetxea, E., Bloch, I., and Larrañaga, P. (2005). Inexact graph matching for model-based recognition: Evaluation and comparison of optimization algorithms. *Pattern Recognition*, 38:2099–2113.

Chopra, S. and Rao, M. R. (1993). The partition problem. *Mathematical Programming*, 59:87–115.

Duarte, A. R. (2004). Novas heurísticas e uma abordagem por programação inteira para um problema de correspondência inexata de grafos. Master's thesis, Pontifícia Universidade Católica do Rio de Janeiro.

Ferreira, C. E. and Freire, A. S. (2009). The graph mapping problem: NP-hardness proof, formulations and algorithms. Submetido.

Freire, A. S. (2008). Correspondência inexata entre grafos. Master's thesis, Instituto de Matemática e Estatística da Universidade de São Paulo.

Freire, A. S., Noma, A., Ferreira, C. E., and Cesar-Jr, R. M. (2009). Inexact graph matching: a heuristic based on spanning trees. Em preparação.

Garey, M. R. and Johnson, D. S. (1979). *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman.

Graciano, A. B. V. (2007). Rastreamento de objetos baseado em reconhecimento estrutural de padrões. Master's thesis, Universidade de São Paulo - Instituto de Matemática e Estatística.

Hopcroft, J. E. and Wong, J. K. (1974). Linear time algorithm for isomorphism of planar graphs (preliminary report). In *STOC '74: Proceedings of the sixth annual ACM symposium on Theory of computing*, pages 172–184, New York, NY, USA. ACM Press.

Jiang, X., Münger, A., and Bunke, H. (1999). Synthesis of representative graphical symbols by generalized median graph computation. In *3rd IAPR Workshop on Graphics Recognition*.

Kuhn, H. W. (1955). The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, pages 83–97.

L. Lovász, M. D. P. (1986). Matching theory. *Elsevier Science*.

Lee, S. W., Kim, J. H., and Groen, F. C. A. (1990). Translation, rotation and scale invariant recognition of hand drawn symbols in schematic diagrams. *Pattern Recognition and Artificial Intelligence*.

Lu, S. W., Ren, Y., and Suen, C. Y. (1991). Hierarchical attributed graph representation and recognition of handwritten chinese characters. *Pattern Recognition*.

Luks, E. M. (1982). Isomorphism of graphs of bounded valence can be tested in polynomial time. *Journal of Computer and System Sciences*, pages 42–65.

Noma, A., Graciano, A. B. V., Consularo, L. A., Cesar-Jr, R. M., and Bloch, I. (2008). A new algorithm for interactive structural image segmentation. Technical report, IME-USP. arXiv:0805.1854v2 [cs.CV].

Rocha, J. and Pavlidis, T. (1994). A shape analysis model with applications to a character recognition system. *IEEE Trans. PAMI*.

Saarinen, K. (1994). Color image segmentation by a watershed algorithm and region adjacency graph processing. *Image Processing, IEEE International Conference*.

Tengan, E. (2002). Automorfismo de grafos. Master's thesis, Universidade de São Paulo – Instituto de Matemática e Estatística.