Um Enfoque Analógico para a Solução do Problema do Caixeiro Viajante através do Método Elástico

Maria Claudia Silva Boeres

TESE SUBMETIDA AO CORPO DOCENTE DA COORDENAÇÃO DOS PROGRAMAS DE PÓS-GRADUAÇÃO DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO.

Aprovada por:

Prof. Luis Alfredo Vidal de Carvalho, D. Sc.
(presidente)

Prof. Valmir Carneiro Barbosa, Ph. D.

Profa. Suzana Scheimberg de Makler, D. Sc.

Profa. Nair Maria Maia de Abreu, D. Sc.

RIO DE JANEIRO, RJ – BRASIL
FEVEREIRO DE 1992

BOERES, MARIA CLAUDIA SILVA

Um Enfoque Analógico para a Solução do Problema do Caixeiro Viajante através do Método Elástico [Rio de Janeiro] 1992

VI, 89 p., 29.7 cm, (COPPE/UFRJ, M. Sc., ENGENHARIA DE SISTEMAS E COMPUTACÃO, 1992)

TESE – Universidade Federal do Rio de Janeiro, COPPE

1 – *NP*-completo 2 – Algoritmo Elástico 3 – Filtro 4 – Algoritmo LK

I. COPPE/UFRJ II. Título(Série).

A meus pais
e a minha irmã, Cristina

Agradecimentos

Agradeço ao Luís Alfredo e ao Valmir, aos quais admiro principalmente pelo dinamismo, objetividade e seriedade profissional. O pronto atendimento de ambos em muitos momentos de dificuldade foi de grande incentivo para este trabalho.

Ao Luís Alfredo em particular, agradeço pelo apoio, compreensão e amizade demonstrados em todos os momentos.

A Nair Abreu, pela constante atenção, confiança e principalmente sua amizade.

A Dóris Aragon, a quem muito admiro pelo espírito de pesquisa, agradeço profundamente a sua confiança, solidariedade e amizade.

A Sueli Mendes, pelo apoio e compreensão demonstrados no início desse trabalho.

Ao Luiz Adauto, que muito me ajudou, mesmo de longe, com sua experiência e seus conselhos. Agradeço em especial a atenção e força, fundamentais ao amadurecimento deste trabalho. Sua ajuda na correção do texto foi de grande importância.

Ao Delfim, muito presente em minha iniciação na "arte" de implementar. Agradeço seu carinho e amizade, demonstrados em todos os momentos.

Ao Maurício Raposo, por ter sido meu grande amigo, companheiro e "psicólogo". Suas dicas e sugestões foram fundamentais para este trabalho. Agradeço também as sugestões na correção do texto.

Ao Ricardo Bianchini, pelas rotinas gráficas usadas no meu trabalho quando implementado na SUN.

As grandes amigas, Lúcia, Inah, Rose, Jane, Cecília e Claudia Linhares, por momentos muito divertidos, por experiências significativas, pelo carinho e principalmente por suas amizades.

A todos os amigos da COPPE, pela simpatia com que sempre me acolheram.

A "galera" do laboratório, em especial o Eliseu, pelos vários momentos divertidos e por vários "galhos" resolvidos.

A Denise, Cláudia e Ana Paula, por estarem sempre presentes e prontas a ajudar em qualquer problema, agradeço por suas amizades.

Ao ILTC, pela possibilidade de livre acesso e total confiança no uso de qualquer equipamento, principalmente as workstations, ajudando a acelerar o término desse trabalho.

A todos os amigos do ILTC, em especial Emília e Magali, que participaram ativamente da fase final deste trabalho. O apoio de todos foi muito importante.

A pessoas cujos agradecimentos por suas participações estão completamente implícitas nas nossas relações:

Meus pais, muito presentes e solidários. Agradeço a minha mãe pela correção do texto e a meu pai pela confecção dos desenhos.

Meu irmão, Deo e a Sônia, pela força e confiança.

Minha avó, que sempre torceu por mim.

Minha irmã, simplesmente por tudo.

A Deus, por esta oportunidade e por estes amigos.

Resumo da Tese apresentada à COPPE como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M. Sc.)

Um Enfoque Analógico para a Solução do Problema do Caixeiro Viajante através do Método Elástico

Maria Claudia Silva Boeres

Fevereiro de 1992

Orientadores: Luis Alfredo Vidal de Carvalho
Valmir Carneiro Barbosa.
Programa: Engenharia de Sistemas e Computação.

É apresentada neste trabalho, uma versão modificada do Algoritmo Elástico denominada por *Filtro*, aplicada sobre um problema de otimização combinatória muito famoso, o problema do Caixeiro Viajante. Com o objetivo de reduzir o tempo de processamento do Algoritmo Elástico, desenvolveu-se uma modificação do mesmo, no sentido de minimizar o número de cálculos de distâncias necessários em cada iteração do algoritmo. A aplicação do método em vários conjuntos de cidades gerou resultados que foram comparados com aqueles gerados por outro algoritmo, inplementado também neste trabalho, o algoritmo LK. Foi observado que tanto o Elástico quanto o Filtro geraram resultados piores que o LK, em termos de custo de *tour*, porém num tempo de processamento substancialmente melhor. Os testes foram desenvolvidos nas estações de trabalho SUN, utilizando a linguagem *C* como linguagem de programação.

Abstract of Thesis presented to COPPE as partial fulfillment of the requirements for the degree of Master of Science (M. Sc.)

An Analogue Approach to the Travelling Salesman Problem using an Elastic Net Method

Maria Claudia Silva Boeres

February, 1992

Thesis Supervisors: Luis Alfredo Vidal de Carvalho

Valmir C. Barbosa.

Department: Programa de Engenharia de Sistemas e Computação.

It is presented in this work, a modified version of Elastic Net denoted by *the Filter*. The method is applied to a classical problem in the field of combinatorial optimization, the Travelling Salesman Problem. The main computacional cost is in calculating the distances at each iteration. Concerned with the redution of the computational time required, it was proposed in this work, a minimization of this calculation. The application of the method on random sets of cities, gives worse quality solutions, but much better computational times than another algorithm developed here, the *LK*. The tests have been performed on a workstation (SUN), utilizing the C language as programming tool.

Índice

Lista de Figuras

Capítulo I

Introdução

Um problema de Otimização Combinatória pode ser caracterizado pela minimização ou maximização de uma função f de variáveis independentes $x_1, \ldots, x_n$ definidas sobre um domínio D enumerável, quase sempre de cardinalidade finita. Usualmente denominada função-objetivo ou função-custo, f pode ser definida por

$$f : D^n \longrightarrow \mathcal{R}.$$

A busca do mínimo (ou máximo) de f é realizada sobre um subconjunto de D^n, determinado por um conjunto de restrições ou critérios C, especificado para o problema. Esse subconjunto é denominado conjunto dos Pontos Viáveis.

Como o conjunto de Pontos Viáveis pode assumir uma quantidade de valores muito grande, a obtenção de uma solução exata para o problema através da especificação de um algoritmo para resolvê-lo, pode exigir um *tempo* de computação inviável. A medida desse *tempo* determina a *complexidade* do algoritmo. Quando a *complexidade* de um algoritmo pode ser limitada por uma função polinomial no tamanho do problema, diz-se que o algoritmo é *eficiente*. Caso contrário, o algoritmo é dito *ineficiente* ou *exponencial*.

Um problema é dito NP-completo quando não se conhece nenhum algoritmo *eficiente* capaz de resolvê-lo. Muitos problemas de Otimização Combinatória são NP-completos. Em função da importância desses problemas, surgiu uma série de heurísticas ou técnicas aproximativas para resolvê-los num tempo de computação viável, porém sem a garantia de obtenção da solução ótima, como por

exemplo as estratégias de divisão e conquista e métodos iterativos de melhoria.

A estratégia de divisão e conquista consiste na divisão do problema em subproblemas, resolvendo-os localmente. A combinação adequada dos resultados constitui a solução final para o problema. É vital que se tenha cuidado ao escolher o problema a ser resolvido por esse tipo de estratégia. A eficiência de sua aplicação depende tanto de sua correta divisão, quanto da adequada junção dos resultados, para que os ganhos obtidos nas soluções de cada subproblema, não sejam perdidos na formação da solução final.

Os métodos iterativos de melhoria traduzem-se em uma busca no espaço de soluções do problema. Uma configuração ou estado inicial (representando uma solução qualquer no espaço de soluções) é escolhida. Esse estado sofre uma transformação — especificada pela heurística — gerando um novo estado que deve ser de melhor custo. Esse processo é repetido até que não se consiga mais realizar melhorias. Esse tipo de estratégia caracteriza-se por apresentar uma solução que geralmente representa um mínimo local (e não global), não sendo muitas vezes satisfatória. Para evitar a eventual determinação de uma solução local de alto custo, é necessário que o processo seja reiniciado a partir de diversos pontos do espaço de soluções, escolhendo-se assim, o menor dos mínimos locais alcançados.

Nesse sentido, o algoritmo conhecido como "*Simulated Annealing*" [16] apresenta uma filosofia diferente, pois ao contrário das heurísticas descritas acima, permite que ocasionalmente sejam aceitas configurações que representam soluções de pior custo. Esse algoritmo, análogo ao processo físico "*Annealing*", é uma variação do algoritmo de Metrópolis [3] [5] [16]. O decréscimo lento e gradual de um parâmetro T, que representa a temperatura inerente ao processo, simula o resfriamento do sistema. A cada valor de T, são aceitas configurações, de acordo com uma probabilidade guiada pela distribuição de Boltzmann, que eventualmente correspondem a aumentos na função de custo. Esse fato possibilita a "fuga" de mínimos locais em busca de outros de menor valor [3] [5].

As técnicas citadas acima são caracterizadas como métodos discretos seqüenciais, pois operam seqüencialmente trocando ou adicionando elementos em

alguma configuração inicial dada. Diversas das técnicas empregadas em problemas de Otimização Combinatória fazem uso desse tipo de abordagem.

O problema que estamos interessados aqui, é o problema do Caixeiro Viajante. Aparentemente simples, é um problema de Otimização Combinatória muito conhecido por ser de difícil solução. Ele pertence à classe de problemas NP-completo e é famoso pois, mesmo sendo de difícil solução, ele pode ser enunciado de maneira indiscutivelmente simples e por isso mesmo, inúmeras propostas e técnicas distintas têm sido empregadas para solucioná-lo. Assim, ele é bastante utilizado para testes de eficiência de diversos algoritmos aproximativos, como em alguns dos métodos descritos acima [16] [19]. Dentre as inúmeras versões existentes, a que adotaremos neste trabalho é a seguinte:

- Dado um conjunto de N cidades distribuídas num plano Euclideano, o problema consiste em se determinar o menor caminho fechado (*tour*) que visite todas as cidades somente uma vez.

Menos comuns são os métodos analógicos de solução. Como exemplo desse tipo de abordagem, foi proposta por Hopfield & Tank [14] uma rede neuronal constituída de neurônios de atividade contínua para solucionar o problema do Caixeiro Viajante. A função-objetivo do problema pode ser definida em termos da minimização da função de energia associada à rede.

Durbin & Willshaw [7] propuseram uma nova técnica, também analógica, para resolver o problema do Caixeiro Viajante. Nesse modelo, denominado Algoritmo Elástico, um anel é gradativamente alongado em direção às cidades que se encontram distribuídas num plano Euclideano. No final do processamento, o anel passa por todas as cidades, formando uma *tour*. O algoritmo possui características essencialmente geométricas, já que a deformação do anel depende diretamente da atualização das coordenadas de pontos situados sobre ele, realizada em função de suas distâncias às cidades no plano. O trabalho desta tese será baseado nesse algoritmo, fazendo-se não só um estudo de suas características, como também uma proposta de modificação e uma análise de resultados experimentais.

I.1 Uma visão geral

Esse trabalho está dividido em mais cinco capítulos.

No capítulo II, a descrição do método proposto para resolver o Caixeiro Viajante — o Algoritmo Elástico — é realizada explicando-se em detalhes o funcionamento do algoritmo, suas características, mostrando-se a origem de sua motivação e finalmente um levantamento da bibliografia relacionada.

No capítulo III descrevem-se detalhadamente as modificações feitas no algoritmo inicial, as quais foram propostas neste trabalho no sentido de reduzir o número de cálculos necessários para a evolução do algoritmo, e assim, conseqüentemente, reduzir seu tempo de processamento.

No capítulo IV é apresentado um algoritmo proposto por Lin & Kernighan [19] em 1973, considerado um método clássico na solução do problema do Caixeiro Viajante, pois até hoje seus resultados são bastante razoáveis quando comparados aos resultados apresentados por outros métodos. Esse algoritmo constitui-se num exemplo típico de um método iterativo de melhoria. Os seus resultados serão, neste trabalho, comparados com os resultados obtidos com o Algoritmo Elástico.

No Capítulo V serão apresentados, analisados e avaliados os testes realizados com o algoritmo Elástico, na sua versão original, na versão modificada, proposta neste trabalho e com o algoritmo de Lin & Kernighan.

Finalmente, no Capítulo VI, será realizada uma análise de todo o trabalho através da apresentação das conclusões.

Capítulo II

O Algoritmo Elástico

II.1 Introdução

Para atingir o objetivo principal de muitos dos problemas de Otimização Combinatória — o de minimizar uma função objetivo obedecendo a um conjunto de restrições — foram propostos diversos algoritmos que consistem em métodos heurísticos, ou melhor, métodos que produzem soluções aproximadas, porém, muitas vezes bastante razoáveis para o problema.

Muitos desses métodos consistem basicamente na minimização de uma função, muitas vezes denominada de *energia*, através do uso do *método do gradiente*. Alguns deles minimizam a *energia* através da constante atualização de parâmetros escolhidos num conjunto discreto, como por exemplo, a rede neuronal desenvolvida por Hopfield ([13], 1984) e outros através de parâmetros escolhidos num domínio contínuo, como o método elástico desenvolvido por Durbin e Willshaw ([7], 1987). Sendo esse último a base de todo o trabalho desenvolvido nesta tese, será dada ênfase nesse capítulo, à explicação minunciosa do método, da sua motivação e à enumeração de alguns dos trabalhos relacionados com esse método.

II.2 A motivação do método

Muitas partes do sistema nervoso de vertebrados se interconectam topograficamente, ou seja, as características ou peculiaridades presentes em uma determinada região,

quando mapeada ou conectada a outra, são preservadas. Por exemplo, axônios de células que compõem estruturas sensoriais projetam-se topograficamente em estruturas centrais do cérebro.

Mais rigorosamente, um mapeamento entre dois espaços é dito topográfico quando as características e relações de vizinhança entre os elementos do espaço de saída são preservadas no espaço de chegada. Intuitivamente, é como se fosse estabelecida uma correspondência entre os dois espaços. Sendo assim, é perfeitamente viável a realização de mapeamentos entre espaços de diferentes dimensões, pois o importante é que apenas propriedades de vizinhança (i.e., topográficas) sejam preservadas.

Um exemplo muito estudado de mapeamento topográfico entre estruturas biológicas, é aquele entre a retina e regiões do cérebro que respondem a estímulos visuais. Em vertebrados superiores, por exemplo, existe um mapa da retina na superfície do córtex estriado (Talbot & Marshall, 1941). Em vertebrados inferiores, nervos óticos são mapeados diretamente no tecto ótico (Gaze, 1958). Com o desenvolvimento e estudo da observação dessas características, surgiu uma série de teorias e modelos para o estabelecimento de mapeamentos topográficos durante o desenvolvimento ou regeneração do sistema nervoso.

Nesta linha, Malsburg e Willshaw [20] [29] propuseram um modelo formal para o estabelecimento de projeções organizadas topograficamente entre uma superfície pré-sináptica e uma superfície pós-sináptica de células nervosas . As relações de vizinhança entre os elementos da superfície pré-sináptica são expressas através de um conjunto de marcadores químicos. Adota-se a hipótese de que, devido a algum processo de difusão, cada célula se encontra associada a um conjunto de marcadores químicos único que codifica sua posição na superfície, em relação às células vizinhas. A partir dessas células são desenvolvidas aleatoriamente conexões com a superfície pós-sináptica. Tais conexões conduzem os marcadores químicos provenientes das células pré-sinápticas para a superfície pós-sináptica, onde a princípio, são inexistentes. Ao chegarem à superfície pós-sináptica, os marcadores químicos se difundem, concentrando-se, como na superfície pré-sináptica, em pequenos núcleos associados às células pós-sinápticas.

Segundo o modelo, esses marcadores constituem os principais responsáveis pela orientação das conexões nervosas no momento da ligação entre as duas superfícies. Cada conexão é intensificada ou não, de acordo com a similaridade dos marcadores conduzidos por ela e aqueles já existentes no ponto de ligação na superfície pós-sináptica. Conexões que conduzem tipos de marcadores químicos completamente diferentes daqueles já existentes na região de sua terminação tendem a desaparecer, enquanto aqueles que conduzem tipos de marcadores bastante semelhantes àqueles existentes na superfície se intensificam. Ao longo do processo de mapeamento, as conexões se auto-organizam associando grupos de células pré-sinápticas a grupos de células pós-sinápticas, de acordo com a influência dos marcadores químicos. Dois tipos de mecanismo distintos estão em operação:

- um *mecanismo local*, referente a cada superfície isoladamente, onde são expressas relações geométricas entre células,

- um *mecanismo global*, referente ao mapeamento como um todo, onde se é especificada uma maneira de se traduzir na superfície de células pós-sinápticas as características ou similaridades das relações entre as células da superfície pré-sináptica.

O que nos interessa, neste momento, não é descrever minunciosamente os detalhes biológicos do modelo citado acima e sim, explorar a idéia do mecanismo de auto-organização usado para desenvolver os mapeamentos, uma vez que é ele a principal motivação para o desenvolvimento do Algoritmo Elástico. Exibiremos então, uma hipotética situação de comércio entre a Índia e Inglaterra, descrita em [24], que explica intuitivamente o mecanismo de auto-organização utilizado no modelo descrito sucintamente acima.

O fato de os ingleses serem conhecidos como grandes apreciadores de chá incentiva a comercialização desse produto com a Índia, por lá se encontrarem várias plantações de ervas que constituem tipos básicos de chá. São formadas, de acordo com o processo a ser descrito, várias rotas de comércio entre esses dois países.

É interessante observar que a formação dessas rotas é feita segundo um mecanismo de auto-organização.

Espalhadas por toda a Índia, existem plantações de ervas que constituem diferentes tipos de chá. Cada comerciante elabora misturas diversas com ervas provenientes de plantações próximas das cidades onde moram. A mistura de cada comerciante é caracterizada por essas ervas. A quantidade de cada erva é proporcional à distância entre a plantação e a cidade. Quanto mais próxima for a plantação, maior quantidade daquele tipo de erva conterá a mistura. Desta forma, como existem comerciantes de chá em vários pontos diferentes da Índia, existirão também vários tipos de mistura diferentes. Cada comerciante pode ser unicamente caracterizado pela mistura de chá de que ele dispõe. Além disso, dois comerciantes que sejam vizinhos possuem misturas similares, pois muitas das plantações próximas às suas cidades são comuns aos dois.

Ao viajar para a Inglaterra, cada comerciante leva consigo, como mercadoria, a sua mistura de ervas. No entanto, a quantidade total que cada comerciante pode levar em cada viagem é limitada.

Por sua vez, consumidores de chá espalhados por toda a Inglaterra, compram ingredientes para suas misturas diretamente dos comerciantes vindos da Índia. Além disso, diversificam suas opções comprando, ou trocando ervas que sejam do seu gosto com outros consumidores de chá que morem próximos a eles. Desta forma, consumidores vizinhos possuirão misturas parecidas. A barganha de ervas acarretará a formação de misturas com praticamente os mesmos tipos de ervas, porém em quantidades diferentes.

Ao partirem de suas cidades, os comerciantes têm liberdade de viajar para os pontos mais diversos. Dois comerciantes vizinhos podem, por exemplo, viajar para pontos bem distantes na Inglaterra, como para regiões no sul e no norte, respectivamente.

A figura II.1 mostra uma configuração inicial para o sistema de comércio que está sendo descrito. A seguir, tentaremos explicar como esse sistema se desenvolve. É interessante notar a mudança das rotas durante a evolução do

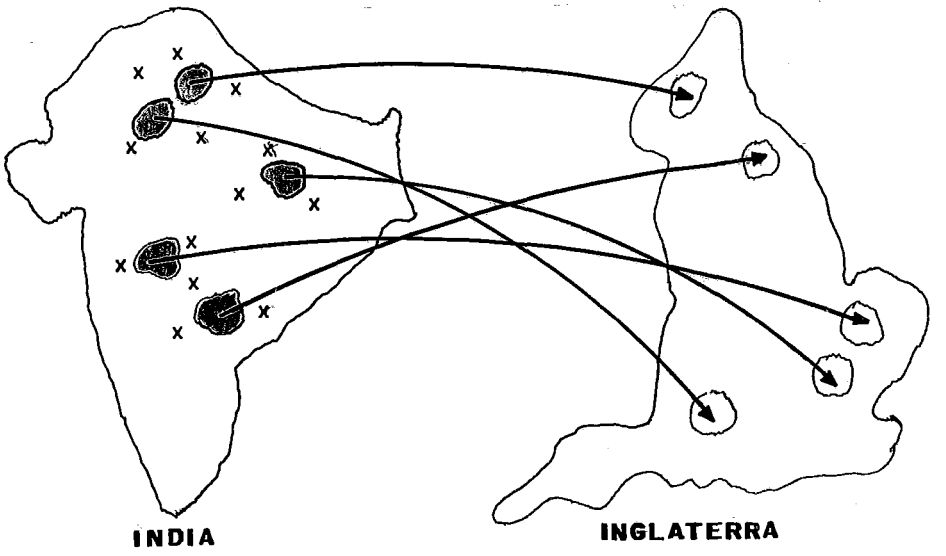


Figura II.1: Configuração inicial do sistema de comércio

comércio.

Cada comerciante possui um caderno de anotações com nomes de prováveis fregueses e a quantidade de ervas referente ao consumo de cada um deles. O comerciante escolhe a região para a qual ele pretende viajar, e lá mostra o seu produto. Dependendo do sucesso da venda, ele pode ou não modificar sua rota de viagem, tendendo para lugares de maior aceitação de sua mistura. De acordo com a similaridade entre a sua mistura e a do consumidor, a quantidade de cada erva é ajustada — nas próximas cotas, a quantidade das ervas que estiverem também presentes nas misturas de seus fregueses será aumentada.

O aumento da quantidade de ervas em favor de consumidores que possuírem misturas similares à do comerciante causará a diminuição, ou praticamente perda total, de ervas de interesse daqueles que não a possuírem, já que a quantidade total da mistura levada em cada viagem é limitada. Em função dessa similaridade, nas próximas viagens, cada comerciante terá como destino, regiões onde moram consumidores com misturas bem parecidas com a dele. Além disso, o comerciante aumentará sua venda estabelecendo contatos com consumidores vizinhos, já que estes possuem interesse nos mesmos tipos de ervas. Para isso, é importante que ele traga algumas amostras extras para esses novos prováveis fregueses. É neste

ponto que começa o processo de organização em grupos. Uma vez que vários consumidores possuem diversas ervas semelhantes em suas misturas, a tendência é a aproximação de cada um deles em torno do comerciante que tiver a mistura mais adequada a seus gostos, organizando-se para cada comerciante um grupo específico de fregueses. Desta forma, cada comerciante se tornará específico de uma determinada região.

A organização em grupos também acontece entre comerciantes na Índia, pois como a aquisição de ervas se dá em função da distância das plantações das mesmas, um comerciante que tem acesso a certos tipos de ervas irá compartilhá-las com outros comerciantes que estejam próximos a ele. É bom ressaltar também que, se um comerciante obtêm lucro num determinado ponto da Inglaterra, outro, vizinho a ele na Índia, viajará para uma região bem próxima, pois sua mistura também terá uma boa aceitação naquela região, já que os dois possuem misturas bem similares.

O sistema de comércio descrito acima pode ser entendido como um processo de auto-reforço. Cada comerciante atrai para si, além daqueles que já são fregueses, outros consumidores de chá que são vizinhos e que possuem gostos similares aos deles. Como a cota de chá para cada comerciante é limitada, alguns consumidores que até podem ter mantido contato inicialmente, são descartados da lista de fregueses do comerciante, por exigirem tipos de ervas cada vez mais distintas daquelas pertencentes à sua mistura.

A localização de cada grupo de consumidores dependerá não só da influência que o comerciante exerce sobre o grupo, mas também da influência que outros comerciantes exercem sobre os outros grupos de consumidores. Um comerciante que possui uma mistura parecida com a de outro comerciante viaja para a mesma região que o outro, já que ele já sabe que ali sua mistura terá uma boa aceitação. É neste sentido que as relações de vizinhança são preservadas. Se dois comerciantes possuem misturas similares é porque eles moram próximos um do outro, como já explicado inicialmente. Eles também irão vender suas misturas para clientes vizinhos pois é sabido que eles possuem gostos parecidos. Assim, grupos de comerciantes tornam-se associados a grupos de consumidores. Comerciantes com interesses em

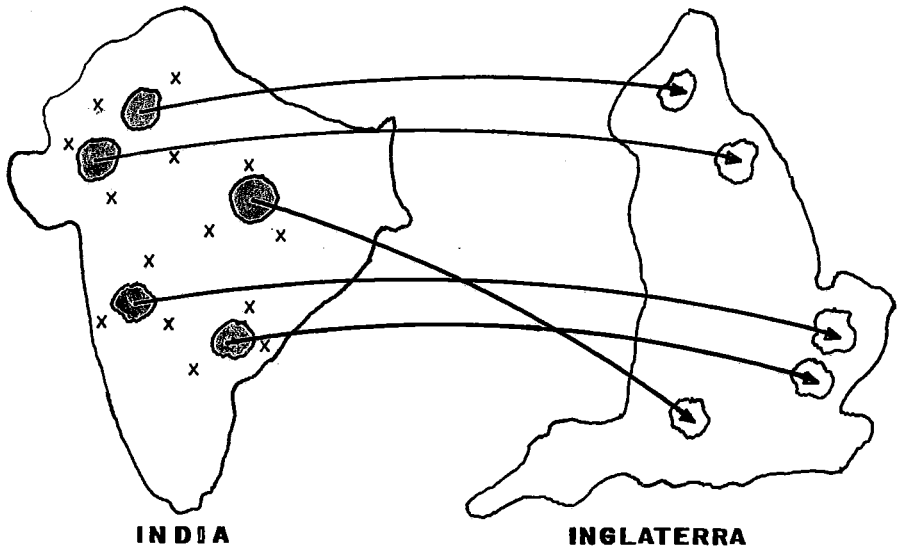


Figura II.2: Uma configuração final para o sistema de comércio

comum intensificarão seus contatos em determinadas regiões, extinguindo-os completamente com regiões que não possuírem nenhuma similaridade com suas misturas. Eles cooperam entre si a fim de desenvolver seus contatos.

Uma configuração final para esse sistema de comércio poderia então ser dada pela figura II.2. O desenvolvimento das rotas de comércio simula intuitivamente o comportamento de sistemas dinâmicos que evoluem para estados de equilíbrio. Tais estados equivaleriam a configurações finais do sistema de comércio, equivalentes a rotas estáveis e lucrativas. As configurações iniciais corresponderiam a pontos de instabilidade. Supondo-se que essas rotas foram estabelecidas sem nenhum conhecimento prévio das regiões de chegada, não há nenhuma garantia inicial do estabelecimento de um mercado lucrativo. A evolução do sistema então se dá em busca desse objetivo. Na sua grande maioria, essas configurações possuem relações de vizinhança bem definidas entre cidades da Índia e cidades da Inglaterra. Podemos interpretar esses estados finais de comércio como um mapeamento contínuo por partes, sem nenhuma orientação pré-definida.

A analogia desse sistema de comércio com o modelo de Malsburg e Willshaw citado anteriormente é direta. É interessante ressaltar que as rotas de comércio representam justamente o mapeamento entre os dois países. Com a

evolução desse mapeamento, as rotas se modificam, auto-organizando-se. Comportamento semelhante ocorre entre as conexões entre as superfícies de células nervosas no modelo biológico. Associaremos, através de uma tabela, os elementos e todo o comportamento dos dois modelos.

Modelo do comércio de chá	Modelo de Malsburg e Willshaw
Comerciantes indianos	Células da superfície pré-sináptica
Consumidores ingleses	Células da superfície pós-sináptica
Cada tipo de erva presente na mistura	Um dos diferentes tipos de marcadores químicos
Rotas de comércio	Conexões entre as superfícies
Quantidade de erva associada a cada consumidor	Taxa de difusão de um conjunto de marcadores químicos numa célula pós-sináptica
Relações de vizinhança entre comerciantes de chá expressas pela similaridade de suas misturas	Relações de vizinhança entre as células da superfície pré-sináptica expressas pelos marcadores químicos
Relações de vizinhança entre consumidores de chá expressas pela similaridade de suas misturas	Relações de vizinhança entre as células da superfície pós-sináptica expressas pela similaridade entre marcadores químicos provenientes das conexões e aqueles já existentes na superfície

Segundo Willshaw [11], o modelo do comércio de chá, exposto acima, também motivou Kohonen, ([17], 1982) ao desenvolvimento de um método para o estabelecimento de mapeamentos topográficos entre espaços de diferentes dimensões. Análises matemáticas das condições de estabilidade de mapeamentos topográficos

em geral, foram desenvolvidas, por exemplo, por Takeuchi &Amari ([28], 1979) e Häussler e von der Malsburg ([12], 1983).

O problema da formação de mapeamentos topográficos pode também ser definido como um problema de Otimização Combinatória. A função objetivo do problema se traduz em se encontrar as posições relativas das extremidades das conexões, na superfície pós-sináptica, levando-se em conta a restrição de que as origens dessas ligações, na superfície pré-sináptica, sejam vizinhas. O mapeamento é feito de maneira a maximizar a preservação, no espaço de chegada, das relações de vizinhança presentes no espaço de saída [11].

O método Elástico é uma extensão direta do modelo de comércio de chá. Ele foi desenvolvido inicialmente por Durbin & Willshaw em 1987 para resolver o problema do Caixeiro Viajante, e possui uma filosofia essencialmente geométrica. A idéia de preservação de vizinhanças discutida acima como uma característica de mapeamentos topográficos é usada amplamente no algoritmo. Durante toda a sua evolução, trabalha-se com pontos distribuídos sobre um anel de maneira a serem associados a pontos em um plano Euclideano que representam posições de cidades distribuídas sobre esse plano. Essa associação, ou mapeamento, é tal que as distâncias sejam preservadas, ou seja, dois pontos próximos no anel são mapeados em duas cidades similarmente próximas no plano. As relações de vizinhança são expressas de acordo com a distância entre os pontos. As propriedades e características desse método serão discutidas com mais detalhes nas próximas seções.

II.3 A influência das Teorias da Mecânica Estatística no Algoritmo Elástico

O comportamento de alguns sistemas físicos termodinâmicos sugere alternativas eficientes para a formulação de algoritmos para solucionar aproximadamente problemas de grande complexidade computacional, como muitos de Otimização Combinatória.

Esses sistemas termodinâmicos basicamente traduzem-se na evolução dinâmica do movimento de partículas que afetam a energia do sistema, guiando-a

em direção a mínimos de energia, equivalentes a estados estáveis do sistema. Durante toda a evolução, esses sistemas são resfriados lentamente com o objetivo de se chegar ao menor mínimo de energia atingido pelo sistema, para que este atinja uma configuração uniforme onde é possível observar da melhor maneira suas características.

A idéia desse resfriamento foi usada em alguns algoritmos formulados para a solução de problemas de Otimização como o problema do Caixeiro Viajante. Exemplos como "*Simulated Annealing*" [16] e o *Algoritmo Elástico* possuem parâmetros que simulam o comportamento desse resfriamento. Tais parâmetros permitem que a função de energia associada ao sistema não estacione de vez em mínimos locais, que eventualmente podem estar muito distantes da melhor solução para o problema. A análise detalhada do comportamento do "*Simulated Annealing*" pode ser vista em [3] [5] [16].

Como a maioria dos problemas de Otimização Combinatória se reduz à minimização ou maximização de uma função, é intuitivo resolvê-los através da simulação do comportamento de tais sistemas. É possível mostrar a equivalência existente entre esses sistemas e o Método Elástico, simplesmente definindo-o como um conjunto de equações diferenciais de primeira ordem, que minimizam uma função de Lyapunov definida para o problema e que simulam o comportamento da função de energia de tais sistemas.

A possibilidade da formulação do problema do Caixeiro Viajante em termos da minimização de uma função de energia permite também interpretá-lo probabilisticamente. Tal interpretação é importante, já que assim é possível que se faça uma conexão com métodos probabilísticos clássicos, ferramentas fundamentais na elaboração de um bom embasamento teórico para esses modelos, e com isso fazer também uma conexão com técnicas usadas na Mecânica Estatística. Muitas dessas técnicas já foram usadas como base na elaboração de modelos importantes como os modelos de Hopfield & Tank (1984, 1985) e "*Simulated Annealing*" [Kirkpatrick, 1983], já mencionados anteriormente.

Yuille [31] e Simic [27], em seus trabalhos, demonstraram uma co-

nexão muito interessante existente entre o Algoritmo Elástico e o modelo neuronal de Hopfield & Tank. Ambos podem ser derivados de certos modelos fisícos e possuem características relacionadas a princípios fundamentais da Mecânica Estatística. Tal conexão é sugerida por Yuille quando ele mostra que, tanto a função de energia do Algoritmo Elástico, quanto a função de energia do modelo neuronal de Hopfield & Tank, podem ser obtidas de uma função de energia mais geral, definida em função das características do problema do Caixeiro Viajante.

II.4 Descrição do algoritmo

O *Algoritmo Elástico* pode ser aplicado ao problema do Caixeiro Viajante definido em um espaço Euclideano de dimensão n. Para fins de maior visualização e compreensão do método, trabalharemos apenas no espaço bi-dimensional.

Consideremos então, como espaço de trabalho, um quadrado de lado L, onde N cidades se encontram distribuídas. Consideremos, também, M pontos distribuídos igualmente sobre uma circunferência cujo centro coincide com a centróide das cidades.

Sejam X_i, $i = 1,..., N$ e Y_j, $j = 1,..., M$, respectivamente, as coordenadas da cidade i e do ponto j situado sobre a circunferência.

O *Algoritmo Elástico* basicamente "deforma" gradualmente esse círculo, através da constante atualização das coordenadas dos pontos j, obedecendo a uma determinada equação que rege esses deslocamentos. Visualmente, é como se esticássemos a circunferência lentamente até que ele se ajustasse às posições das cidades, formando um caminho fechado passando por todas elas uma única vez. Associada a esse caminho, existe uma função de energia, denotada por E, que é minimizada de acordo com a redução das distâncias entre os pontos do anel e as cidades no plano e do comprimento da *tour*. A função E e a equação de deslocamento dos pontos do anel são definidas em função de um parâmetro K, que determina, durante toda a evolução do algoritmo, a maior ou menor deformação do anel. Esse parâmetro possui comportamento semelhante ao parâmetro que simula a tempera-

tura no algoritmo "*Simulated Annealing*". Mais formalmente:

Dados de entrada:

Posições de N cidades distribuídas sobre um quadrado de lado L.

Posições de M pontos distribuídos igualmente sobre uma circunferência de centro no centróide das cidades.

$M = 2N$.

Inicialização de todos os parâmetros necessários ao desenvolvimento do algoritmo.

Estabelecimento da taxa de redução do parâmetro K.

Evolução:

Enquanto $K > 0$ faça

Para cada uma de n iterações faça

Para cada $j \in \{1, \ldots, M\}$ faça

atualize as coordenadas de Y_j segundo II.1

Reduza o valor de K de acordo com

uma taxa de redução bem pequena

Critério de Parada:

O algoritmo pára quando para cada uma das N cidades i existir pelo menos um ponto do anel situado a uma distância desprezível.

O deslocamento sofrido por cada ponto do anel obedece à equação

$$\Delta Y_j = \alpha \sum_i w_{ij}(X_i - Y_j) + \beta K(Y_{j+1} - 2Y_j + Y_{j-1}) \qquad \text{(II.1)}$$

O primeiro termo de II.1 representa a força de atração que as cidades exercem sobre o ponto j do anel. É ele que "puxa" j na direção das cidades. O

coeficiente w_{ij} responde pela influência que a cidade i exerce sobre o ponto j. Essa influência é calculada em função da distância entre o ponto e a cidade, e em função do parâmetro K. Exercerá maior influência sobre o ponto a cidade que estiver situada mais próxima a ele. Conseqüentemente, cidades muito distantes do ponto, exercerão influência ínfima sobre ele. Para simular tal comportamento, adotou-se o uso da função exponencial.

O número máximo de pontos distribuídos pelo anel (M), corresponde aproximadamente ao dobro do número de cidades. Estabelecer um valor para M maior que N é interessante, pois permite uma maior flexibilidade na escolha, pelo algoritmo, do ponto do anel que irá convergir para cada uma das cidades do plano, contribuindo a todo momento, para a preservação de uma continuidade na deformação da *tour*.

O termo w_{ij} é então definido como

$$w_{ij} = \frac{\phi(\mid X_i - Y_j \mid, K)}{\sum_k \phi(\mid X_i - Y_k \mid, K)}, \tag{II.2}$$

onde $\phi(\mid X_i - Y_k \mid, K)$ é uma função positiva definida por

$$e^{-|X_i - Y_j|^2/2K^2}. \tag{II.3}$$

A normalização de w_{ij}, realizada pelo denominador de II.2, é importante para que se garanta a distribuição uniforme dos pontos do anel por todas as cidades do plano, possibilitando assim o funcionamento do critério de parada do algoritmo. Se existir pelo menos uma cidade para qual não convirja nenhum dos pontos do anel, o algoritmo nunca terminará. Essa situação seria comum se w_{ij} não fosse normalizado. Nesse caso, como cada ponto seria atraído para a cidade que estiver mais próxima de si, basta considerar a possibilidade, por exemplo, de uma mesma cidade possuir a menor distância dentre todas as outras, em relação a dois pontos distintos do anel, como na figura II.3.

Observando o comportamento da função ϕ e a figura II.3, é possível concluir que os pontos 1 e 2 do anel se deslocarão em direção à cidade A até que suas posições coincidam com as coordenadas dessa cidade. Desta maneira, mesmo que o

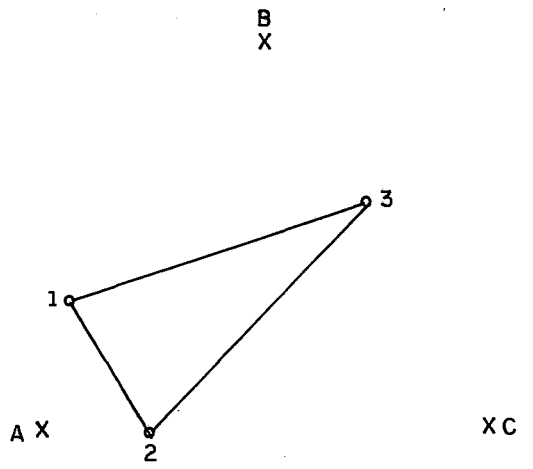


Figura II.3: A cidade A é a a mais próxima dos pontos 1 e 2

outro ponto convirja para alguma das outras cidades, existirá pelo menos uma pela qual não passará nenhum ponto do anel, e assim não existirá uma *tour* passando por todas as cidades. A normalização de w_{ij} então, permite que a força de atração das cidades seja "balanceada" pelos pontos do anel. Desta forma, na situação acima, a influência total de cada cidade seria unitária, e então, um peso maior seria atribuído ao ponto que realmente estivesse mais próximo, em detrimento dos pontos restantes, possibilitando que estes fossem atraídos por outras cidades.

Observando-se a equação II.3, é interessante notar que, quando K assume um valor bem pequeno, os deslocamentos sofridos pelos pontos em direção às cidades são bem suaves, pois como eles já estão perto de seus objetivos, o avanço em direção às cidades torna-se bem lento.

O parâmetro α no primeiro termo da equação II.1, funciona como um "regulador" da maior ou menor atração do ponto pelas cidades. O segundo termo da equação II.1 representa a força de atração que os vizinhos do ponto j no anel, exercem sobre ele. O coeficiente βK responde pela intensidade dessa atração. É o segundo termo da equação II.1 que é relacionado com o comprimento da *tour*. O parâmetro K é decrementado lentamente até que, no final do processamento do algoritmo, K assuma um valor próximo de zero. Assim, para um determinado valor

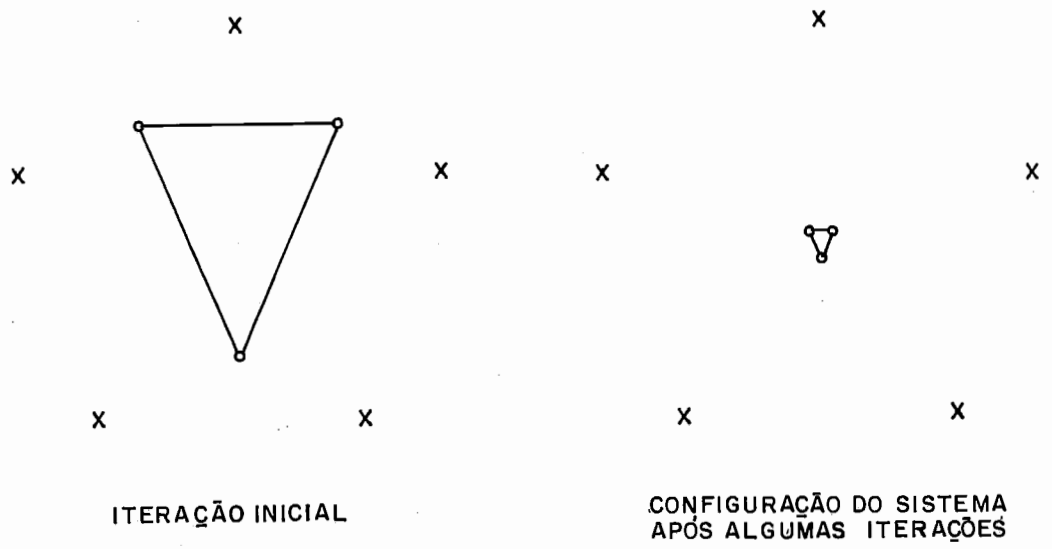


Figura II.4: O "encolhimento" do anel

de β, a intensidade da atração do ponto pelos seus vizinhos diminui de acordo com o decréscimo de K. Se o valor de β for alto, a circunferência inicial, onde estão distribuídos os pontos, "encolhe", transformando-se quase que num ponto. Com a evolução do algoritmo, ele volta a se deformar, transformando-se por fim, na *tour* que passa por todas as cidades. Essa situação é ilustrada na figura II.4.

É interessante observar que os parâmetros α e β são responsáveis pelo balanceamento das forças na equação (II.1). Com a ajuda deles, é possível ajustar o equilíbrio entre a força que puxa os pontos para as cidades e a força que os atrai entre si.

À cada valor de K, cada ponto do anel sofre n deslocamentos, de acordo com o número de iterações estabelecido para cada K. Na verdade, para cada K é interessante que o algoritmo seja processado, calculando sucessivamente os deslocamentos de cada ponto do anel, até que a função de energia E, associada ao algoritmo, chegue a um mínimo. Essa função foi definida como

$$E(Y_j, K) = -\alpha K \sum_i \ln \sum_j e^{\frac{-|X_i - Y_j|^2}{2K^2}} + \frac{\beta}{2} \sum_j |Y_{j+1} - Y_j|^2,$$

que possui a propriedade de ser reduzida a cada deslocamento sofrido por cada ponto

do anel. Mais formalmente, poderíamos descrever tal propriedade como

$$\Delta Y_j = -K\frac{\partial E}{\partial Y_j}.$$

Como E é limitada inferiormente, essa redução ocorre até que a função atinja algum mínimo. A constante mudança do valor de K é interessante, para que seja possível assim, a procura de mínimos locais mais próximos do ótimo global. Com o decréscimo do valor de K, a energia se desestabiliza e então novas iterações são necessárias para calcular outro mínimo de energia, para o novo valor de K. Para efeitos de rapidez na execução do algoritmo, foi fixado o número de iterações para cada K.

Durante a evolução do algoritmo, a influência das cidades sobre os pontos do anel vai se modificando. Em outras palavras, no decorrer do algoritmo cada cidade se torna associada a seções do anel, ou seja, alguns pontos do anel se tornam associados a ela, de acordo com a distância a que eles se encontram dela. É o parâmetro K o responsável pelo aumento de especificidade de cada cidade em relação aos pontos do anel. Quando K é pequeno, seções do anel tornam-se associadas a subconjuntos de cidades, dispostas na forma de conglomerados ou "clusters". Dessa maneira, todos os pontos no final do processamento estarão associados a todas as cidades do plano. Nesse momento, também o anel estará deformado de tal maneira que se transformará numa *tour* que passa por todas as cidades.

II.5 Aplicações do Algoritmo Elástico e alguns trabalhos relacionados

Wong & Funka-Lea [30] utilizaram o algoritmo Elástico desenvolvido por Durbin & Willshaw em uma extensão do Caixeiro Viajante. Essa extensão consiste basicamente no Caixeiro Viajante com mais uma restrição: vários obstáculos espalhados entre as cidades. O que se quer é se encontrar o menor caminho fechado, que passe por todas as cidades, evitando os obstáculos. Uma alteração realizada por Wong & Funka-Lea no Elástico foi na definição da função de energia associada ao algoritmo

Foi introduzido um terceiro termo que funciona como um termo de "penalização", para prevenir que o caminho entre cada duas cidades do *tour* se interceptem com algum dos obstáculos. Resultados de testes com até 100 cidades, com razoável desempenho, foram exibidos neste trabalho.

Já em [4], Burr propôs uma melhoria para o Algoritmo Elástico de Durbin & Willshaw adicionando a idéia de *forças simétricas*. Foi adicionado mais um termo na equação de cálculo de deslocamento do ponto do anel. Enquanto um termo responde pela força de atração que a cidade i exerce sobre o ponto do anel que vai ser deslocado no momento, o outro representa a força contrária que atrai a cidade para o ponto do anel que estiver mais próximo dela. A variação do parâmetro K é feita de acordo com o cálculo da distância média entre cada cidade e o ponto do anel mais próximo dela. Essa medida é interessante pois ajuda na rapidez da convergência dos pontos do anel às cidades no plano. Com essa melhoria, a convergência do algoritmo foi atingida mais rapidamente.

Kohonen [17], em 1982, propôs um algoritmo motivado pelo processo de auto-organização de certas estruturas biológicas. Apesar de ser facilmente especificado, a análise matemática desse algoritmo é bastante sofisticada. Esse algoritmo já foi utilizado em várias aplicações, como em reconhecimento de fala (Kohonen, 1988). Fort [10] reformulou o algoritmo de Kohonen para aplicá-lo no problema do Caixeiro Viajante.

A motivação e a idéia desse algoritmo se assemelha bastante ao Algoritmo Elástico. Existem, basicamente, duas diferenças; a primeira é que o Elástico é determinístico; a segunda, no Elástico, uma função de energia é especificamente definida para o problema com o objetivo de minimização de alguma função. Baseando-se no algoritmo de Kohonen, Hueter [15] propôs a construção de um anel adaptativo para a solução do Caixeiro Viajante. Basicamente, uma solução é alcançada, adaptando-se um conjunto de N nós a um conjunto de N cidades espalhadas dentro de um quadrado de lado unitário. É então selecionada uma cidade k. Os nós do anel competem entre si, selecionado-se aquele que se encontra mais próximo de k. As novas posições dos pontos são ajustadas obedecendo à regra de aprendizado de Kohonen. Analogamente ao Elástico, esse algoritmo pára quando todos os nós

tiverem convergido para as cidades.

Angéniol, La Croix Vaubois, e Texier [1] propuseram um algoritmo com uma idéia semelhante ao Elástico. Nós distribuídos em um anel continuamente evoluem num processo iterativo, até que cada cidade no plano tenha sido "capturada" por algum ponto do anel. A idéia da atração dos pontos pelas cidades e da atração de cada ponto pelos seus vizinhos no anel tem aqui um enfoque um pouco diferente. O nó do anel que estiver mais próximo da cidade em questão se move em direção a ela, induzindo os outros nós do anel a fazerem o mesmo, porém numa intensidade de deslocamento menor. Essa correlação entre os nós do anel determina implicitamente a tendência de minimização das distâncias entre os pontos do anel. Determinado pelo processo de competição inerente ao algoritmo, processos de eliminação e criação de nós são associados a ele, permitindo a redução de cálculos através da eliminação de pontos não selecionados pelo processo de competição. Seu desempenho quanto à rapidez de execução mostrou-se bastante satisfatório, mas os resultados de comprimento de *tour* foram piores que os obtidos pelo Elástico, quando aplicado aos mesmos conjuntos de cidades.

Em [8], uma comparação de desempenhos de alguns algoritmos aplicados ao problema do Caixeiro Viajante é realizada. Dentre todos, o algoritmo Elástico apresenta os melhores resultados, perdendo apenas para o Algoritmo Genético.

Goodhill & Willshaw ([11], 1989) usaram o Algoritmo Elástico numa nova aplicação — o desenvolvimento, no sistema visual dos vertebrados, de projeções binoculares em forma de faixas. Em 1979, Malsburg já havia proposto uma extensão do modelo de comércio de chá entre a Índia e Inglaterra, descrito neste capítulo, para a formação de faixas após o estabelecimento dos mapeamentos topográficos. Nesse modelo, ele postulou a existência de marcadores químicos para indicar a visão.

Para utilizar as características do Algoritmo Elástico aplicado ao Caixeiro Viajante, Goodhill & Willshaw formularam o problema em termos do alongamento de uma corda elástica entre pontos, representando respectivamente o córtex visual e células pré-sinápticas. Foram estudados dois casos nesse trabalho: o ma-

peamento de duas retinas numa região do cérebro no caso de uma e duas dimensões. No primeiro caso, as células das duas retinas dispunham-se paralelamente em forma de fila, separadas por uma distância especificada pelo algoritmo. O córtex, representado pelo anel elástico, move-se livremente entre as células. A distância entre um ponto situado no anel e uma célula na retina representa a quantidade de conexões existentes entre aquela parte do córtex e a célula. A topologia do anel representa a topologia do córtex. Regiões vizinhas no córtex são representadas por pontos vizinhos no anel. A disposição em faixas no córtex representa justamente a intensidade de conexões existentes nas suas diversas partes. A equivalência pode ser feita diretamente com o Algoritmo Elástico aplicado ao Caixeiro Viajante, associando-se as células da retina às cidades no plano, e as células no córtex aos pontos no anel. O caso bi-dimensional possui um mecanismo similar.

Capítulo III

O Filtro — uma variação do Algoritmo Elástico

III.1 Introdução

Será apresentada neste capítulo uma variação do Algoritmo Elástico proposta neste trabalho — denominada **Filtro** — que visa melhorar o seu desempenho em termos de tempo de execução. A avaliação das possíveis alterações nos resultados, no que diz respeito ao cálculo do comprimento final da *tour* e tempos de execução, será apresentada no capítulo V.

Nas próximas seções, serão discutidos tanto os motivos que deram origem à elaboração da variação, quanto a sua descrição.

III.2 Motivação

Como explicado no capítulo anterior, o Algoritmo Elástico, quando aplicado ao problema do Caixeiro Viajante, consiste na sua essência, em sucessivas atualizações de pontos distribuídos sobre um anel, de maneira a deformá-lo, transformando-o numa *tour* que passa pelas cidades distribuídas no plano.Essas atualizações obedecem a equação II.1 que basicamente rege os deslocamentos dos pontos em função de suas relativas distâncias aos seus vizinhos no anel e às cidades no plano. O termo dessa equação que responde pela atração do ponto pelas cidades possui um coeficiente, w_{ij}, já mencionado e descrito no capítulo II, que fornece a influência que a cidade i exerce

sobre o ponto j do anel em função da distância entre i e j. Segundo essa equação, a atualização das coordenadas de um ponto j do anel exige o cálculo das distâncias de todas as cidades do plano a j. Como existem M pontos distribuídos pelo anel, são realizadas $O(NM)$, ou melhor, $O(2N^2)$ cálculos a cada uma das iterações realizadas pelo algoritmo, já que M corresponde aproximadamente ao dobro do número de cidades. Conseqüentemente, à medida que o número de cidades cresce, o tempo de processamento requerido a cada iteração aumenta muito.

A preocupação com a redução do tempo de processamento do algoritmo motivou a elaboração, nesta tese, de modificações realizadas em cima de dois pontos: o primeiro, o número de pontos do anel nas primeiras iterações, e o segundo, o número de cálculos de influências, a cada iteração.

III.3 Descrição

A seguir serão descritas as duas modificações que constituem a variação do Algoritmo Elástico.

A primeira consiste na manipulação dos pontos do anel. Inicia-se o processamento com poucos pontos e a cada decréscimo no valor de K, duplica-se essa quantidade até que se atinja o número máximo de pontos, correspondente ao dobro de cidades, aproximadamente. Considere, para uma maior visualização, a aplicação do Algoritmo Elástico na solução do Caixeiro Viajante com 200 cidades. Com 400 pontos distribuídos pelo anel, 80.000 cálculos seriam realizados a cada iteração, durante todo o processamento do algoritmo. Se, no entanto, iniciássemos o processamento com apenas três pontos, numa primeira iteração seriam realizados apenas 600 cálculos. Na próxima redução do valor de K, esse número aumentaria para 1.200 e assim sucessivamente, até que fosse atingido o número máximo de pontos do anel. Procedendo dessa forma, uma quantidade razoável de cálculos pode ser evitada, reduzindo conseqüentemente o tempo de processamento. No início da deformação, o número reduzido de pontos no anel não prejudica a evolução do algoritmo, desde que seja estabelecido proporcionalmente ao número de cidades, alterando-se apenas a maneira como o anel é alongado. No final do processamento,

no entanto, é importante que se tenha um número de pontos maior que o de cidades, para que se possibilite a convêrgencia do algoritmo, associando-se pelo menos um ponto a cada uma das cidades.

A outra modificação consiste na redução do cálculo de distâncias, levando-se em conta a influência das cidades sobre os pontos do anel.

Através da análise da equação II.3, no capítulo anterior, é possível concluir que, dado um ponto do anel, as cidades mais próximas exercem uma influência muito mais significativa sobre o ponto do que as outras. O cálculo da influência de uma cidade muito distante do ponto pode ser desnecessária, acarretando apenas desperdício de tempo de processamento, pois sua contribuição para o deslocamento do ponto é praticamente desprezível.

Nesse caso, dado um ponto, é importante que se calcule apenas as influências das cidades que efetivamente contribuem para o seu deslocamento. É aí que surge o problema. Como avaliar adequadamente se uma determinada cidade é ou não significativa para o cálculo do deslocamento de um ponto? Como determinar exatamente aquelas que devem ser realmente levadas em consideração nesse cálculo e as que podem ser desprezadas, sem alterar a evolução da deformação da *tour*, evitando-se perdas significativas em seu comprimento?

Considerando C como o conjunto de todas as cidades distribuídas pelo plano, nossa proposta se resume em determinar, para cada ponto j do anel e cada valor de K, um subconjunto de C associado a j, $C_\omega(j,K)$, que compreenda apenas as cidades que realmente possuam uma influência significativa sobre o ponto. Deseja-se então que

$$\sum_{i \in C_\omega(j,K)} \phi(d_i, K) = \omega \sum_{i \in C} \phi(d_i, K). \qquad \text{(III.1)}$$

onde $d_i = |X_i - Y_j|$ e $\omega \in (0,1)$ é responsável por "filtrar" a quantidade de cálculos a ser realizada.

A tarefa de se identificar os elementos de cada um desses subconjun-

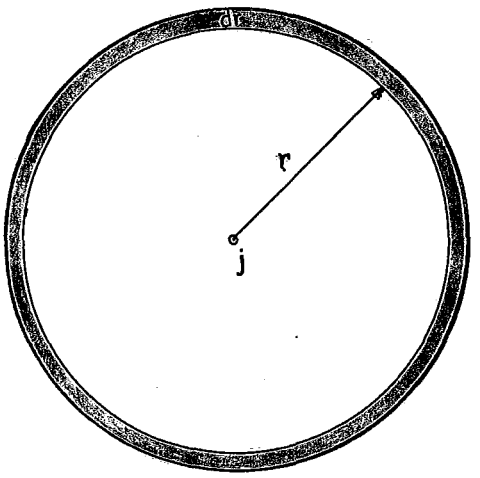


Figura III.1: Área de influência das cidades que distam r de j

tos é muito difícil, uma vez que eles variam de acordo com a posição de cada ponto j. Por esse motivo, adotou-se uma aproximação, baseada na hipótese da existência de um número muito grande de cidades, distribuídas uniformemente por todo o plano, com densidade ρ.

De acordo com essa hipótese, considerando-se que os elementos de $C_\omega(j,K)$ se encontram no interior de um círculo de raio R e centro Y_j, o somatório das influências das cidades pertencentes a $C_\omega(j,K)$ pode ser dado aproximadamente por

$$\sum_{i \in C; d \leq R} \phi(d,K) \approx \rho \int_{r=0}^{R} 2\pi\, \phi(d,K) r dr \qquad \text{(III.2)}$$

onde $2\pi r dr$ representa a área que compreende todas as cidades de $C_\omega(j,K)$ que estão situadas à uma distância r de j.

Desenvolvendo o lado direito da equação III.2 obtemos

$$\begin{aligned} \rho \int_{r=0}^{R} 2\pi\, \phi(d,K) r dr &= 2\pi\rho \int_{r=0}^{R} e^{\frac{-r^2}{2K^2}} r dr \\ &= 2\pi\rho K^2 \left(\exp(-R^2/2K^2) - 1\right). \end{aligned} \tag{III.3}$$

Se considerarmos todas as cidades do quadrado de lado L, como compreendidas em um círculo de raio $R = L\sqrt{\frac{1}{\pi}}$ cuja área corresponde à área do quadrado, podemos deduzir, pela aproximação desenvolvida em III.2 e III.3 que

$$\sum_{i \in C; d \leq L\sqrt{\frac{1}{\pi}}} \phi(d,K) \approx 2\pi\rho K^2 \left(\exp(-L^2/2\pi K^2) - 1\right). \tag{III.4}$$

Considerando-se que para cada valor de ω e cada valor de K, os elementos do conjunto $C_\omega(j,K)$ estão compreendidos em um círculo de raio $R_\omega(K)$, igual para todos os pontos do anel, podemos deduzir, das equações III.1, III.2, III.4 e III.4 que

$$\exp(-R_\omega^2(K)/2K^2) - 1 = \omega(\exp(-L^2/2\pi K^2) - 1).$$

de onde se obtêm o valor desejado de $R_\omega(K)$

$$R_\omega(K) = \sqrt{-2K^2\, ln(-\omega\,(1 - e^{\frac{-L^2}{2K^2\pi}}) + 1)}. \tag{III.5}$$

Analisando a expressão III.5, observa-se que o raio do círculo diminui de acordo com o decréscimo de K, o que é razoável, pois, como discutido no capítulo II, à medida que o valor de K diminui, seções do anel se tornam mais específicas de pequenos grupos de cidades. Desta forma, não se tem necessidade de que o número de cidades a ser considerado no cálculo do deslocamento de um ponto seja muito grande. Sendo assim, é interessante que o círculo de raio $R_\omega(K)$ que envolve o ponto

seja pequeno. O valor de ω determina, no início do processamento, o maior ou menor comprimento do raio do círculo referente a cada ponto. Esse parâmetro age como um "filtro" que controla o número de cidades a serem pesquisadas. Se ω assumisse o valor unitário, o comportamento do algoritmo, com o filtro, seria semelhante ao original, pois $R_\omega(K)$ seria, a toda iteração, fixo e igual a $L\sqrt{\frac{1}{\pi}}$. No entanto, algumas cidades seriam desprezadas pois o que se tem, é a igualdade das áreas do círculo e do quadrado. Quanto menor for o seu valor, menor será a área considerada para cada círculo no início da evolução do algoritmo.

No Filtro, o somatório do primeiro termo da equação de deslocamento de cada ponto do anel é realizado apenas sobre as cidades que pertencem ao conjunto $C_\omega(j, K)$, assim como a normalização do coeficiente w_{ij} é realizada apenas sobre os pontos que sofrem influência de i.

III.4 A implementação

O processo de duplicação dos pontos sobre o anel foi implementado de maneira a manter uma uniformidade em sua distribuição. Para representá-los, foi criada uma lista encadeada contendo os valores de suas coordenadas. A cada duplicação, novos pontos são inseridos entre os elementos da lista, preservando uma distribuição uniforme.

Já para facilitar a implementação da outra modificação, referente à formação de círculos, foi realizada a divisão do quadrado onde estão distribuídas todas as cidades, em pequenos quadrados de lado δ. O tamanho de δ foi estimado como a distância média entre as cidades.

Desta maneira, pode-se associar diretamente cada cidade ao quadrado a que ela pertence. Analogamente, pode-se identificar o quadrado $\delta \times \delta$ a que cada ponto do anel está associado. O processo então se resume no seguinte:

Dado um ponto do anel, calcula-se o valor de $R_\omega(K)$ para aquele ponto através de III.5. Uma vez calculado, identifica-se quais quadrados $\delta \times \delta$ interceptam o círculo de raio $R_\omega(K)$. O deslocamento do ponto será efetuado ape-

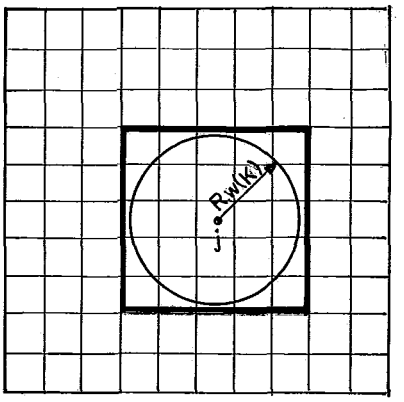


Figura III.2: Área considerada no cálculo do deslocamento de j

nas em função do cálculo das influências das cidades situadas no interior desses quadrados. Esse procedimento é repetido para todos os pontos do anel. A figura III.2 representa, para um determinado ponto j, os quadrados que interceptam o círculo de raio $R_\omega(K)$, onde estão situadas as cidades consideradas para o cálculo do deslocamento de j.

Na nossa implementação, encontram-se associadas a cada um dos quadrados $\delta \times \delta$, duas listas, referentes respectivamente, às cidades e aos pontos situados no seu interior.

O número de quadrados considerados diminui de acordo com a evolução do algoritmo, assim como diminui a área dos círculos centrados nos pontos do anel.

Desta forma, o algoritmo é processado "localmente", i.e., apenas seções do quadrado de lado L são consideradas no cálculo do deslocamento dos pontos.

Capítulo IV

O Algoritmo de Lin & Kernighan — LK

IV.1 Introdução

O problema do Caixeiro Viajante, por ser NP-completo, é extensamente usado para testar a eficiência de muitos algoritmos importantes. A simples enumeração de configurações viáveis para o problema cresce exponencialmente com o número de cidades. Por esse motivo, foi proposta uma série de algoritmos como alternativas mais eficientes de solução, no sentido de, mesmo sem a garantia do ótimo, encontrar uma solução razoável num tempo de computação viável, ou seja, polinomial. A avaliação da eficiência desses algoritmos se dá, muitas vezes, em função da comparação de seus resultados.

Neste capítulo será apresentado um algoritmo proposto por Lin & Kernighan (1973), referenciado ao longo de todo este trabalho por LK, considerado como clássico na solução do problema do Caixeiro Viajante, já que, apesar de ter sido proposto há bastante tempo, possui um desempenho melhor que várias abordagens propostas recentemente. Por isso, será usado aqui como comparação para o Algoritmo Elástico.

IV.2 O algoritmo

Dado um conjunto S, descobrir um subconjunto T de S que satisfaça algum critério C e minimize uma função f, é um típico problema de Otimização. Como já foi dito, em caso de S finito ou enumerável, o problema é considerado Combinatório.

Nesses termos, o problema do Caixeiro Viajante se resume em descobrir o subconjunto T, do conjunto S de todas as arestas de um grafo completo de N nós (correspondentes ao número de cidades do problema), que forma uma tour de comprimento mínimo.

Para resolver tais problemas, surgiram muitos métodos aproximativos (heurísticas). A maioria desses métodos consiste em processos iterativos de melhoria, onde soluções iniciais geradas aleatoriamente para o problema são modificadas segundo algum critério, gerando soluções melhores(i.e., de menor custo). É a formulação desses critérios que diferencia uma heurística da outra. Em outras palavras, esses processos iterativos resumem-se em gerar uma solução T' a partir de uma solução inicial T, satisfazendo a condição $f(T') < f(T)$. $f(T)$ representa a função de custo do problema associada à solução T. Esse processo é repetido até que a condição acima não seja mais satisfeita, o que indica que um mínimo local foi encontrado. Esse mínimo corresponderá à solução final gerada pelo algoritmo. Se essa solução estiver muito distante do ótimo, novas configurações iniciais poderão ser geradas e então todo o processo pode ser repetido em busca de um novo ponto de mínimo.

Um exemplo desse tipo de heurística consiste em um procedimento discreto de troca de um número fixo de elementos entre os conjuntos T e $S - T$, modificando T e gerando uma solução T', viável e melhor. Quando não for mais possível obter soluções melhores através dessas trocas sucessivas, é porque algum mínimo local foi encontrado. O que se deseja é que esse mínimo coincida com o mínimo global correspondente à solução ótima para o problema. Para isso, é preciso que se escolham os elementos certos a serem trocados, assim como o número k apropriado de elementos. Infelizmente, a satisfação dessas condições não é uma tarefa trivial.

Esse tipo de estratégia já foi aplicada em trabalhos anteriores com considerável sucesso, com k fixado em 2 (Croes [6]) e em 3 (Lin [18]). Fixar o valor de k, no entanto, pode não ser vantajoso. Se k for grande, o número de elementos a serem trocados entre os conjuntos também o será, e assim, conseqüentemente, o tempo de computação crescerá também. A escolha de k pequeno pode, muitas vezes, fazer com que boas soluções para o problema não sejam alcançadas. É preciso então encontrar um valor para k de modo que se obtenha uma solução de boa qualidade num tempo viável de computação. Baseando-se nisso, Lin & Kernighan propuseram uma maneira de se determinar o valor de k iterativamente, durante o processamento do algoritmo. Dada uma solução inicial T, deseja-se encontrar k elementos $x_1, \ldots, x_k$ de T que ao serem trocados por $y_1, \ldots, y_k$ de $S - T$, transformam T numa nova solução T', de melhor custo. Como não se sabe a priori, o valor apropriado de k e os elementos $x_1, \ldots, x_k$ e $y_1, \ldots, y_k$ a serem trocados, tenta-se descobrí-los iterativamente, elemento a elemento. Em outras palavras, tenta-se identificar o par de elementos (x_1, y_1) cuja troca acarreta um ganho ou melhoria na função de custo associada à solução. Feito isso, determina-se o segundo par de elementos, que junto com (x_1, y_1) continua produzindo ganho. O processo continua até que a troca do conjunto $X = \{x_1, \ldots, x_k\}$ por $Y = \{y_1, \ldots, y_k\}$ não produza mais ganhos.

Mais formalmente, o processo pode ser descrito como:

Geração aleatória de uma solução inicial T.

i = 1;

Enquanto (**a troca de elementos de X por elementos de Y produzir uma melhoria na função de custo**)

- Selecione o par (x_i, y_i) escolhido de maneira a maximizar a melhora alcançada na troca de $x_1, \ldots, x_i$ por $y_1, \ldots, y_i$;
- $x_i \in T - \{x_1, \ldots, x_{i-1}\}$ e $y_i \in S - T - \{y_1, \ldots, y_{i-1}\}$;
- $i = i + 1$;

k = i;

Troque $x_1, \ldots, x_k$ por $y_1, \ldots, y_k$, gerando T'.

Repita todo o processo acima a partir de T'.

Caso não se tenha mais melhorias, repita todo o processamento a partir de uma nova solução gerada aleatoriamente, se desejado.

Algumas restrições, regras e estratégias de controle precisam ser especificadas a fim de tornar o algoritmo mais eficiente. É preciso que se defina:

1. Uma *Regra de Seleção* eficiente — a escolha dos elementos a serem trocados entre os conjuntos deve ser feita de uma maneira eficiente e rápida. A determinação de uma ordem na escolha dos elementos é importante, principalmente para evitar a geração de soluções não viáveis para o problema, disperdiçando conseqüentemente tempo de computação.

2. Uma *função* que represente o *ganho total*, dado um conjunto de trocas. Para representar o ganho obtido dada a troca de dois elementos, define-se g_i como o ganho associado à troca do elemento x_i por y_i na iteração i, dado que $x_1, y_1, \ldots, x_{i-1}, y_{i-1}$ já foram trocados.

Assim, quando $\sum_{i=1}^{k} g_i \leq 0$, para um determinado k, o processo de troca de elementos é interrompido, pois a troca de uma seqüência maior de elementos não será mais lucrativa.

3. Uma vez terminado o processo de seleção, é preciso verificar se a troca de $x_1, \ldots, x_k$ por $y_1, \ldots, y_k$, para um determinado k, é *viável*. Essa verificação é importante para que se previna a possibilidade de gerar, após a troca de uma sequência grande de elementos, uma configuração não viável para o problema, desperdiçando tempo de computação.

4. Uma *Regra de Parada* indicando que o conjunto de possíveis elementos a serem trocados já foi totalmente examinado ou que as trocas efetuadas já não produzem mais ganhos.

5. $\{x_1, \ldots, x_k\} \cap \{y_1, \ldots, y_k\} = \emptyset$. Se essa condição for satisfeita, a seqüência de elementos selecionada para troca será conservada durante toda uma iteração, ou melhor, nenhuma das arestas já selecionada será novamente marcada

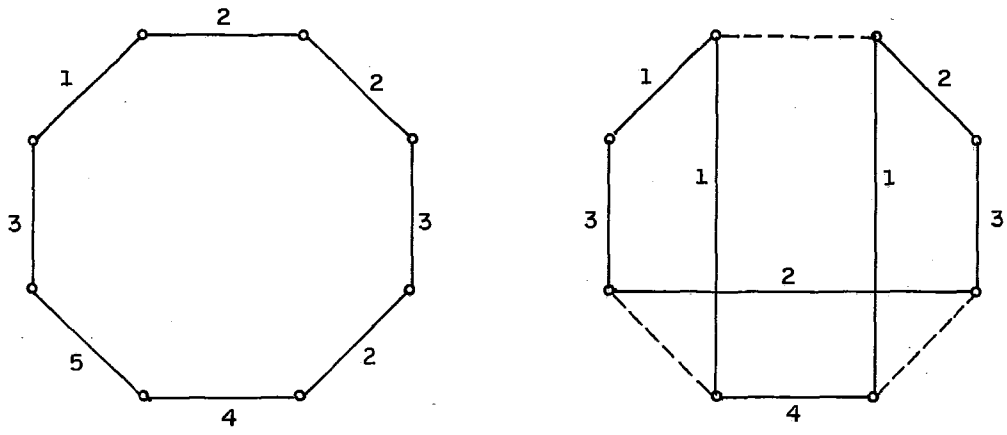


Figura IV.1: T e T'

Essa medida é importante pois assim, a seqüência de elementos que produzem uma melhoria na função de custo não é danificada, reduzindo tempo de computação, simplificando a função de ganho e possibilitando a definição de uma regra de parada eficiente.

As trocas são efetivadas, gerando novas soluções usadas recursivamente como pontos de partida para o processo descrito acima, até o momento em que o ganho obtido com a troca de elementos não seja mais positivo, indicando que mais nenhuma melhora pode ser alcançada pelo algoritmo.

IV.3 O algoritmo na solução do problema do Caixeiro Viajante

Nesta seção, será descrita a aplicação do algoritmo de Lin & Kernighan ao problema do Caixeiro Viajante. Para isso, é preciso que se considere o conjunto S de todas as arestas de um grafo completo de N vértices, T como um subconjunto de S que corresponde a uma tour e $f(T)$ a função que representa o comprimento da tour T.

Seja T' uma tour com comprimento $f(T') < f(T)$, como na figura IV.1.

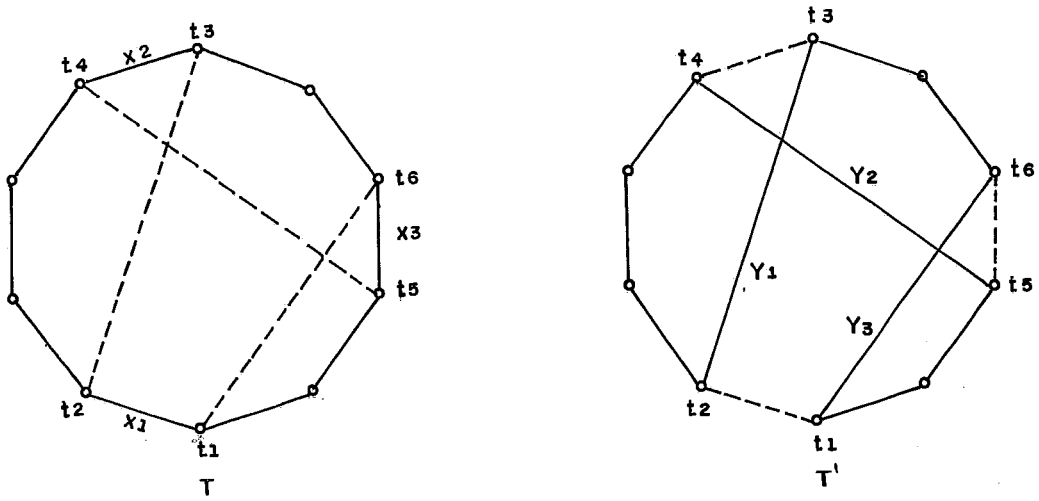


Figura IV.2: Transformação de T em T'

A existência de 3 arestas distintas entre T e T' responde pela diferença entre seus comprimentos. O algoritmo tenta transformar T em T', identificando sequencialmente as k arestas em T que podem ser trocadas pelas k arestas em T'. Em outras palavras, identifica-se o conjunto de arestas $X = \{x_1, \ldots, x_k\}$ em T e $Y = \{y_1, \ldots, y_k\}$ em T' de tal maneira que as arestas de X são substituídas pelas arestas de Y. O resultado é uma tour de custo menor, igual a $f(T')$. A diferença entre os comprimentos de T e T' é justificada pela existência de arestas distintas entre as duas tours. Se considerarmos $\mid x_i \mid$ e $\mid y_i \mid$ como os comprimentos das arestas x_i e y_i e $g_i = \mid x_i \mid - \mid y_i \mid$ como o ganho obtido ao trocar x_i por y_i, é imediato concluir que $\sum_1^k g_i = f(T) - f(T')$. Como $f(T') < f(T)$, então $\sum_1^k g_i > 0$. É possível, no entanto, que ocorra alguma troca cujo ganho g_i, para algum i, seja negativo ($\mid y_i \mid > \mid x_i \mid$). Essa troca é considerada válida, desde que a soma total dos ganhos obtidos com todas as trocas seja positiva, indicando que um tour de menor custo foi gerada.

As arestas a serem trocadas são identificadas e numeradas sequencialmente. As arestas x_i e y_i compartilham um mesmo nó, assim como y_i e x_{i+1}. A aresta x_1 será trocada por y_1, x_2 por y_2, e assim sucessivamente. A figura IV.2 ilustra como as trocas de arestas são realizadas quando $k = 3$.

É preciso também que se defina o parâmetro G^* que representará a melhoria realizada sobre a tour T, a cada conjunto de trocas. G^* é inicializado com 0 e assume valores positivos e monotonicamente crescentes durante a evolução do algoritmo. A cada valor de G^* é determinado também, um valor para k que indica o número de arestas a serem trocadas para alcançar G^*.

Tendo em mente todos esses conceitos, será exposto, a partir de agora, o algoritmo em todos os seus detalhes.

Inicializações

Geração aleatória de uma tour inicial T;
$G^* = 0$;
Escolha aleatória de t_1 entre os N vértices da tour;
Escolha de uma das arestas adjacentes a t_1: $x_1 = (t_1, t_2)$;
$i = 1$;
Escolha da aresta $y_1 = (t_2, t_3)$ com $g_1 =\mid x_1 \mid - \mid y_1 \mid > 0$;
$G_1 = g_1$;

A aresta y_1 é escolhida dentre as arestas de $S - T$ que partem de t_2 e chegam a um dos $N - 3$ vértices de T não-adjacentes a t_1. A aresta escolhida será aquela que produzir o melhor ganho dentre todas, ou seja, será aquela de menor custo. Caso não exista tal aresta, retoma-se o processo a partir da outra aresta adjacente a t_1 em T, $x_1 = (t_1, t_2')$. Caso não se obtenha sucesso novamente, retorna-se ao nível anterior, escolhendo outro nó t_1 como ponto de partida para o algoritmo.

Processo de Seleção de Arestas

Enquanto ((**todos os candidatos a arestas** x_i **e** y_i **não tiverem sido examinados**)
e $(G_i > G^*)$)
$i = i + 1$;
Escolha $x_i = (t_{2i-1}, t_{2i})$ tal que:
A sequência $(t_1, \ldots, t_{2i}, t_1)$ forme uma tour; (Criterio de Viabilidade)

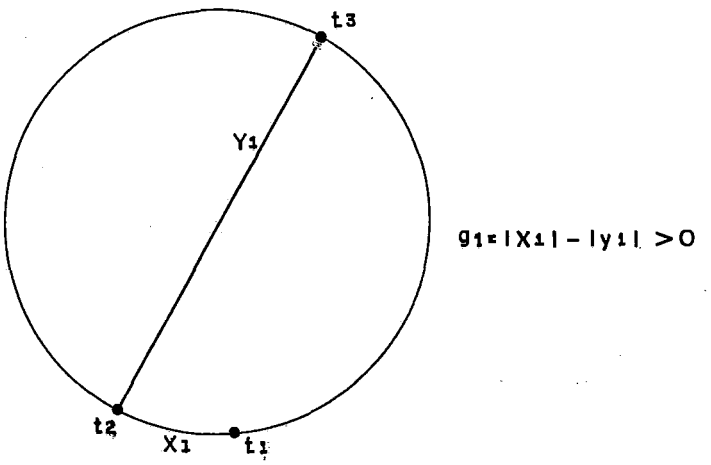


Figura IV.3: Passo inicial do algoritmo

Escolha $y_i = (t_{2i}, t_{2i+1})$ tal que:

- $\{x_1, \ldots, x_i\} \cap \{y_1, \ldots, y_i\} = \emptyset$;
- $G_i = \sum_1^i g_j > 0$;
- permita a escolha de uma aresta x_{i+1} que satisfaça o Criterio de Viabilidade no passo $i + 1$;

$y_i^* = (t_{2i}, t_1)$;
$g_i^* = \mid x_i \mid - \mid y_i^* \mid$;
Se $G_{i-1} + g_i^* > G^*$ então
$G^* = G_{i-1} + g_i^*$;
$k = i$;

Se $G^* > 0$ então

Considere a tour T' formado apos as k trocas de arestas;
$f(T') = f(T) - G^*$;
Repita todo o processo usando T' como tour inicial;

Se $G^* = 0$ então

efetue BACKTRACKING;

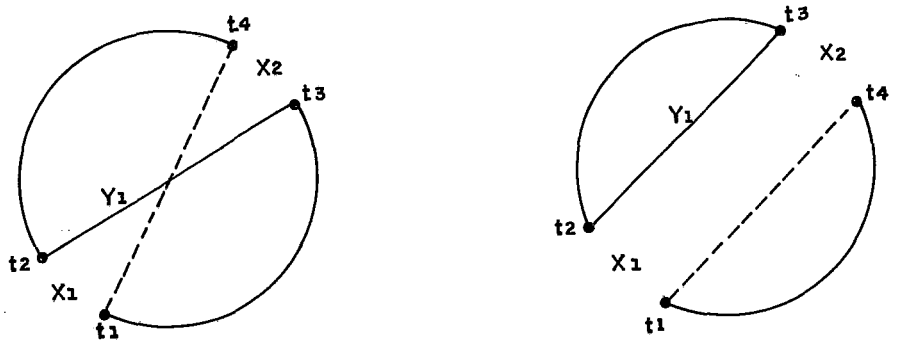


Figura IV.4: (a)Forma uma tour (b)Não forma uma tour

Critério de Parada

O algoritmo termina quando todos os vértices da tour são escolhidos como ponto de partida para o processo de seleção de arestas, sem que se obtenha uma tour de menor custo.

A escolha da aresta x_i é unicamente determinada pela aresta y_{i-1}, selecionada no passo anterior. Das duas candidatas a aresta x_i, apenas uma possibilitará a formação de uma tour. O Critério de Viabilidade garantirá a escolha dessa aresta. A aplicação desse critério é importante para que se possa evitar esforços computacionais desnecessários, como o de submeter o algoritmo a um processo de seleção de arestas que não conduzem a uma tour.

Dentre todas as arestas de $S-T$, possíveis candidatas a y_i, é escolhida aquela que:

• Já não tenha sido escolhida em passos anteriores do processo de seleção, evitando assim que a seqüência de trocas seja alterada e alguma aresta eventualmente escolhida anteriormente para ser retirada, seja agora selecionada como uma das novas componentes da tour.

• Mantenha o ganho — referente as trocas até aquele momento —

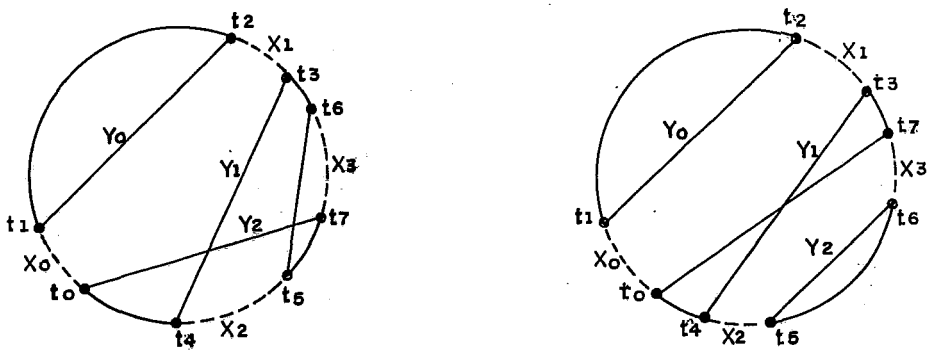


Figura IV.5: Justificativa da terceira condição para a escolha da aresta y_i

positivo, indicando ser vantajosa para a formação de uma tour de menor custo a troca das arestas selecionadas até então.

• Assegure que o Critério de Viabilidade será garantido no passo $i+1$. Nem toda aresta que atenda as condições acima garante a formação de uma tour, como mostra a figura IV.5.

Antes da inserção da aresta y_i ao conjunto Y, verifica-se se a tour $(t_1, \ldots, t_{2i}, t_1)$ produzirá um ganho melhor que G^*. Em caso afirmativo, o valor de G^* é atualizado. É importante ressaltar que G^* representa a melhoria alcançada levando-se em conta a tour formado após as trocas de arestas. À cada passo do processo de seleção, é verificado se a nova tour formada produz um ganho melhor que o anterior. Desta maneira, a última tour formada com a mudança das k arestas será sempre aquela que produzirá a melhor redução no custo computado.

É possível que G^* não seja atualizado por muitas iterações e o conjunto de arestas a serem trocadas continue a ser atualizado. Isso acontece porque o que se compara com o último valor de G^* é o comprimento da tour formada até a escolha de x_i. Se o ganho obtido com a formação dessa tour não for melhor que G^* então G^* não é atualizado. Em compensação, se a escolha de y_i acarretar num ganho G_i maior que G^*, então o processo de seleção continua.

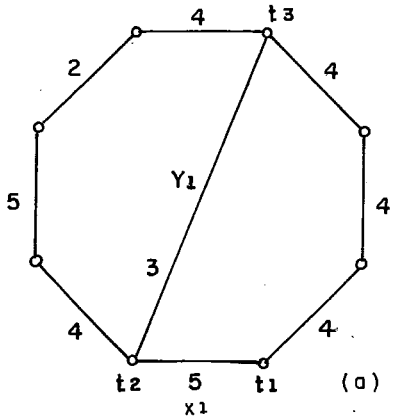
A seguir será descrito um exemplo que mostra em detalhes como funciona o algoritmo.

Seja a tour T cujas arestas possuem custos especificados como na figura IV.6 e e $x_1 = (t_1, t_2)$ uma aresta de T. y_1 foi a aresta escolhida em $S - T$ para ser trocada por x_1. Assim, já sabemos que $g_1 =\mid x_1 \mid - \mid y_1 \mid > 0$. Caso não existisse tal aresta, seria preciso que o Backtracking fosse efetuado, escolhendo-se a outra aresta x_1 adjacente a t_1. Uma vez escolhido o primeiro par de arestas, passamos para a próxima iteração, onde será escolhido o próximo par a ser trocado. É escolhida então a aresta x_2 em T que possibilite a formação de uma tour. Daí, antes de selecionar a aresta y_2, verifica-se se o ganho obtido com a tour formada é maior do que o ganho G^* computado até então. Para prosseguir com o processo de seleção de arestas, é preciso que o ganho computado até então com as trocas seja maior que o ganho total, G^*, caso contrário, a última troca não valerá mais a pena. Nesse momento então, todos os pares de arestas selecionados para serem trocados até essa iteração são efetivamente trocados e uma nova tour T', de menor custo é formada. Todo esse processo é novamente aplicado sobre T', na busca de uma nova tour de menor custo, até que não se consiga mais encontrar tais tours. Esse momento indica que um mínimo local foi encontrado e este mínimo representa a menor tour encontrada pelo algoritmo para o problema.

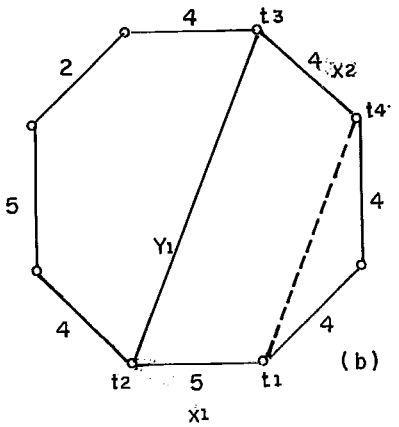
O BACKTRACKING, citado no algoritmo, só é efetuado nos dois primeiros níveis do processo de seleção, ou seja, só se tem a possibilidade de escolher novas candidatas para as arestas x_1, y_1, x_2 e y_2. O BACKTRACKING é efetuado por passos, como descrito a seguir.

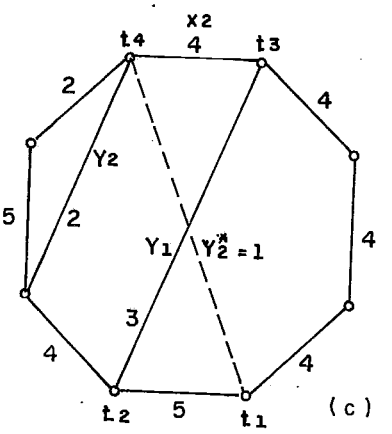
Quando se chega ao ponto de trocas em que G^* é igual a zero, o BACKTRACKING é efetuado, ignorando-se todas as arestas selecionadas no processo de trocas até então e retornando-se ao ponto de escolha da aresta y_2. Assim, os passos para a realização do BACKTRACKING são:

1. Repita o processo de seleção de arestas considerando agora uma nova aresta y_2 de maior peso, até que $g_1 + g_2 > 0$. A partir daí, continue o processo normalmente.(BACKTRACKING 1)



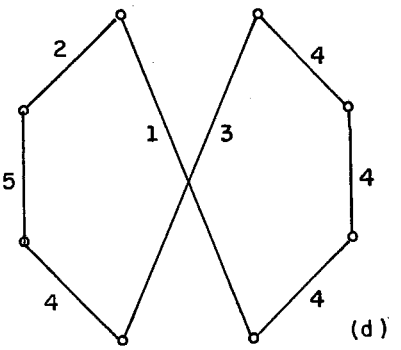
T = 32
G* = 0





G* = 2 + 3 = 5
G2 = 4 < 5

NÃO FORMA UMA TOUR



T' = 27

Figura IV.6: Um exemplo

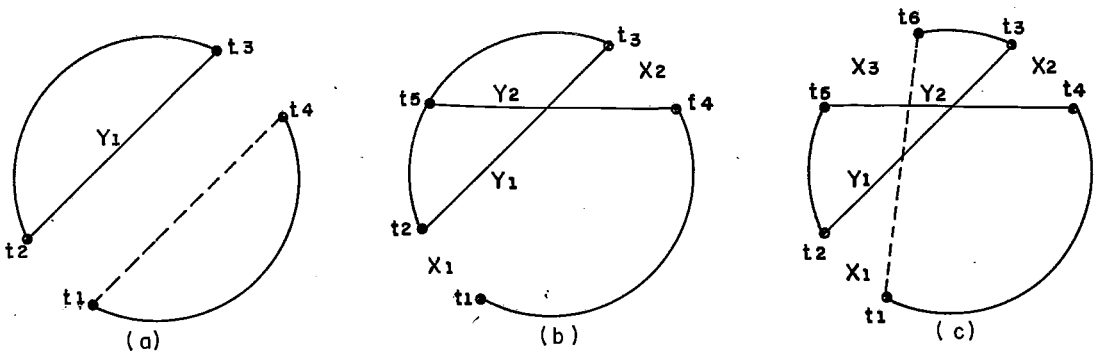


Figura IV.7: (a) Não forma uma *tour* (b) A violação do Critério de Viabilidade (c) Forma uma *tour*

2. Se todas as escolhas para y_2 não causarem sucesso, retroceda mais profundamente no Backtracking, escolhendo agora a outra candidata à aresta x_2. Já sabemos que essa aresta violará o critério de viabilidade. Mas tal violação é permitida nesse momento pois ocasionará posteriormente a volta ao curso normal do processamento do algoritmo, como mostra a figura IV.7 , possibilitando o encontro oportuno de um tour de menor custo. (BACKTRACKING 2)

3. Se mesmo assim nenhum ganho for produzido, escolha agora uma aresta y_1 de comprimento maior e retome todo o processo novamente. (BACKTRACKING 3)

4. Se todos os candidatos a y_1 são examinados sem produzir ganho, tente a outra candidata a aresta x_1.(BACKTRACKING 4)

5. Caso todos os items acima falharem, um novo nó t_1 é selecionado e todo o processo é reiniciado a partir desse novo nó.

Depois que todo o algoritmo foi exposto em todos os seus detalhes e peculiaridades, é interessante notar que o processo de BACKTRACKING é bastante custoso e complicado. Ele se processa no pior caso, ou seja, no caso em que todos

os seus níveis são efetuados, da seguinte maneira.

```
BACKTRACKING()
{
        Se (existe outro candidato a aresta y2)
            BACKTRACKING1;
        senão
        {
            Se (a outra candidata a x2 não foi escolhida)
                BACKTRACKING2;
            senão
            {
                Se (existe outro candidato a aresta y1)
                    BACKTRACKING3;
                senão
                {
                    Se (a outra candidata a x1 não foi escolhida)
                        BACKTRACKING4;
                }
            }
        }
}
```

É interessante notar também que o BACKTRACKING poderia ser realizado em todos os níveis do processo de seleção. No entanto, isso ocasionaria um tempo de processamento absurdo e não traria resultados tão vantajosos a ponto de justificar tal perda de tempo, como relata o artigo [19]. Experiências citadas nesse artigo mostram que os melhores resultados obtidos pelo algoritmo foram obtidos com o BACKTRACKING restrito aos dois primeiros níveis de seleção de arestas.

Capítulo V

Avaliação dos resultados

V.1 Introdução

Apresentaremos nesse capítulo, uma análise dos resultados obtidos a partir de testes realizados com os três algoritmos implementados neste trabalho, utilizando como instâncias do problema do Caixeiro Viajante, conjuntos de cidades geradas aleatoriamente.

A variação de valores para o conjunto de parâmetros usado tanto no Algoritmo Elástico quanto no Filtro, foi a base dos testes, realizados em busca de uma *tour* próxima do ótimo.

V.2 A implementação

V.2.1 Utilização de memória

Uma das preocupações na implementação de algoritmos elaborados para a solução do Problema do Caixeiro Viajante é a utilização mínima de memória de maneira a não comprometer a quantidade total de tempo de processamento. Tal minimização é importante, pois como esse problema é NP-completo, a utilização de instâncias de grande porte possibilita uma melhor avaliação da eficiência dos algoritmos, necessitando-se nestes casos, de um maior espaço de memória para a manipulação dos dados.

A quantidade de memória utilizada pelo Algoritmo Elástico torna-se um problema quando o número de cidades é grande. Em função disso, na sua implementação, foram introduzidos meios alternativos de estruturação do armazenamento dos dados necessários ao processamento do algoritmo.

O cálculo do deslocamento de cada ponto do anel depende de um coeficiente que mede as influências exercidas por cada cidade sobre cada ponto. Deste modo, a cada iteração do algoritmo, esse coeficiente é atualizado $N \times M$ vezes, sendo N o número de cidades e M o número de pontos do anel. Desta maneira, seria necessário o armazenamento desses valores numa matriz de dimensão $N \times M$, uma vez que, para o cálculo de cada coeficiente, é necessário que se tenha armazenado os valores das influências de cada cidade i a todos os pontos do anel.

Com esse tipo de armazenamento, instâncias grandes do problema não poderiam ser avaliadas, pois não existiria memória suficiente.

Com o objetivo de remediar essa situação, tornou-se necessária a especificação de tipos de estruturas que permitissem uma economia de memória, mas que não penalizassem o tempo de processamento do algoritmo. Sendo assim, foi apenas necessário definir um vetor contendo as coordenadas das N cidades distribuídas sobre o quadrado, uma lista encadeada com as posições de todos os pontos situados sobre o anel e um vetor , também de dimensão N, contendo em cada uma de suas posições, o somatório das influências da cidade i sobre todos os pontos. Esse dado é utilizado para a normalização do coeficiente w_{ij}. Desta forma , a memória utilizada pelo Algoritmo Elástico é da ordem de $\mathcal{O}(N + M)$.

Para o Filtro, proposto neste trabalho, foi preciso que se definisse mais uma estrutura. Agora, o deslocamento de cada ponto não é realizado em função das influências de todas as cidades, e sim daquelas que contribuem significativamente para o deslocamento. Tais cidades encontram-se no interior de um cículo, determinado no Filtro, cujo centro coincide com o ponto. A fim de facilitar a implementação, considera-se a subdivisão do quadrado de lado L em quadrados menores, de lado δ, correspondente à distância média das cidades. Os cálculos realizados são apenas aqueles referentes às cidades situadas no interior dos quadrados $\delta \times \delta$ que

circunscrevem o círculo.

Para armazenar esses dados, foi preciso definir dois vetores de listas encadeadas, ambos de dimensão Q, correspondente ao número de quadrados $\delta \times \delta$ existentes. Cada vetor contém respectivamente em cada uma de suas posições, a lista das cidades e dos pontos que se encontram no quadrado q, $1 \leq q \leq Q$. Sendo assim, a memória utilizada pelo Filtro é da ordem de $\mathcal{O}((N + M)(1 + Q))$, no pior caso.

V.3 Avaliação do comprimento da *tour* gerada pelo algoritmo

Em todos os trabalhos relacionados ao Algoritmo Elástico, a *tour* gerada pelo algoritmo é entendida como o comprimento do anel, cujos pontos são mapeados nas cidades.

Neste trabalho, a concepção da *tour* é um pouco diferente. Considera-se a convergência do algoritmo quando, para cada uma das cidades, existir pelo menos um ponto do anel em sua **vizinhança**, sendo associada a cada cidade i o ponto que estiver mais próximo dela. Desta forma, a avaliação do comprimento da *tour*, considerada apenas como a soma das distâncias entre os pontos do anel, não resultaria numa medida real.

O fato de se comparar, neste trabalho, o desempenho do Algoritmo Elástico e do Filtro com um algoritmo que realiza melhorias no custo da *tour* a partir das próprias cidades — o algoritmo de Lin & Kernighan (LK) — reforça o intuito de não se estabelecer o comprimento do anel como o custo da *tour*. Dentre os pontos que se encontram na **vizinhança** de cada cidade, considera-se a seqüência daqueles que estiverem mais próximos delas. O custo da *tour* é então aquele equivalente à permutação das cidades associada à essa seqüência.

V.4 Apresentação e análise dos resultados

Com o objetivo de explorar a potencialidade do Algoritmo Elástico e analisar suas características de acordo com o crescimento do número de cidades, foram realizados testes com conjuntos de 100, 200, 300, 400, 500, 750 e 1000 cidades.

A escolha das estações de trabalho SUN para efetuar tais testes foi realizada basicamente devido a sua grande velocidade de processamento e extensa capacidade de memória, necessárias principalmente para os testes com 1000 cidades.

Nesta seção, apresentaremos tabelas com os resultados dos testes que convergiram para cada conjunto de cidades, para que seja possível a interpretação e análise do efeito de cada parâmetro na equação de deslocamento de cada ponto do anel, apresentada novamente em V.1 a fim de facilitar tal análise.

$$\Delta Y_j = \alpha \sum_i w_{ij}(X_i - Y_j) + \beta K(Y_{j+1} - 2Y_j + Y_{j-1}) \qquad \text{(V.1)}$$

Se observarmos cada parcela P_i, $1 \leq i \leq N$, do somatório do primeiro termo da equação V.1 e considerarmos a definição do coeficiente w_{ij} apresentada no capítulo II, podemos concluir que as parcelas que contribuem com valores altos para o somatório e consequentemente para um maior deslocamento do ponto, são aquelas referentes às cidades que estiverem mais próximas dele.

O parâmetro α é responsável pelo "controle" da maior ou menor atuação do primeiro termo no deslocamento. Valores de α próximos de 1 contribuem para uma atração muito grande do ponto pelas cidades uma vez que praticamente todo o somatório é considerado, induzindo ao ponto um deslocamento muito brusco e contribuindo para a não convergência do algoritmo. Estipular valores para α maiores que 0.8 provocam situações como a da figura V.1.

Todos os conjuntos de cidades foram gerados aleatoriamente dentro de um quadrado de mesmo tamanho. Desta forma, quanto maior for o tamanho do conjunto, maior também será a densidade de cidades no interior do quadrado, existindo nestes casos muito mais cidades próximas de cada ponto do anel. Con-

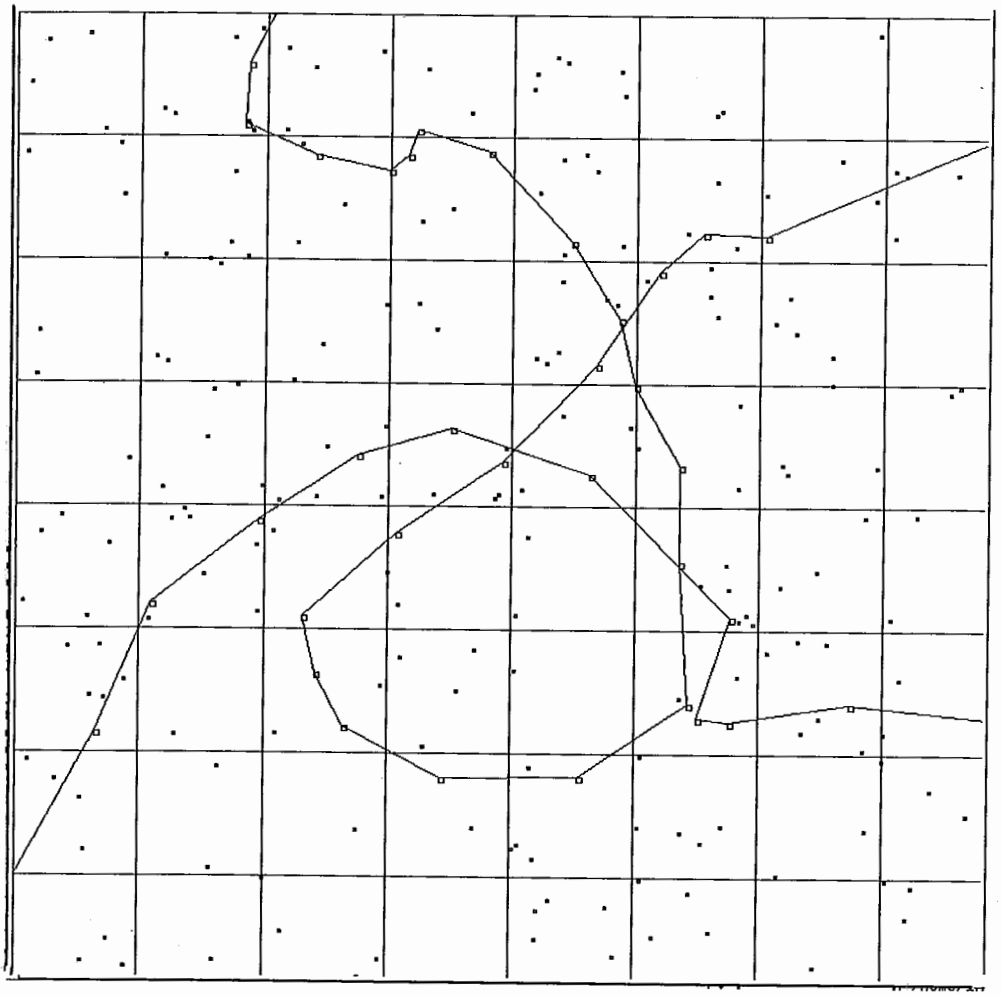


Figura V.1: O anel não convergiu

seqüentemente muito mais parcelas do somatório do primeiro termo de V.1 contribuirão com maior peso para o deslocamento, causando com maior facilidade situações como a exibida na referida figura.

No sentido de proporcionar aos pontos deslocamentos mais amenos em direção às cidades, seria mais prudente considerar, a medida do crescimento do número de cidades, valores cada vez menores para α.

É preciso no entanto, que se tome cuidado para não estimar valores para α extremamente pequenos pois isto também poderia ocasionar a não convergência do algoritmo. A influência das cidades sobre o ponto decresce a medida que o valor de K se aproxima de 0, sendo aproximadamente 1 apenas a influência

daquela cidade que estiver bem próxima dele. Por esse motivo, neste estágio do processamento, o primeiro termo de V.1 influencia muito pouco no deslocamento (praticamente apenas uma parcela do somatório contribuirá com um valor alto). A atração exercida por esse termo sobre o ponto torna-se menor ainda pelo estabelecimento de um valor muito pequeno para α. Em muitos casos, neste momento do processamento, nem todos os pontos estão suficientemente próximos das cidades para que o algoritmo convirja, como o que aconteceu com os testes realizados, por exemplo, com os seguintes conjuntos de parâmetros

N	α	β	$K_{inicial}$	TRK	ε
750	0.1	3.0	0.24	5%	0.03
1000	0.06	1.0	0.24	2%	0.02

A situação descrita acima pode ser ilustrada pela figura V.2, que representa a configuração do sistema para 750 cidades, quando $K = 0.000056$. Nota-se que existem ainda várias cidades que não foram atingidas por nenhum ponto do anel e neste estágio do processamento, os deslocamentos sofridos pelos pontos em direção às cidades são praticamente desprezíveis.

Para densidades grandes de cidades, o melhor procedimento seria então estabelecer valores para α que não permanecessem constantes ao longo do processamento do algoritmo. Como nas primeiras iterações o valor de K ainda é alto, a contribuição do primeiro termo da equação para o deslocamento é muito grande, sendo necessário o estabelecimento de um valor pequeno para α, não muito adequado, no entanto, no final do processamento, onde a contribuição do primeiro termo é ínfima, não produzindo praticamente nenhuma alteração da posição do ponto em direção da cidade à qual ele será associado.

Seria interessante definir α inversamente proporcional a K. É preciso, no entanto, que essa definição seja cuidadosamente especificada, para que não se altere o comportamento do algoritmo no final do processamento, pois a constante diminuição de K contribui para amenizar os deslocamentos dos pontos, quando estes se encontram mais próximos das cidades. O estabelecimento inadequado de α em função de K poderia causar também a não convergência do algoritmo.

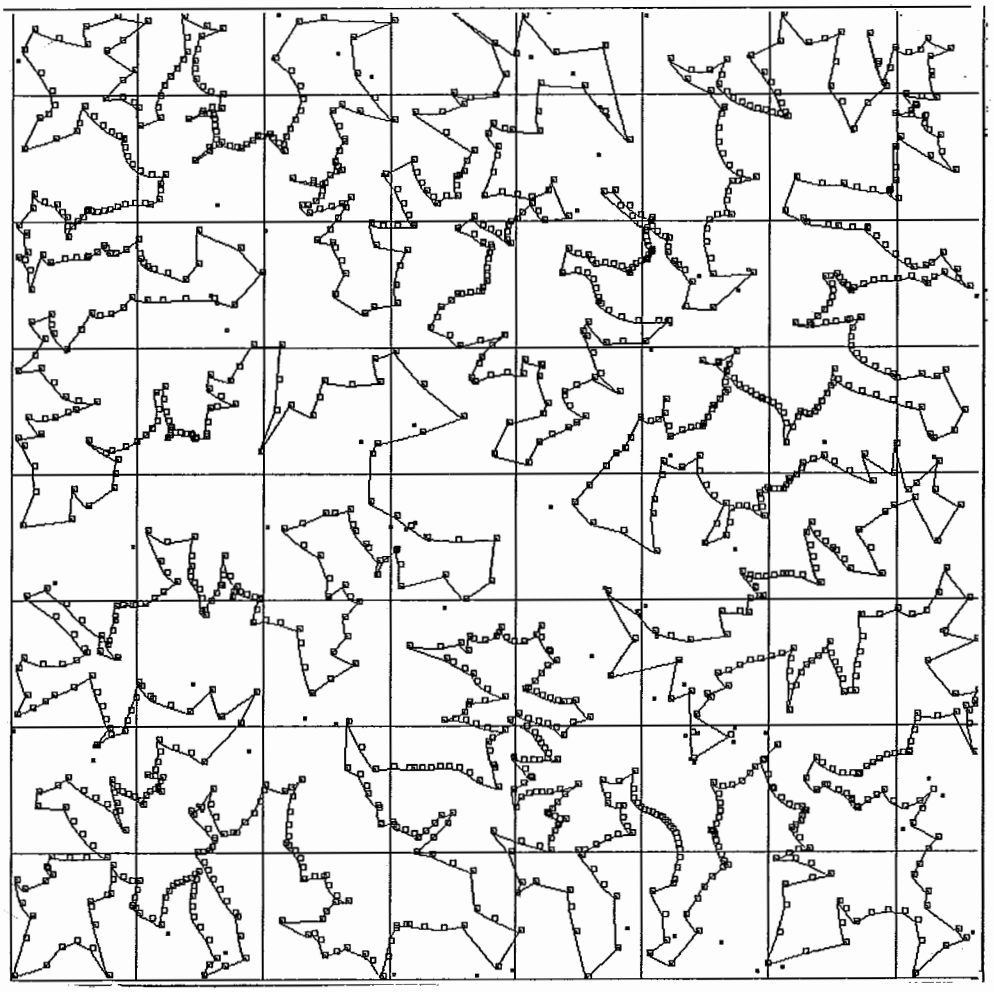


Figura V.2: O anel não convergiu para 750 cidades

Para um valor de α constante, observou-se a diminuição do custo da *tour* de acordo com o aumento do valor de β. O estabelecimento de um valor alto para β induz um "enrijecimento" do anel, deformando-o em direção das cidades, de maneira lenta e gradual.

Entretanto, quando esse valor é muito grande, o coeficiente βK aumenta muito também, principalmente no início do algoritmo, quando o valor de K é alto, causando um efeito contrário ao desejado. A força atribuída ao ponto em direção aos seus vizinhos é muitas vezes maior que a própria distância entre eles, forçando-o a um deslocamento brusco e causando um comportamento semelhante àquele da figura V.1. Nos testes, valores de β maiores que 4.0 provocaram tal situação.

Por sua vez, o crescimento de α pode induzir o crescimento do comprimento da *tour*. O estabelecimento de um valor alto para α pode provocar deslocamentos grandes dos pontos, acarretando a geração, pelo algoritmo, de uma *tour* de alto custo.

Outro fator importante para a determinação do comprimento da *tour* é o número de pontos que se encontram, na primeira iteração, sobre uma circunferência de raio 0.1, valor este estabelecido em todos os testes realizados. Uma vez que as influências das cidades sobre os pontos são normalizadas pelo número total de pontos existentes no anel, quanto maior for o número de pontos, maior será o denominador da razão que define o coeficiente w_{ij}. Em alguns testes verificou-se uma diminuição no comprimento da *tour*, apenas com o aumento do número de pontos iniciais, mantendo-se, no entanto, o mesmo conjunto de parâmetros.

Com o estabelecimento de mais pontos sobre o anel, cada cidade exerce uma força menor sobre cada ponto (sua influência foi dividida por mais elementos) e consequentemente contribui para um deslocamento menor, equivalendo ao fato de se estabelecer para β um valor maior, "enrijecendo" a deformação do anel. Deste modo, para um mesmo conjunto de parâmetros, podemos obter uma *tour* de melhor custo, como mostra a tabela abaixo, se aumentarmos o número inicial de pontos, pois a deformação do anel tende a ser menos brusca.

200 cidades					
$M_{inicial}$	α	β	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
52	0.8	3.0	0.24	475	14.202735
104	0.8	3.0	0.24	522	12.290537

Se o valor de β já é alto, esse procedimento pode aumentar o comprimento da *tour*, em vez de diminuir. O fato é que o aumento de pontos provoca uma redução das suas relativas distâncias. Como β é grande, a força de atração exercida sobre cada ponto por seus vizinhos o "empurra" em direção oposta a estes, provocando o alargamento do anel, como já discutido anteriormente.

Quando o número de cidades distribuídas pelo quadrado é muito pequeno, também é pequena a força de atração exercida por estas. Desta forma, ao atribuir um valor grande para β, o anel,ao invés de se deformar gradualmente rumo às cidades, se "encolhe", pois a força de atração entre os pontos é tão grande que eles se aproximam todos entre si. A medida que K decresce, o valor do coeficiente βK fica cada vez menor, relaxando a força de encolhimento do anel e consequentemente favorecendo a força de atração das cidades, que se torna mais significativa. Com o aumento da importância dessa força, o anel volta a se deformar em direção das cidades, rumo à convergência do algoritmo.

Essa situação pode ser observada na figura V.3.

Na maioria dos testes realizados adotamos uma taxa de redução para K de 5%, que permite um decréscimo mais rápido sendo menor a quantidade de valores altos de K que contribuem para um maior deslocamento dos pontos. No entanto, para os conjuntos de 750 e 1000 cidades, foi necessária para certos conjuntos de parâmetros, a realização de testes com o Filtro, para uma taxa de redução de K de 2%. Foram realizadas 2 sub-iterações para cada valor de K em todos os testes, pelo fato de não se alcançar, para um maior número de sub-iterações, melhorias significativas no custo da *tour*, aumentando-se apenas o tempo de processamento.

Discutiu-se até agora todas as características observadas no comportamento, tanto do Algoritmo Elástico quanto do Filtro, na maioria dos testes realizados para todos os conjuntos de cidades, no sentido de se encontrar o conjunto de parâmetros que permitisse a geração de uma *tour* de menor comprimento possível

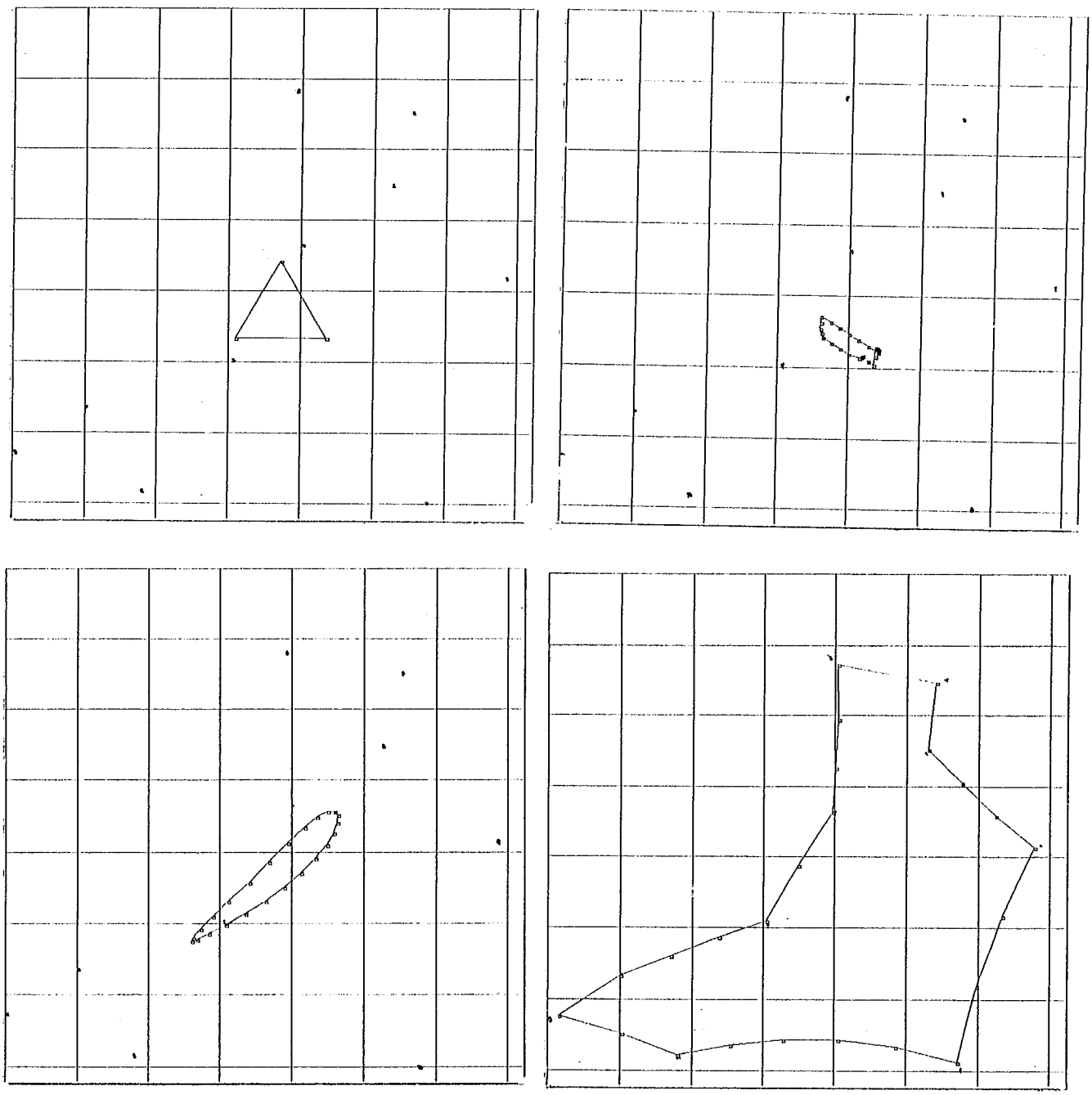


Figura V.3: A evolução da deformação do anel para 10 cidades

A seguir serão apresentadas as tabelas com todos os testes que convergiram para o Algoritmo Elástico (AE), Filtro (F) e o Algoritmo de Lin & Kernighan (LK). Oportunamente, serão feitas algumas observações durante a análise de cada conjunto de cidades.

(AE) - 100 cidades $-\ \varepsilon = 0.05$				
α	β	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.2	1.0	0.24	130	8.475682
0.2	2.0	0.24	147	8.460747
0.2	3.0	0.24	150	8.480978
0.2	4.0	0.24	146	8.194427
0.4	1.0	0.24	127	8.914408
0.4	2.0	0.24	136	8.760423
0.4	3.0	0.24	144	8.529709
0.6	1.0	0.24	146	10.07593
0.6	2.0	0.24	127	8.920962
0.8	1.0	0.24	119	9.685364
0.8	2.0	0.24	118	8.958533
0.8	3.0	0.24	112	8.534197

O conjunto de parâmetros testados para 100 cidades foi o mesmo para os conjuntos de 200, 300, 400 e 500. Para 750 e 1000 cidades não foi mantida essa uniformidade de testes, uma vez que a medida que a densidade de cidades aumenta, diminuem os casos, dentre esses parâmetros, que convergem. Por esse motivo, para esses conjuntos, foram escolhidos parâmetros que parecessem mais adequados para atingir a convergência do algoritmo e fornecessem os menores comprimentos de *tour* possíveis.

Observa-se na tabela acima que para valores de β maiores que 4.0, o algoritmo não convergiu, exceto para α igual a 0.2. Uma vez que a densidade de cidades e o número de pontos iniciais distribuídos sobre a circunferência são pequenos, as distâncias entre os pontos tornam-se mais significativas, contribuindo para a atração destes pelos seus vizinhos e possibilitando o "enrijecimento" da deformação da *tour*. Com o aumento do valor de α, o primeiro termo do deslocamento torna-se muito grande e o número de pontos do anel não é grande o suficientemente para que as parcelas relativas as influências diminuissem (em função da normalização de w_{ij}), permitindo um menor deslocamento do ponto.

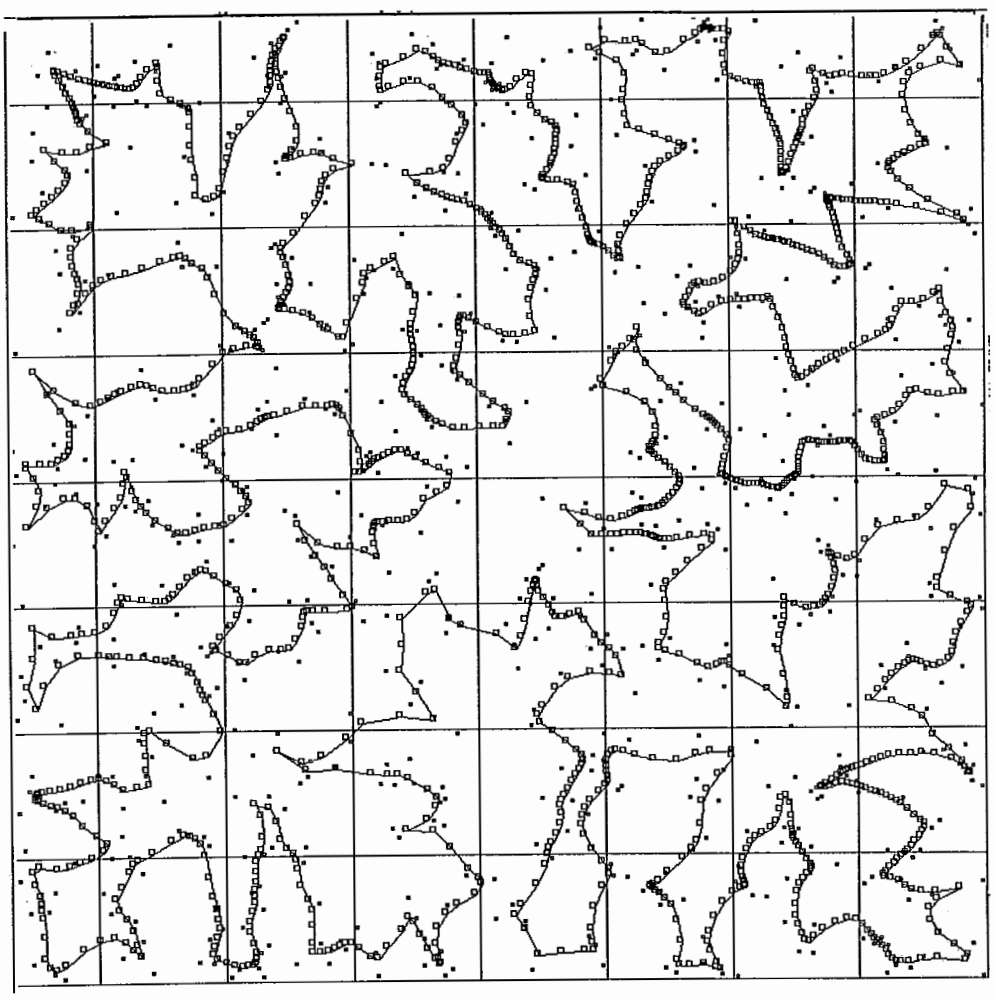


Figura V.4: Uma tour no momento da convergência

É importante relembrar aqui o critério de convergência usado no Algoritmo Elástico. É preciso que para cada uma das cidades, exista pelo menos um ponto do anel em sua vizinhança. Um ponto será associado à cidade i se ele for o mais próximo dentre todos aqueles que estiverem na vizinhança de i. Para os testes até 500 cidades, o raio da vizinhança (denotada por ε) foi de 0.05.

A definição de vizinhanças para as cidades é interessante, uma vez que quando os pontos se encontram à uma distância relativamente pequena das cidades já se sabe qual ponto será associado a que cidade e assim não é preciso gastar mais tempo de processamento. Entretanto, existem situações em que mesmo com uma pequena densidade de cidades, podem existir grupos em que elas estejam

muito próximas umas das outras. Nestes casos, pode ocorrer a associação de um ponto à duas cidades distintas, que estejam muito próximas, pois suas vizinhanças se interceptam. A escolha de qual cidade ele será associado pode acarretar a escolha de uma permutação de cidades que gere uma *tour* de maior comprimento. Nesses casos, a escolha de vizinhanças menores pode eventualmente sanar esse problema, uma vez que a probabilidade de mais de um ponto estar situado na vizinhança de cada cidade é muito pequena.

Por esse motivo, foram feitos novos testes, com todos os conjuntos de cidades considerando vizinhanças de raio menor para os conjuntos de parâmetros que geraram as melhores *tours*, tanto para o Algoritmo Elástico quanto para o Filtro. Para 100 cidades obtemos

100 cidades						
	ε	α	β	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
AE	0.02	0.2	4.0	0.24	187	8.190472
	0.05	0.2	4.0	0.24	146	8.194427
F	0.02	0.2	3.0	0.24	110	8.45
	0.05	0.2	3.0	0.24	85	8.470696

Podemos observar que não houve uma melhoria significativa nos comprimentos das *tours*, o que era esperado pois, além da densidade ser pequena, poucas eram as cidades muito próximas umas das outras.

As tabelas a seguir, nos mostram os resultados dos testes com o Filtro, realizados com o mesmo conjunto de parâmetros usados para o Elástico, porém avaliados para valores de ω iguais a 0.8, 0.6, 0.4 e 0.2. O parâmetro ω foi introduzido no capítulo III e faz parte da equação que determina o raio da c'rculo que define a região de atuação do algoritmo para cada ponto do anel.

(F) – 100 cidades – $\varepsilon = 0.05$					
α	β	ω	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.2	1.0	0.8	0.24	93	8.533650
0.2	2.0	0.8	0.24	90	8.478294
0.2	3.0	0.8	0.24	96	8.480454
0.4	1.0	0.8	0.24	90	8.896465
0.4	2.0	0.8	0.24	86	8.976612
0.4	3.0	0.8	0.24	93	8.524789
0.6	1.0	0.8	0.24	84	9.079422
0.6	2.0	0.8	0.24	83	9.072547
0.6	3.0	0.8	0.24	90	8.749887
0.8	1.0	0.8	0.24	82	9.895248
0.8	2.0	0.8	0.24	82	9.354962

(F) – 100 cidades – $\varepsilon = 0.05$					
α	β	ω	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.2	1.0	0.6	0.24	79	8.533589
0.2	2.0	0.6	0.24	78	8.505203
0.2	3.0	0.6	0.24	85	8.470696
0.4	1.0	0.6	0.24	79	8.868616
0.4	2.0	0.6	0.24	77	8.914053
0.4	3.0	0.6	0.24	79	8.714546
0.6	1.0	0.6	0.24	74	9.147588
0.6	2.0	0.6	0.24	76	9.306929
0.6	3.0	0.6	0.24	83	8.713329
0.8	1.0	0.6	0.24	75	9.725945
0.8	2.0	0.6	0.24	75	9.462613

(F) – 100 cidades – $\varepsilon = 0.05$					
α	β	ω	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.2	1.0	0.4	0.24	74	8.480421
0.2	2.0	0.4	0.24	84	8.68552
0.2	3.0	0.4	0.24	93	8.612960
0.4	1.0	0.4	0.24	70	8.824617
0.4	2.0	0.4	0.24	70	8.805823
0.4	3.0	0.4	0.24	86	9.652685
0.6	1.0	0.4	0.24	79	9.198747

(F) – 100 cidades – $\varepsilon = 0.05$					
α	β	ω	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.2	1.0	0.2	0.24	71	8.732889
0.2	2.0	0.2	0.24	73	8.776586
0.2	3.0	0.2	0.24	87	8.756747
0.4	1.0	0.2	0.24	78	8.934305
0.4	2.0	0.2	0.24	73	8.588531

Observamos que a razão entre o tempo de execução do Algoritmo Elástico e do Filtro aumenta de acordo com o decréscimo do valor de ω, ao contrário do que acontece com o comprimento da *tour*, não sendo, no entanto, muito maior do que aquela gerada pelo Elástico. Para densidades maiores de cidades, foi possível gerar com o Filtro, comprimentos de *tour* até menores do que os gerados pelo Elástico. Isso se justifica pela hipótese de elaboração do Filtro, que se aproxima mais do algoritmo Elástico para densidades maiores de cidades.

Na busca de valores menores para a *tour* gerada pelo Filtro para 100 cidades, foram feitas experiências, mudando valores de alguns dos parâmetros mantidos fixos em todos os testes, como por exemplo, a taxa de redução e o número de sub-iterações de K.

No teste A, aumentou-se o número de sub-iterações para cada valor de K para 3, conseguindo-se uma *tour* de menor valor, porém num tempo de processamento maior, uma vez que mais iterações foram realizadas.

$(F) - A - \varepsilon = 0.05$					
α	β	ω	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.15	3.5	0.8	0.24	180	8.273021

No teste B aumentamos o número de sub-iterações, a cada valor de K, para 5. A *tour* foi reduzida mais ainda, mas o tempo de processamento praticamente dobrou.

$(F) - B - \varepsilon = 0.05$					
α	β	ω	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.15	3.5	0.8	0.24	339	8.190655

Por último, além de ter aumentado o número de sub-iterações para 5, diminuiu-se a taxa de redução de K para 3%. O tempo de processamento aumentou assustadoramente porém o comprimento da *tour* gerada foi bastante próximo daquele conseguido pelo algoritmo LK.

$(F) - C - \varepsilon = 0.05$					
α	β	ω	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.15	3.5	0.8	0.24	559	8.162786

Por fim, para efeito de comparação, realizou-se também, uma série de testes com o algoritmo LK. Esse algoritmo se mostrou bastante eficiente para 100 cidades, pois comprimentos de *tours* menores que o Elástico e o Filtro foram conseguidos, num tempo de processamento igual ou melhor que o Elástico.

(LK) - 100 cidades	
Tempo de execução(seg)	*Percurso*
42	8.012726
116	8.068161
111	8.164400
132	7.976528
68	7.948021
88	8.085966
175	8.456726

Nos primeiros testes realizados com esse algoritmo, todos os níveis de Backtracking foram executados, gerando tempos de processamento muito grandes. Constatou-se entretanto, que a eliminação de alguns desses níveis durante a execução do algoritmo não acarretava numa perda no custo da *tour* gerada. Essa eliminação foi então adotada no sentido de se obter tempos de execução mais rápidos.

A seguir serão mostradas as tabelas dos testes realizados com o Algoritmo Elástico e com o Filtro, para 200 cidades.

(AE) - 200 cidades - $\varepsilon = 0.05$				
α	β	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.2	1.0	0.24	534	12.156892
0.2	2.0	0.24	524	12.185989
0.2	3.0	0.24	527	12.082848
0.2	4.0	0.24	561	11.913215
0.4	1.0	0.24	487	12.196545
0.4	2.0	0.24	505	12.195210
0.4	3.0	0.24	487	12.391413
0.6	1.0	0.24	499	12.603031
0.6	2.0	0.24	475	12.350251
0.6	3.0	0.24	523	12.577269
0.8	1.0	0.24	490	12.739984
0.8	2.0	0.24	474	12.565989
0.8	3.0	0.24	522	12.290537

(F) – 200 cidades – $\varepsilon = 0.05$					
α	β	ω	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.2	1.0	0.8	0.24	373	12.268305
0.2	2.0	0.8	0.24	378	12.134330
0.2	3.0	0.8	0.24	375	12.102608
0.4	1.0	0.8	0.24	357	12.261155
0.4	2.0	0.8	0.24	357	12.232881
0.6	1.0	0.8	0.24	361	12.305346
0.6	2.0	0.8	0.24	317	13.774194
0.8	1.0	0.8	0.24	303	15.473497

(F) – 200 cidades – $\varepsilon = 0.05$					
α	β	ω	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.2	1.0	0.6	0.24	367	12.434623
0.2	2.0	0.6	0.24	346	12.142917
0.2	3.0	0.6	0.24	346	12.220894
0.4	1.0	0.6	0.24	317	12.356052
0.4	2.0	0.6	0.24	308	12.751890
0.6	1.0	0.6	0.24	312	12.510341
0.6	2.0	0.6	0.24	296	12.388120
0.6	3.0	0.6	0.24	329	12.327866
0.8	1.0	0.6	0.24	360	12.981072
0.8	2.0	0.6	0.24	382	12.442481
0.8	3.0	0.6	0.24	340	13.683556

(F) – 200 cidades – $\varepsilon = 0.05$					
α	β	ω	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.2	1.0	0.4	0.24	355	12.034768
0.2	2.0	0.4	0.24	358	12.143627

(F) – 200 cidades – $\varepsilon = 0.05$					
α	β	ω	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.2	1.0	0.2	0.24	326	12.264388
0.2	2.0	0.2	0.24	286	12.592870

Com o conjunto de parâmetros com o qual obteve-se o menor comprimento de *tour*, realizou-se, tanto para o AE quanto para o Filtro, um teste com $\varepsilon = 0.02$. A diferença entre os comprimentos de *tour* foi um pouco maior, o que era de se esperar pois a densidade de cidades aumentou.

200 cidades						
	ε	α	β	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
AE	0.02	0.2	4.0	0.24	859	11.72
	0.05	0.2	4.0	0.24	561	11.91
F	0.02	0.2	1.0	0.24	389	12.02
	0.05	0.2	1.0	0.24	355	12.03

Observou-se nos testes realizados, que a propriedade de diminuição do comprimento da *tour* de acordo com o aumento de β, para um valor de α constante, muitas vezes não foi satisfeito. Este fato pode ser justificado, em casos de leves diferenças, pelo critério de avaliação da *tour* gerada pelo algoritmo, discutido em V.3.

Os testes realizados com o algoritmo LK para 200 cidades foram

(LK) – 200 cidades	
Tempo de execução(seg)	*Percurso*
894	11.228683
637	11.257902
570	11.123185
754	11.219497
599	11.194624
413	11.234623
913	11.195530
894	11.074368
929	11.196472

No caso de 300 cidades , o Algoritmo Elástico se comportou como para os outros conjuntos de cidades. No entanto, para $\alpha = 0.2$ e $\beta = 4.0$, ao invés de uma *tour* menor, obteve-se uma maior. Isso se deve ao fato de se ter estipulado um número inicial de pontos da circunferência bem maior do que nos outros casos, tornando a distância entre eles mínima. Como o valor de β é muito grande, os pontos se deslocam em direção oposta a seus vizinhos, 'àumentando" o comprimento do anel.

(AE) - 300 cidades - $\varepsilon = 0.05$				
α	β	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.2	1.0	0.24	1645	14.822573
0.2	2.0	0.24	1184	14.734464
0.2	3.0	0.24	1292	14.800883
0.2	4.0	0.24	1346	15.028765
0.4	1.0	0.24	1617	17.211405
0.4	2.0	0.24	1134	14.913396
0.4	3.0	0.24	1177	14.922175
0.6	1.0	0.24	1163	15.452832
0.6	2.0	0.24	1149	15.327007
0.6	3.0	0.24	1272	15.236254
0.8	1.0	0.24	1228	15.724594

Dentre os testes realizados com o Filtro, a menor *tour* obtida foi para $\omega = 0.8$.

(F) - 300 cidades - $\varepsilon = 0.05$					
α	β	ω	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.2	1.0	0.8	0.24	820	14.898540
0.2	2.0	0.8	0.24	874	14.903065
0.2	3.0	0.8	0.24	898	14.548990
0.4	1.0	0.8	0.24	845	14.831373
0.4	2.0	0.8	0.24	856	14.646933
0.4	3.0	0.8	0.24	741	14.851957
0.6	1.0	0.8	0.24	818	15.425999
0.6	2.0	0.8	0.24	798	15.587884
0.8	1.0	0.8	0.24	719	16.956596

(F) - 300 cidades - $\varepsilon = 0.05$					
α	β	ω	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.2	1.0	0.6	0.24	605	14.893279
0.2	2.0	0.6	0.24	629	14.698704
0.2	3.0	0.6	0.24	670	15.203262
0.4	1.0	0.6	0.24	673	14.942806
0.4	2.0	0.6	0.24	605	15.383842
0.4	3.0	0.6	0.24	643	15.422233
0.6	1.0	0.6	0.24	656	15.032210
0.6	2.0	0.6	0.24	665	15.186347
0.8	1.0	0.6	0.24	644	15.736422
0.8	2.0	0.6	0.24	636	17.590281

(F) - 300 cidades - $\varepsilon = 0.05$					
α	β	ω	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.2	1.0	0.4	0.24	621	16.681480

(F) - 300 cidades - $\varepsilon = 0.05$					
α	β	ω	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.1	1.0	0.2	0.24	709	15.501309

Para $\omega = 0.2$, não ocorreu a convergência do algoritmo para nenhum dos parâmetros avaliados com os outros valores de ω, convergindo apenas para $\alpha = 0.1$ e $\beta = 1.0$. A justificativa de uma *tour* melhor neste caso, porém num tempo pior, se deve ao valor de α. Como α é pequeno, a força de β se acentua, tornando mais lenta, a execução do algoritmo, e melhor contribuindo para a minimização da *tour*. Além disso, o número inicial de pontos na circunferência foi também bem maior (208), exigindo mais cálculos nas primeiras iterações.

Podemos comparar, a seguir, os testes realizados para $\varepsilon = 0.02$. As *tours* obtidas foram de menor comprimento.

300 cidades						
	ε	α	β	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
AE	0.02	0.2	2.0	0.24	1352	14.470851
	0.05	0.2	2.0	0.24	1184	14.7344
F	0.02	0.2	3.0	0.24	920	14.350313
	0.05	0.2	3.0	0.24	898	14.548990

Os testes com o LK para 300 cidades foram

(LK) - 300 cidades	
Tempo de execução(seg)	*Percurso*
1412	13.813244
1809	13.475449
1430	13.624891
2178	13.507176
1471	13.346654
1187	13.552829
2556	13.459849
946	13.753555

O Algoritmo Elástico testado para 400 cidades gerou os seguintes resultados:

(AE) - 400 cidades - $\varepsilon = 0.05$				
α	β	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.2	1.0	0.24	2409	17.297983
0.2	2.0	0.24	2285	17.119900
0.2	3.0	0.24	2381	16.937376
0.2	4.0	0.24	2294	17.233791
0.4	1.0	0.24	2254	17.442051
0.4	2.0	0.24	2129	17.291521
0.4	3.0	0.24	2259	16.862270
0.6	1.0	0.24	2133	20.056059
0.6	2.0	0.24	2284	20.916960
0.6	3.0	0.24	2195	20.723671
0.8	1.0	0.24	2165	18.321589
0.8	2.0	0.24	2027	18.866848

No Filtro, basicamente todas as propriedades citadas anteriormente foram satisfeitas. Assim como para 300 cidades, a diminuição do valor de ω causou a diminuição de casos que convergiram. Note que a diferença entre a *tour* gerada pelo Algoritmo Elástico e o Filtro vem diminuindo com o aumento do número de cidades, obtendo-se *tours* melhores em tempos inferiores.

(F) - 400 cidades - $\varepsilon = 0.05$					
α	β	ω	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.2	1.0	0.8	0.24	1928	17.121510
0.2	2.0	0.8	0.24	1660	17.053814
0.2	3.0	0.8	0.24	1738	16.879519
0.2	4.0	0.8	0.24	2010	16.967617
0.4	1.0	0.8	0.24	1576	17.480095
0.4	2.0	0.8	0.24	1551	16.900141
0.4	3.0	0.8	0.24	1473	17.078512
0.6	1.0	0.8	0.24	1465	21.054258
0.6	2.0	0.8	0.24	1669	17.384533
0.8	1.0	0.8	0.24	1678	18.352219

(F) - 400 cidades - $\varepsilon = 0.05$					
α	β	ω	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.2	1.0	0.6	0.24	1248	17.688753
0.2	2.0	0.6	0.24	1351	17.189056
0.2	3.0	0.6	0.24	1381	17.094851
0.4	1.0	0.6	0.24	1320	17.148020
0.4	2.0	0.6	0.24	1373	16.919930
0.4	3.0	0.6	0.24	1251	17.216640

(F) – 400 cidades – $\varepsilon = 0.05$					
α	β	ω	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.2	1.0	0.4	0.24	1288	17.517828

(F) – 400 cidades – $\varepsilon = 0.05$					
α	β	ω	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.1	2.0	0.2	0.24	1436	17.951866

Os casos testados para um valor de ε menor (0.02), nos deu, como esperado, comprimentos menores de *tour*.

400 cidades						
	ε	α	β	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
AE	0.02	0.4	3.0	0.24	2747	16.650024
	0.05	0.4	3.0	0.24	2259	16.86227
F	0.02	0.2	3.0	0.24	1769	16.8
	0.05	0.2	3.0	0.24	1738	16.8795

Os testes realizados para o algoritmo LK foram

(LK) – 400 cidades	
Tempo de execução(seg)	*Percurso*
2309	15.390247
3903	15.456793
2318	15.461168
4092	17.276907
5235	15.508545
3110	15.935468
2466	15.584679
3002	15.851535
1966	15.534723
2988	15.349201

É válido ressaltar que todos os testes com o algoritmo LK foram realizados a partir de uma *tour* gerada por uma outra heurística, a heurística do vizinho mais próximo. A *tour* obtida é usada como *tour* inicial. Com esse conjunto de cidades fizemos uma experiência aplicando o algoritmo LK numa *tour* gerada aleatoriamente, sem nenhuma melhoria. O tempo de processamento aumentou bastante, como mostra a tabela abaixo.

(LK) - 400 cidades	
Tempo de execução(seg)	*Percurso*
8405	15.420321

O comportamento do Algoritmo Elástico e do Filtro, para 500 cidades foi semelhante aos demais.

(AE) - 500 cidades - $\varepsilon = 0.05$				
α	β	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.2	1.0	0.24	3624	18.680317
0.2	2.0	0.24	3197	18.561111
0.2	3.0	0.24	3769	18.547966
0.4	1.0	0.24	3553	18.795948
0.4	2.0	0.24	3403	19.144619
0.4	3.0	0.24	2817	18.996958
0.6	1.0	0.24	3114	20.321362
0.6	2.0	0.24	2928	19.524467
0.6	3.0	0.24	2824	21.977888
0.8	1.0	0.24	2912	19.524467
0.8	2.0	0.24	3670	20.098566
0.8	3.0	0.24	2938	20.110176

(F) - 500 cidades - $\varepsilon = 0.05$					
α	β	ω	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.2	1.0	0.8	0.24	2217	18.707186
0.2	2.0	0.8	0.24	2930	18.887478
0.2	3.0	0.8	0.24	2639	19.131365
0.4	1.0	0.8	0.24	2107	19.158161
0.4	2.0	0.8	0.24	2014	19.368954
0.4	3.0	0.8	0.24	2248	19.153448
0.6	1.0	0.8	0.24	2456	20.840771
0.6	2.0	0.8	0.24	2203	19.216772
0.8	1.0	0.8	0.24	1967	19.739744

(F) - 500 cidades - $\varepsilon = 0.05$					
α	β	ω	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.2	1.0	0.6	0.24	2019	18.571651
0.2	2.0	0.6	0.24	2393	18.636856
0.4	1.0	0.6	0.24	2078	19.947794
0.4	2.0	0.6	0.24	1792	19.433918
0.8	1.0	0.6	0.24	1765	21.148518

(F) - 500 cidades - $\varepsilon = 0.05$					
α	β	ω	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.1	1.0	0.4	0.24	2349	19.424685

(F) - 500 cidades - $\varepsilon = 0.05$					
α	β	ω	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.08	1.5	0.2	0.24	2297	19.91

Em vista da dificuldade de se encontrar um conjunto de parâmetros para $\omega = 0.4$ que resultasse na convergência do algoritmo, optou-se por realizar um teste com uma taxa de redução de K igual a 2%. Obtivemos um valor para a *tour* gerada bem melhor do que aquele realizado, para esse valor de ω, com a taxa de redução igual a 5%. Nota-se que a *tour* gerada foi melhor até do que aquelas geradas pelo Algoritmo Elástico, porém em um tempo pior, uma vez que os testes realizados com o Elástico foram com uma taxa de redução de K (TRK), de 5%.

(F) - 500 cidades - $\varepsilon = 0.05$ - *(TRK = 2%)*					
α	β	ω	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.07	1.0	0.4	0.24	7080	18.303869

O teste para um valor menor para a vizinhança de cada cidade ($\varepsilon = 0.015$) resultou em comprimentos de *tour* razoavelmente menores do que aqueles gerados para $\varepsilon = 0.05$.

500 cidades						
	ε	α	β	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
AE	0.015	0.2	3.0	0.24	5619	18.162119
	0.05	0.2	3.0	0.24	3769	18.547966
F	0.015	0.2	1.0	0.24	2745	18.397887
	0.05	0.2	1.0	0.24	2019	18.571651

Os testes realizados com o algoritmo LK foram

(LK) - 500 cidades	
Tempo de execução(seg)	*Percurso*
5684	17.469341
9561	17.639362
8051	17.005066
7367	17.291508
5872	17.258121
5460	17.674465
4130	17.378239
4244	17.172838
5339	17.546305
5218	17.325338

Em todos os casos testados até agora, o número de pontos iniciais distribuídos pela circunferência foi de $\frac{N}{4}$ ou $\frac{N}{2}$, sendo N ó número de cidades distribuídas pelo quadrado. Já para os conjuntos de 750 e 1000 cidades, estabeleceu-se apenas o valor de $\frac{N}{2}$ como número inicial de pontos da circunferência.

É interessante relembrar neste momento que, associado à evolução do Algoritmo Elástico, existe uma função de Energia, limitada inferiormente, com a propriedade de decrescer a cada deslocamento de um ponto do anel até atingir um mínimo.

Segundo Durbin [8], quando $K \rightarrow 0$, todos os mínimos de energia correspondem a *tours* válidas para o problema do Caixeiro Viajante.

Simmen [26] contradisse esse argumento quando provou a existência de configurações geradas pelo algoritmo, porém não válidas como *tours*, quando $K \rightarrow 0$. Ele provou que configurações onde se encontra algum ponto do anel situado exatamente entre duas cidades, correspondem a situações de equilíbrio (ou mínimos de energia). Ocorre o fato de ambas as cidades exercerem forças opostas, porém de mesma intensidade fazendo com que o ponto não saia do lugar.

Se a vizinhança de cada cidade não for o suficientemente grande para "capturar" esse ponto, o algoritmo nunca convergirá. Foi o que aconteceu em alguns dos testes realizados quando o valor de ε era menor que 0.015.

Os testes com 750 cidades para o Algoritmo Elástico e o Filtro serão mostrados a seguir. Como a densidade de cidades é bem maior, o valor de ε considerado para a convergência do algoritmo foi de 0.03.

(AE) - 750 cidades - $\varepsilon = 0.03$				
α	β	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.2	3.0	0.24	9638	23.508518
0.3	3.0	0.24	9136	23.508518
0.4	3.0	0.24	7817	23.274508
0.6	2.0	0.24	7661	23.830988
0.8	2.0	0.24	8801	24.5028

Os testes com o Filtro foram

(F) - 750 cidades - $\varepsilon = 0.03$					
α	β	ω	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.2	3.0	0.8	0.24	5114	24.460466
0.3	3.0	0.8	0.24	6279	23.588985
0.4	3.0	0.8	0.24	5021	24.679857
0.6	3.0	0.8	0.24	6228	23.994911
0.8	1.0	0.8	0.24	6507	25.052166

(F) - 750 cidades - $\varepsilon = 0.03$					
α	β	ω	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.2	3.0	0.6	0.24	6002	24.019114
0.3	3.0	0.6	0.24	5247	23.430515
0.4	3.0	0.6	0.24	4441	26.187963
0.6	3.0	0.6	0.24	4952	24.913708
0.8	1.0	0.6	0.24	5483	25.235521

Para os valores de ω iguais a 0.4 e 0.2, não se encontrou um conjunto de parâmetros que resultasse na convergência do algoritmo. Pode-se observar, ao longo da descrição dos testes, que quanto maior for a densidade de cidades, menor será o número de casos que convergem para valores pequenos de ω.

Isso se deve ao fato de que, quando a densidade de cidades é muito grande, a contribuição para o cálculo de influências, mesmo quando ω é pequeno, é bem grande também, já que existem muitas cidades próximas do ponto. Como no caso do Filtro, a influência de cada cidade não é normalizada por todos os pontos (apenas por aqueles que são influenciados por ela), então a parcela de influência torna-se ainda maior. Para que os pontos não tenham um deslocamento muito grande, é preciso que se estipule para α um valor bem pequeno.

Acontece que, à medida que $K \rightarrow 0$, o número de quadrados em volta do ponto diminui (ver capítulo III). Daí, a influência em torno do ponto torna-se menor e consequentemente o primeiro termo da equação de deslocamento do ponto fica muito pequeno, em um momento em que as distâncias dos pontos às cidades não são suficientemente pequenas para que se possa considerar a convergência do algoritmo. Sendo assim, aqui também seria interessante definir α como sendo inversamente proporcional a K, sem comprometer, no entanto, a definição do algoritmo. Uma outra alternativa seria estabelecer, nesses casos, testes com a taxa de redução de K igual a 2%. Desta maneira, mesmo estipulando um valor muito pequeno para

α, a convergência do algoritmo torna-se possível, uma vez que são realizadas mais iterações com valores grandes de K, permitindo um deslocamento maior dos pontos em direção das cidades e conseqüentemente a convergência do algoritmo. A única desvantagem é que o tempo de processamento nesses casos, aumenta muito, tanto para o Elástico quanto para o Filtro, como mostram os resultados abaixo.

(AE) - 750 cidades - (TRK: 2%) $-\ \varepsilon = 0.03$				
α	β	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.06	1.0	0.24	17272	23.680923
0.2	3.0	0.24	15747	22.937342
0.3	3.0	0.24	18948	23.187443
0.3	3.5	0.24	19330	22.998579
0.6	2.0	0.24	18350	24.518309

(F) - 750 cidades - (TRK: 2%) $-\ \varepsilon = 0.03$					
α	β	ω	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.2	3.0	0.8	0.24	13789	23.013813
0.3	3.0	0.8	0.24	13263	22.802124
0.3	3.5	0.8	0.24	15684	23.271742

(F) - 750 cidades - (TRK: 2%) $-\ \varepsilon = 0.03$					
α	β	ω	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.2	3.0	0.6	0.24	12790	22.881351
0.3	3.0	0.6	0.24	12903	22.769537
0.3	3.5	0.6	0.24	13274	23.098648

(F) - 750 cidades - (TRK: 2%) $-\ \varepsilon = 0.03$					
α	β	ω	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.06	1.0	0.4	0.24	15931	24.108826
0.06	2.0	0.4	0.24	14507	23.563063

(F) - 750 cidades - (TRK: 2%) $-\ \varepsilon = 0.03$					
α	β	ω	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.06	1.0	0.2	0.24	15573	23.353386

Com o objetivo de encontrar uma *tour* de menor comprimento, diminuiu-se o valor de ε para 0.01 e a taxa de redução de K para 2% para o conjunto de parâmetros que originou o menor comprimento de *tour* quando o decréscimo de K era de 5%. O ganho obtido em relação ao comprimento da *tour* foi bastante razoável.

(F) - 750 cidades - (TRK: 2%) $-\ \varepsilon = 0.01$					
α	β	ω	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.3	3.0	0.6	0.24	18767	22.456593

Outra alteração realizada foi quanto ao tamanho do lado do quadrado onde estão situadas as cidades. Mesmo para valores de ω pequenos, que determinam áreas pequenas de atuação do algoritmo para cada ponto do anel, o número de cidades que influenciam o deslocamento do ponto pode ser grande, causando deslocamentos bruscos. Aumentando-se o lado do quadrado, as cidades ficam mais espalhadas e consequentemente a densidade diminui. Na tentativa de encontrar uma *tour* melhor fizemos testes com o lado do quadrado igual a 2 para os seguintes conjuntos de parâmetros

(AE) - 750 cidades - (TRK: 5%) $-\ \varepsilon = 0.03$				
α	β	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.2	3.0	0.24	8219	46.23365

para o algoritmo Elástico e

(F) - 750 cidades - (TRK: 5%) $-\ \varepsilon = 0.03$					
α	β	ω	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.2	3.0	0.8	0.24	5938	46.350834

(F) - 750 cidades - (TRK: 2%) $-\ \varepsilon = 0.01$					
α	β	ω	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.3	3.0	0.6	0.24	17304	45.273937

para o Filtro.

Dividindo os comprimentos das *tours* obtidas pelo tamanho do lado do quadrado, obtemos 23.116825, 23.175417 e 22.636968, respectivamente. Foram medidas muito boas, obtidas em menor tempo.

Por fim, exibiremos os resultados obtidos pela aplicação do algoritmo LK neste conjunto de cidades.

(LK) – 750 cidades	
Tempo de execução(seg)	*Percurso*
18349	21.308992
20374	21.131855
29307	21.182213
9037	21.124472
14209	21.303377
14567	21.467428
18781	21.005896
11668	21.107832
20659	21.305077
21577	21.380779
16270	21.097866
17465	21.383226
15189	21.186815

O último conjunto de cidades submetido a testes de parâmetros, foi o de 1000 cidades.

Considerou-se nestes testes, a convergência do algoritmo quando existisse pelo menos um ponto do anel numa vizinhança de cada cidade, de raio 0.02.

A tabela abaixo, mostra os resultados obtidos para o Algoritmo Elástico, considerando-se a taxa de redução de K de 5%.

(AE) – 1000 cidades – $\varepsilon = 0.02$				
α	β	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.2	3.0	0.24	17144	26.652563
0.2	3.5	0.24	15727	29.3124
0.4	2.0	0.24	16520	26.630796
0.6	2.0	0.24	15656	26.630796
0.8	2.0	0.24	15727	27.733170

O teste com $\alpha = 0.2$ e $\beta = 3.5$ gerou uma *tour* de custo alto. Isso se justifica pelo número inicial de pontos estabelecido neste caso — 600.

Os resultados obtidos para o Filtro foram

(F) - 1000 cidades - $\varepsilon = 0.02$					
α	β	ω	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.15	3.0	0.8	0.24	12231	26.801439
0.2	3.0	0.8	0.24	11865	26.611864
0.2	4.0	0.8	0.24	11623	26.923534
0.4	2.0	0.8	0.24	10740	26.676498

(F) - 1000 cidades - $\varepsilon = 0.02$					
α	β	ω	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.2	2.0	0.6	0.24	11029	26.846542
0.2	3.0	0.6	0.24	11545	26.141281
0.4	3.0	0.6	0.24	10835	26.319637

Como para o conjunto de 750 cidades, nenhum caso convergiu para os valores de ω iguais a 0.4 e 0.2. Pelo mesmo motivo, foi realizada uma série de testes, tanto para o Elástico quanto para o Filtro, com a taxa de redução de K igual a 2%.

(AE) - 1000 cidades - (TRK:2%) - $\varepsilon = 0.02$				
α	β	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.2	3.0	0.24	24512	25.738924
0.3	3.0	0.24	20166	26.253941
0.6	2.0	0.24	19656	29.083393

Com o intuito de comparação com os casos testados para os valores de ω iguais a 0.4 e 0.2, realizou-se um teste com o Elástico, para os parâmetros $\alpha = 0.06$ e $\beta = 1.0$, que no entanto, não convergiu.

A melhor *tour* obtida pelo Algoritmo Elástico foi para $\alpha = 0.2$ e $\beta = 3.0$. Reduziu-se então, para esse mesmo conjunto de parâmetros, o valor de ε para 0.01, obtendo-se um resultado bastante razoável.

(AE) - 1000 cidades - (TRK = 2%) - $\varepsilon = 0.01$				
α	β	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.2	3.0	0.24	47752	25.524446

Os resultados dos testes realizados com o Filtro foram

(F) - 1000 cidades - (TRK:2%) - $\varepsilon = 0.02$					
α	β	ω	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.06	1.0	0.8	0.24	26315	27.079483
0.2	3.0	0.8	0.24	20862	26.611864
0.3	3.0	0.8	0.24	16011	26.175003
0.4	2.0	0.8	0.24	17029	26.972145
0.6	2.0	0.8	0.24	13087	27.900216

(F) - 1000 cidades - (TRK:2%) - $\varepsilon = 0.02$					
α	β	ω	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.06	1.0	0.6	0.24	23820	26.725420
0.3	3.0	0.6	0.24	25211	26.063679
0.2	3.0	0.6	0.24	16105	25.894808
0.6	2.0	0.6	0.24	12582	28.369907

(F) - 1000 cidades - (TRK:2%) - $\varepsilon = 0.02$					
α	β	ω	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.06	1.0	0.4	0.24	15937	26.817377

(F) - 1000 cidades - (TRK:2%) - $\varepsilon = 0.02$					
α	β	ω	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.05	1.0	0.2	0.24	14331	26.706167
0.06	1.0	0.2	0.24	18805	26.653877

Com o objetivo de se encontrar uma *tour* de melhor custo, modificou-se alguns parâmetros, mantidos fixos em todos os testes. Na primeira tabela a seguir, aumentou-se o número de sub-iterações para cada valor de K para 4 e sua taxa de redução foi de 1%, enquanto que na segunda, manteve-se a taxa em 2%, aumentando-se apenas o número de sub-iterações para 4.

(F) - 1000 cidades - Taxa de redução de K:1% - $\epsilon = 0.01$					
α	β	ω	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.2	3.0	0.8	0.24	66692	25.751961

(F) - 1000 cidades - Taxa de redução de K:2% - $\epsilon = 0.01$					
α	β	ω	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.2	3.0	0.8	0.24	64264	25.849808

A *tour* gerada não foi boa o suficiente para justificar um tempo de processamento tão grande. Algumas mudanças de parâmetros que não comprometem tanto o tempo de processamento, produzem até comprimentos de *tour* melhores, como mostra a tabela abaixo. Escolheu-se neste exemplo, o conjunto de parâmetros que gerou a melhor *tour*, dentre todos os testes realizados com o Filtro, agora para

$\varepsilon = 0.01$.

(F) - 1000 cidades - Taxa de redução de K:2% - ϵ = 0.01					
α	β	ω	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.2	3.0	0.6	0.24	32078	25.521399

Foram também aqui realizadas, experiências mudando-se o tamanho do lado do quadrado onde estão situadas as cidades. Aumentou-se o lado para 3, já que a densidade de cidades é maior. Os resultados obtidos foram bastante razoáveis, se considerarmos principalmente os tempos de execução.

(F) - 1000 cidades - Taxa de redução de K: 5% - ε = 0.01					
α	β	ω	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.2	3.0	0.8	0.24	22109	75.77708

(F) - 1000 cidades - Taxa de redução de K: 5% - ε = 0.03					
α	β	ω	$K_{inicial}$	*Tempo de execução(seg)*	*Percurso*
0.2	3.0	0.8	0.24	14160	77.025

Dividindo os valores das *tours* obtidas pelo tamanho do lado do quadrado obtemos respectivamente, 25.259016 e 25.675. A primeira *tour* foi a melhor obtida, dentre todos os testes realizados com esse conjunto de cidades.

Apresentaremos agora, os resultados obtidos pela aplicação do algoritmo LK.

(LK) - 1000 cidades	
Tempo de execução(seg)	*Percurso*
60843	24.240452
62773	24.271664
55012	24.148943
50043	23.945404
39319	24.539206
52855	24.212910
43109	24.25927
34452	24.465031
49762	24.02960

Uma vez que foram obtidos bons resultados com os testes realizados com a mudança do lado do quadrado, aplicou-se o algoritmo LK para o mesmo conjunto de cidades gerado para esse caso(conjunto este, modificado proporcionalmente ao tamanho do lado do quadrado), obtendo-se

(LK) - 1000 cidades	
Tempo de execução(seg)	*Percurso*
85696	71.497910

A alteração do lado do quadrado é apenas significativa para o Algoritmo Elástico e o Filtro, uma vez que estes são algoritmos essencialmente geométricos, ao contrário do LK.

V.5 Gráficos

Exibiremos a seguir, uma série de gráficos que representam os respectivos desempenhos dos três algoritmos implementados neste trabalho, observados ao longo de todos os testes realizados.

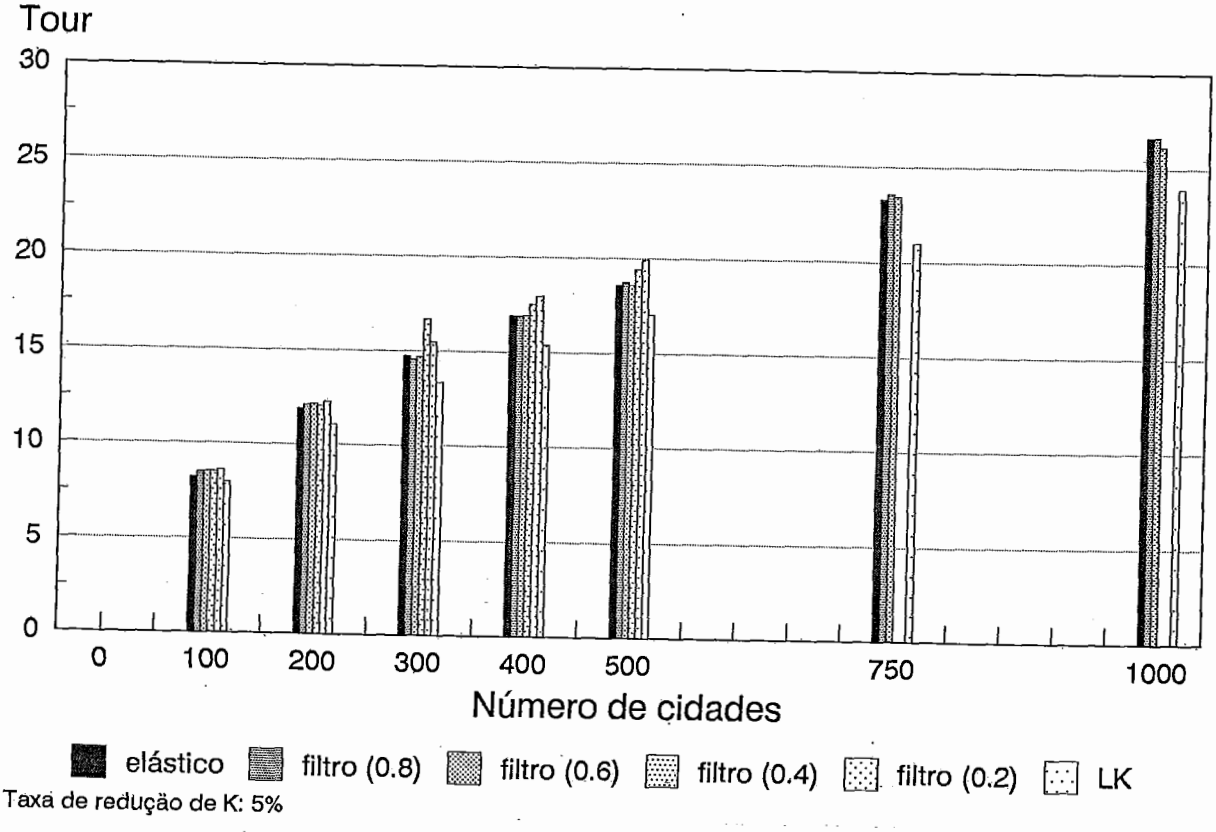


Figura V.5: Menores comprimentos de *tour* obtidos

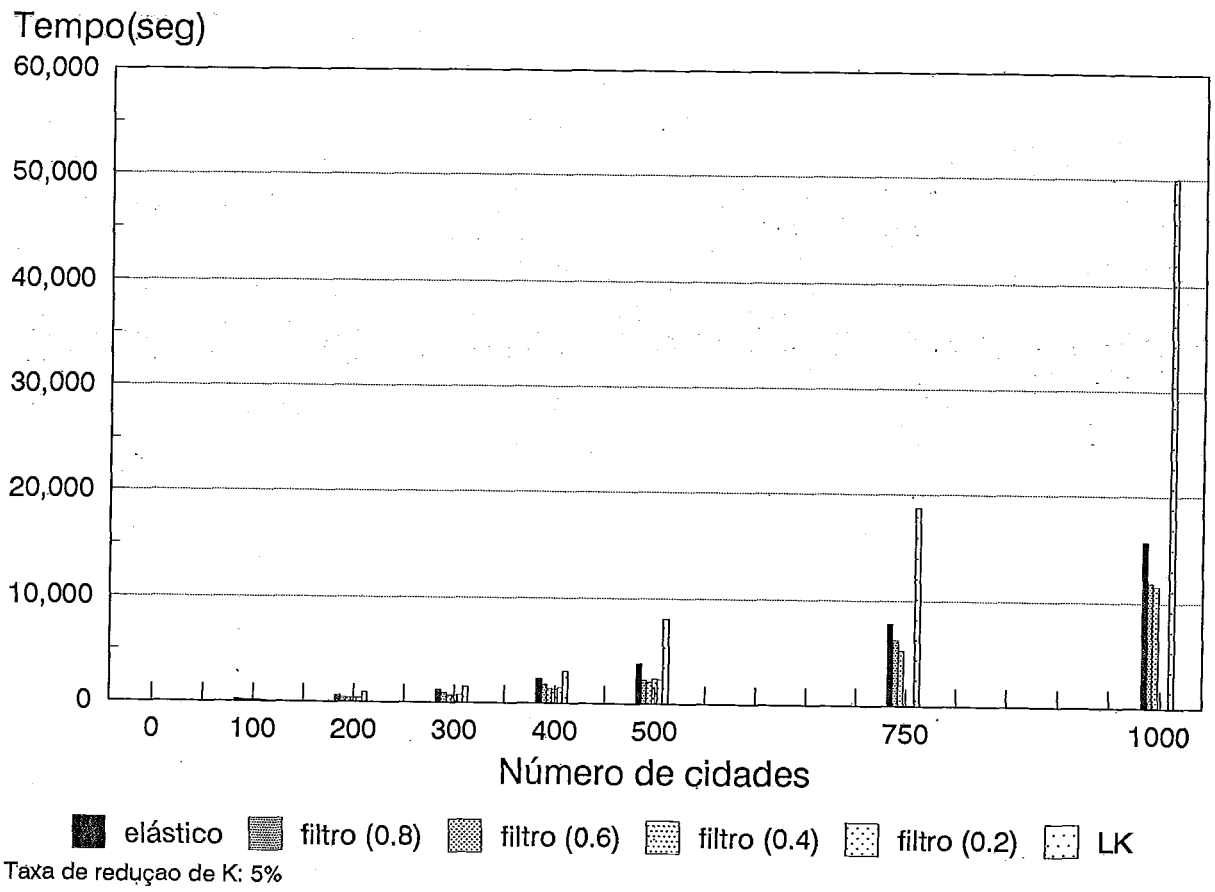


Figura V.6: Tempos de execução

A figura V.5 representa os menores comprimentos de *tour* encontrados. Observa-se que o desempenho do algoritmo LK foi bastante razoável. No entanto, o tempo de execução requerido pelo LK para alcançar tais resultados foi bem pior, principalmente para conjuntos maiores de cidades, como mostra a figura V.6.

O gráfico dos percursos médios foram (obtidos respectivamente, para cada algoritmo, a partir de todas as instâncias que convergiram para cada conjunto de cidades. Tal gráfico pode ser encontrado na figura V.7.

Observa-se que os comprimentos de *tour* obtidos pelo Filtro para 400 e 1000 cidades, foram até melhores do que aqueles obtidos pelo Elástico.

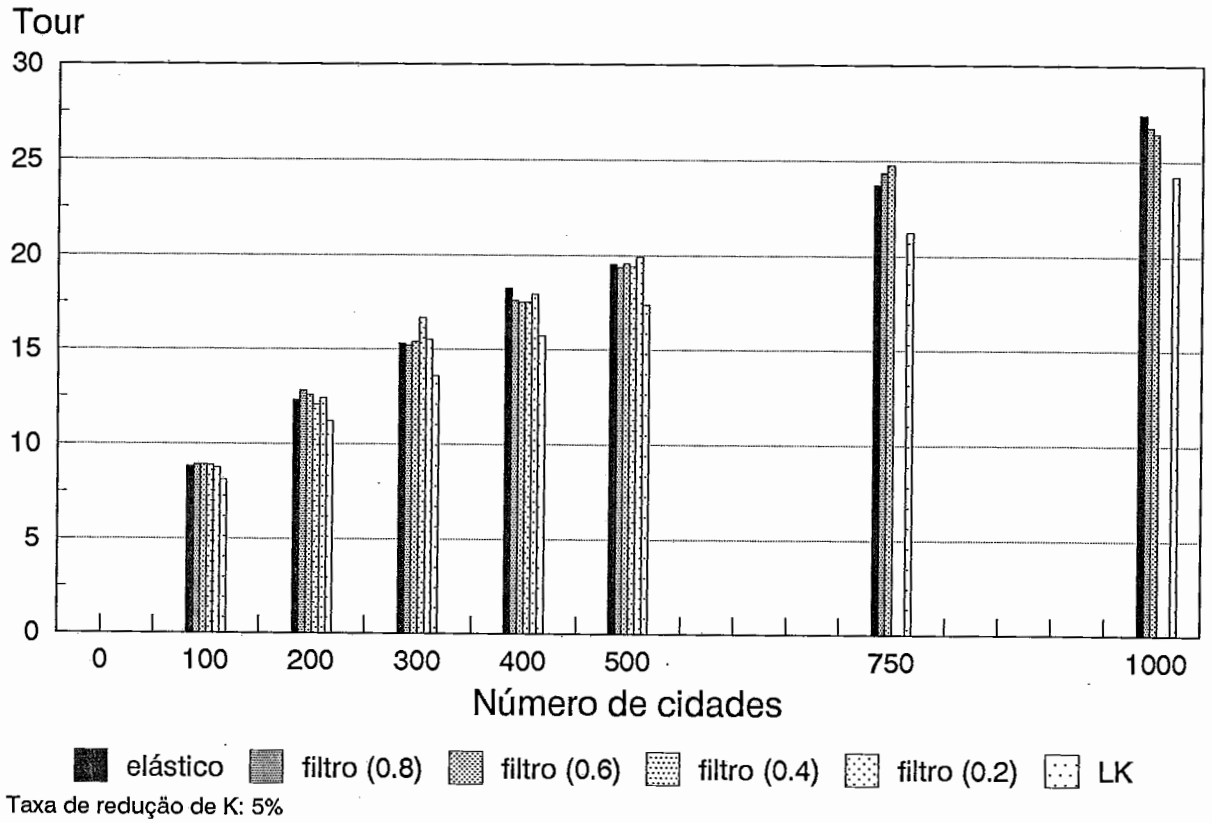


Figura V.7: Percursos médios

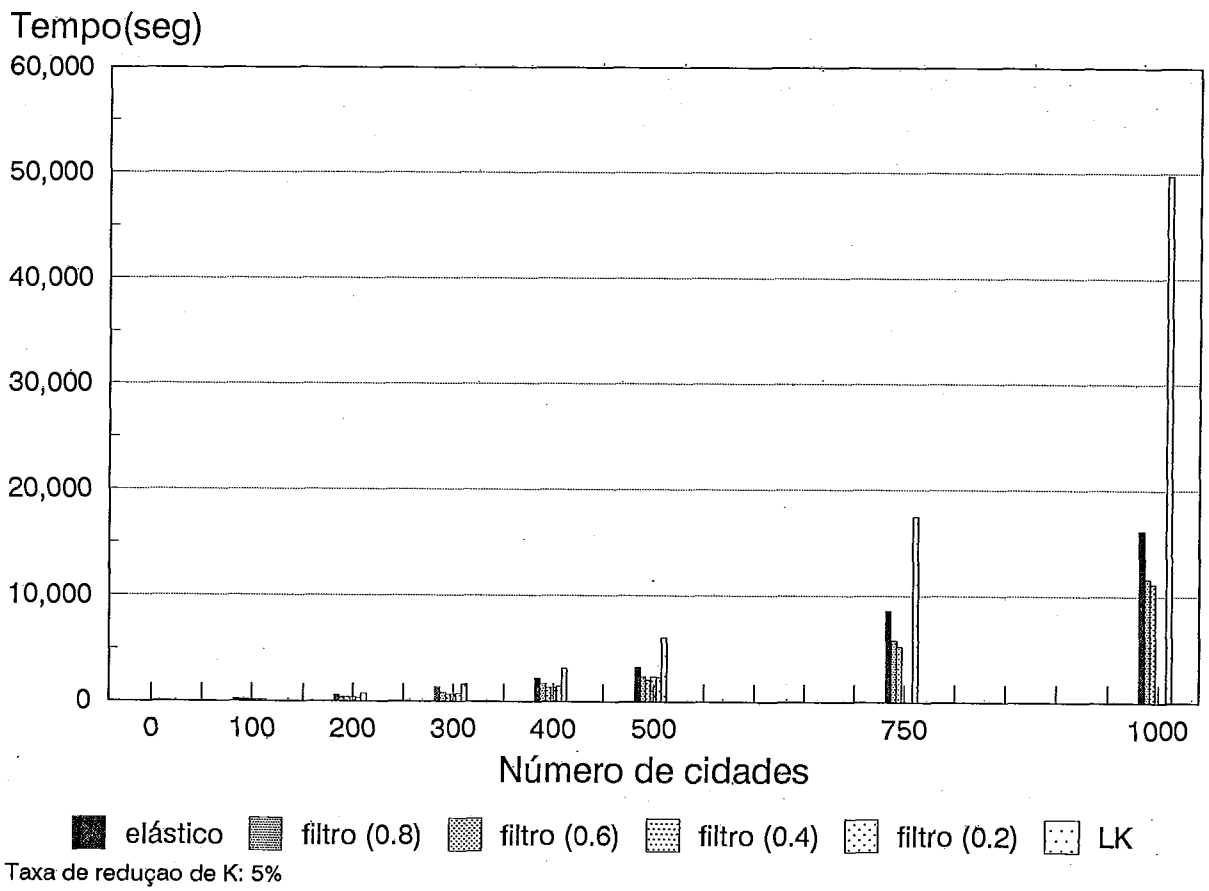


Figura V.8: Tempos médios

Como para os conjuntos de 750 e 1000 cidades nenhum teste realizado com o Filtro, para valores de ω iguais a 0.4 e 0.2, convergiu, não foram apresentadas nos gráficos acima, as medidas correspondentes a esses valores. Por esse motivo, nos gráficos exibidos a seguir, considerou-se para esses dois conjuntos de cidades, a taxa de redução de K de 2%.

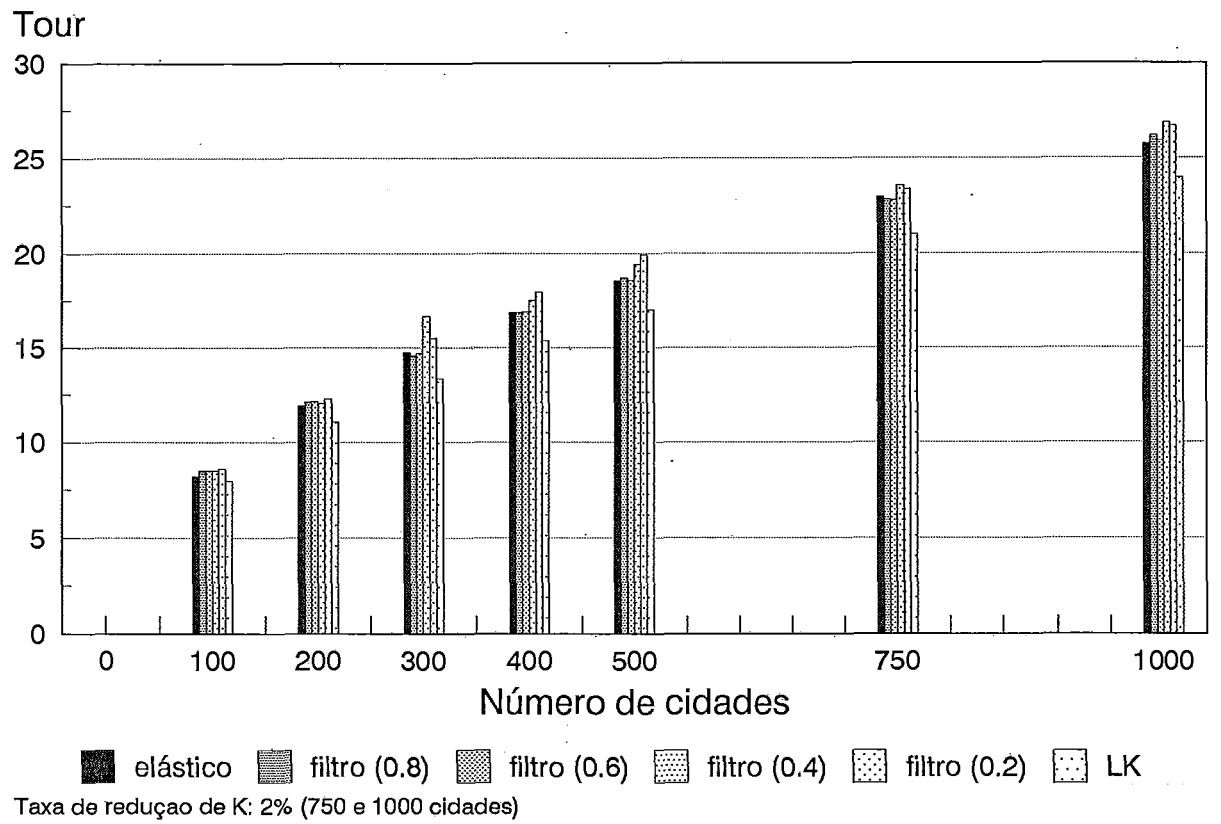


Figura V.9: Menores comprimentos de *tour* obtidos

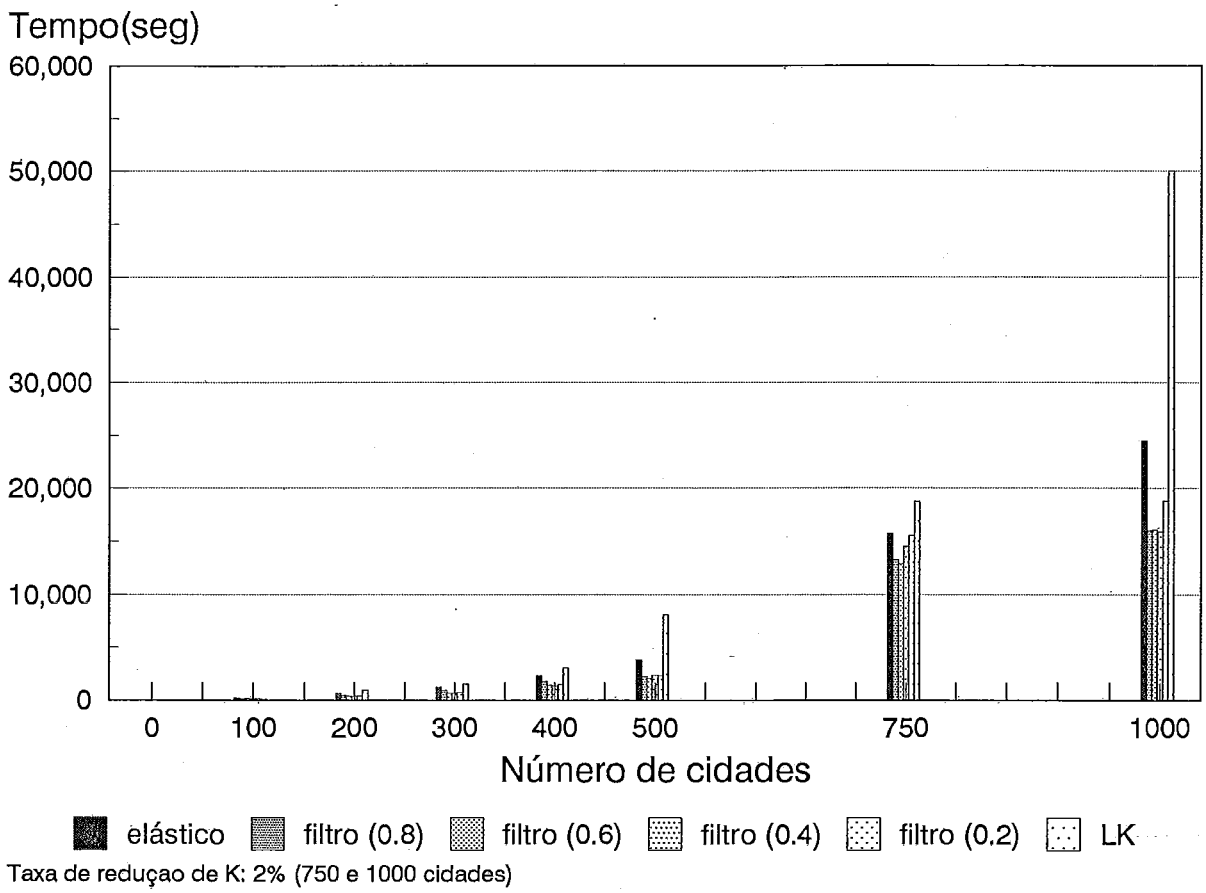


Figura V.10: Tempos de execução

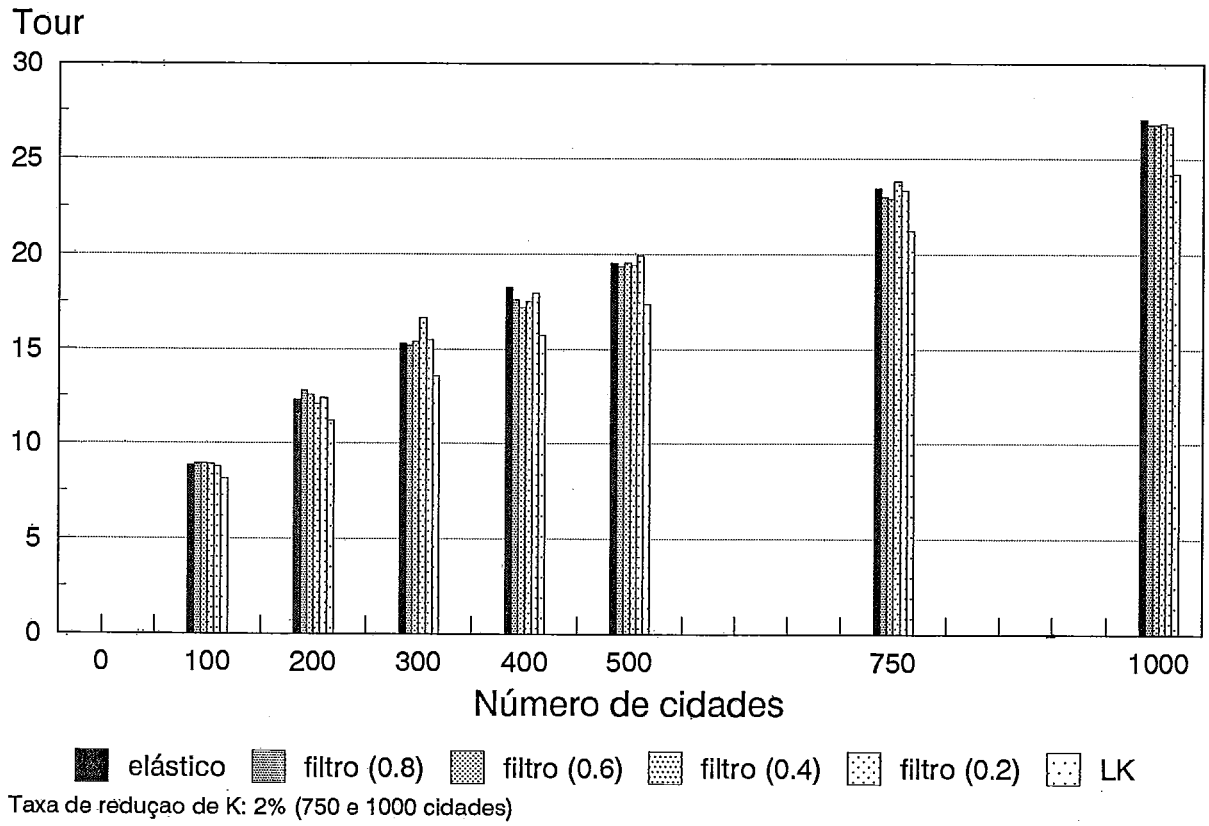


Figura V.11: Percursos médios

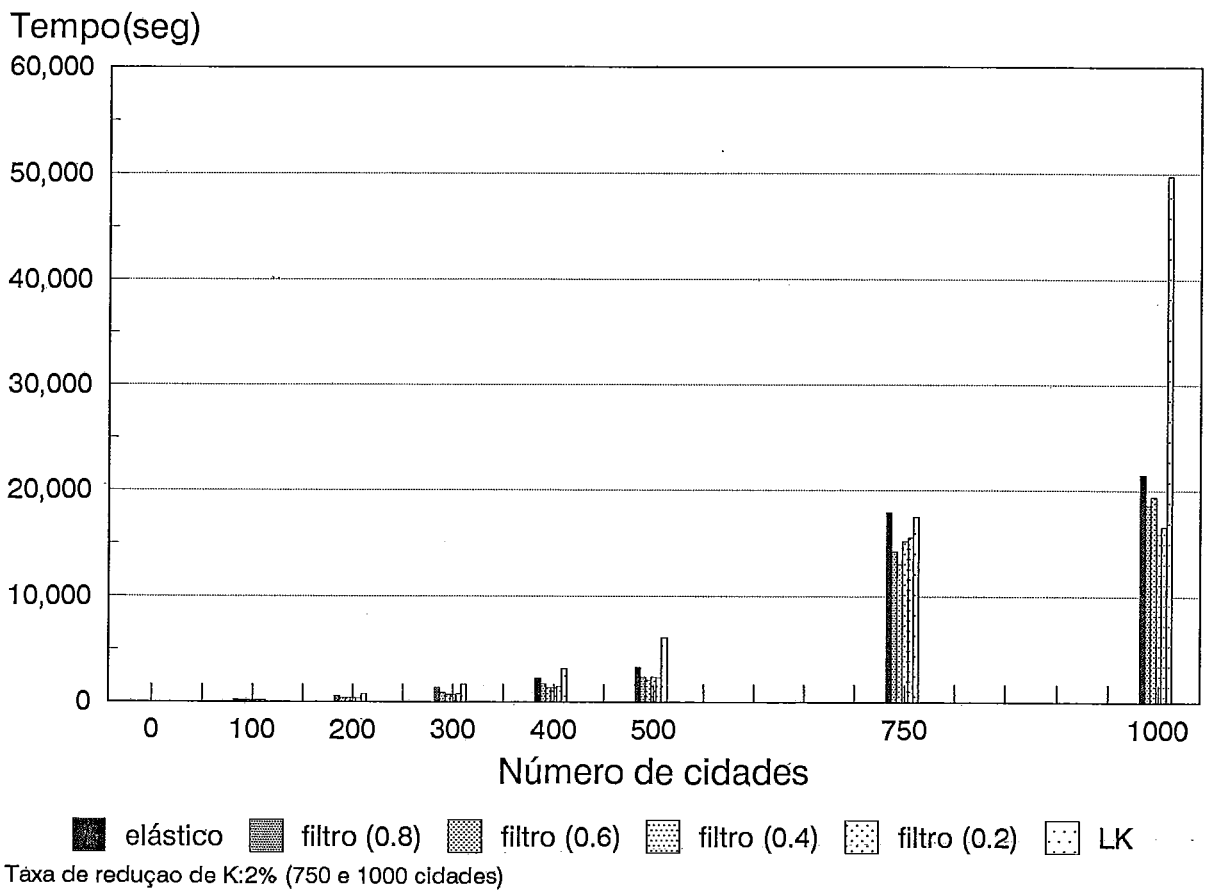


Figura V.12: Tempos médios

Capítulo VI

Conclusões

O Algoritmo Elástico consiste em mais uma das inúmeras heurísticas existentes para obtenção de soluções aproximadas para o problema do Caixeiro Viajante.

A avaliação e comparação de seu desempenho com o de outros algoritmos, através de testes com conjuntos de até 200 cidades, foram realizadas em [23]. Nesse trabalho, o Algoritmo Elástico foi comparado com algoritmos famosos como Redes Neuronais, "*Simulated Annealing*" e Algoritmos Genéticos. Dentre todos, esse algoritmo apresentou um bom desempenho, quanto ao comprimento de *tour* obtido, perdendo apenas para o Algoritmo Genético.

Com a realização dos testes descritos no capítulo V, verificou-se que o custo de *tours* obtidas pelo Filtro é bastante próximo do custo daquelas obtidas pelo Elástico, sendo inclusive, para alguns conjuntos de cidades, um pouco melhor. Isso reforça o argumento de que "filtrar" as influências exercidas pelas cidades sobre os pontos não acarreta necessariamente em uma perda no custo da *tour*.

Observando-se os gráficos apresentados na seção V.5, pode-se notar que, tanto o Algoritmo Elástico quanto o Filtro, apresentam comprimentos de *tour* piores do que o algoritmo LK. As figuras V.11 e V.12 exibem respectivamente os percursos médios e os seus tempos médios de execução, obtidos para cada conjunto de cidades. A razão entre o tempo de execução do Filtro e os tempos de execução dos outros dois algoritmos, cresce rapidamente com o incremento do número de cidades, principalmente em relação ao algoritmo LK, reforçando o argumento de

que o Filtro pode ser útil para a aplicação em problemas onde a restrição de tempo é muito importante. Sua vantagem é apresentar, em melhores tempos, soluções para o problema do Caixeiro Viajante bastante próximas daquelas obtidas pelo Elástico.

Tanto o Algoritmo Elástico quanto o Filtro mostraram-se bastante sensíveis ao conjunto de parâmetros ao qual eles estão subordinados. Nenhum dos trabalhos relacionados com o Algoritmo Elástico apresentou, até hoje, análises teóricas significativas acerca das relações entre esses parâmetros. Entretanto, alguma contribuição neste sentido pode ser encontrada em [26]. Nesta tese, algumas análises realizadas através de avaliações experimentais, relacionadas com o Algoritmo Elástico e o Filtro, foram realizadas. Tais análises possibilitaram a estimativa de valores, tomados neste conjunto de parâmetros, de forma a acarretar melhoras no custo da *tour* obtida, em detrimento do tempo de execução requerido, ou vice-versa.

Ganhos obtidos em tempos de execução consistem em uma restrição muito importante para algumas aplicações práticas do problema do Caixeiro Viajante. Reinelt [25] discute em seu trabalho a aplicação de algumas técnicas aproximativas para a solução do problema do Caixeiro Viajante, quando aplicado a problemas práticos, como a perfuração e gravação de placas de circuitos integrados. Neste trabalho, Reinelt explora as características do problema, quando definido geometricamente, para a obtenção de heurísticas mais rápidas.

Como geralmente, as perfurações realizadas nessas placas — correspondentes aos nós do problema do Caixeiro Viajante — são em grande número, torna-se muito difícil a obtenção de soluções muito boas em tempos curtos. Como a restrição de tempo é muito forte, prefere-se muitas vezes soluções de pior custo, porém, em tempos significativamente rápidos. Neste sentido, o Filtro constitui uma técnica razoável para a obtenção de soluções para esses problemas.

Referências Bibliográficas

[1] Angéniol, B., Vaubois, G. C., Texier, J., "Self-Organizing Feature Maps and the Travelling Salesman Problem," *Neural Networks, Vol. 1, pp. 289-293,* (1988).

[2] Barbosa, V., C., " Redes Neuronais e Simulated Annealing como Ferramentas para Otimização Combinatória", *Investigatión Operativa, Vol. 1, no2, pp. 125-142,* (1989).

[3] Boeres, M. C. S., "Uma Avaliação Experimental de uma Versão Paralela de Simulated Annealing," *Tese de Mestrado, COPPE-UFRJ* (1990).

[4] Burr, D. J., "An Improved Elastic Net Method for the Traveling Salesman Problem," *Proc. Int. Conf. on Neural Networks, I-69–76, San Diego, CA* (1988).

[5] Carvalho, L. A. V., " Síntese de Redes Neuronais com Aplicações à Representação do Conhecimento e à Otimização, " *Tese de Doutorado, COPPE-UFRJ* (1989).

[6] Croes, A., " A Method for Solving Traveling Salesman Problems, " *Opns. Res.,* **5**, *pp. 791-812* (1958).

[7] Durbin, R., Willshaw, D., " An Analogue Approach to the Traveling Salesman Problem using an Elastic Net Method, " *Nature,* **326**, *pp. 689-691* (1987).

[8] Durbin, R., Szeliski, R., Yuille, A., "An Analysis of the Elastic Net Approach to the Traveling Salesman Problem, " *Neural Computation,* **1**, *pp. 348-358* (1989).

[9] Érdi, P., Barna, G., " Self-Organizing Mechanism for the Formation of Ordered Neural Mappings," *Biological Cybernetics* **51**, *pp. 93-101* (1984).

[10] Fort, J.C., "Solving a Combinatorial Problem via Self-Organizing Process: An Application of the Kohonen Algorithm to the Traveling Salesman Problem, " *Biological Cybernetics* **59**, *pp. 33-40* (1988).

[11] Goodhill, G. J., Willshaw, D. J., "Application of the Elastic Net Algorithm to the formation of Ocular Dominance Stripes, " *Network,* **1**, *pp. 41-59* (1990).

[12] Häussler, A. F., Malsburg, C. von der, "Development of retinotopic projections: an analytical treatment, " *J. Theoret. Neurobiol.,* **2**, *pp. 47-73* (1983).

[13] Hopfield, J. J., " Neurons with graded response have collective computational properties like those of two states neurons, " *Proc. Natl. Acad. Sci., USA,* **81**, *pp. 3088-3092* (1984).

[14] Hopfield, J. J., Tank, D. W.," Neural computation of decisions in optimization problems, " *Biol. Cybern.,* **52**, *pp. 141-152* (1985).

[15] Hueter, G. J., " Solution of the Traveling Salesman Problem with an Adaptive Ring, " *Proc. Int. Conf. on Neural Networks, I-85–92, San Diego, CA* (1988).

[16] Kirkpatrick, S., Gellat, C. D., Vecchi, M. P. "Optimization by Simulated Annealing, " *Science, Vol. 220, no 4598, pp. 71-680* (1983).

[17] Kohonen, T., "Self-organized formation of topologically correct feature maps, " *Biol. Cybern.,* **43**, *pp. 59-69* (1982).

[18] Lin, S., " Computer Solutions of the Traveling Salesman Problem, ". *BST J,* **44**, *pp. 2245-2269* (1965).

[19] Lin, S., Kernighan, B. W., " An Effective Heuristic for the Traveling Salesman Problem, ". *Operations Research,* **21**, *pp. 498-516* (1973).

[20] Malsburg, Ch. Von der, Willshaw, D., " How to label nerve cells so that they can interconnect in an ordered fashion," *Proc. Natl. Acad. Sci., Vol 74, no 11, pp. 5176-5178, USA* (1977).

[21] Matsuyama, Y., " Competitive Self-Organization and Combinatorial Optimization: Aplications to the Traveling Salesman Problem, " *Proc. Int. Conf. on Neural Networks, III-819–824, San Diego, CA* (1988).

[22] Pessoa, L. A. F. C., " Aprendizado Não-Supervisionado em Redes Neuronais, " *Tese de Mestrado, COPPE-UFRJ*, (1990).

[23] Peterson, C., " Parallel Distributed Approaches to Combinatorial Optimization: Benchmark Studies on Traveling Salesman Problem, " *Neural Computation* **2**, *pp. 261-269* (1990).

[24] Prestige, M. C., Willshaw, D., " On a role for competition in the formation of patterned neural connexions, " *Proc. R. Soc. Lond. B.* **190**, *pp. 77-98* (1975).

[25] Reinelt, G., " Fast Heuristics for Large Geometric Traveling Salesman Problems, " *Anwendungsbezogene Optimierung und Steuerung, Universität Augsburg, Report no 185* (1989).

[26] Simmen, M., " Parameter Sensitivity of the Elastic Net Approach to the Traveling Salesman Problem, " *Neural Computation* **3**, *pp. 363-374* (1991).

[27] Simic, P., " Statistical mechanics as the underlying theory of elastic and neural optimization, " *NETWORK: Comp. Neural Syst., I(1), pp. 89-103* (1990).

[28] Takeuchi, A., Amari, S., " Formation of Topographic maps and Columnar microstructures in nerve fields, " *Biol. Cybern.* **35**, *pp. 63-72* (1979).

[29] Willshaw, D., Malsburg, Ch Von der, " How patterned neural connections can be set up by self-organization, " *Proc. R. Soc. Lond. B.* **194**, *pp. 431-445* (1976).

[30] Wong, W. S., Funka-Lea C. A., " An Elastic Net Solution to Obstacle Avoidance Tour Planning , " *Proc. Int. Conf. on Neural Networks, III-799–804, San Diego, CA* (1988).

[31] Yuille, A. L., " Generalized Deformable Models, Statistical Physics, and Matching Problems, " *Neural Computation* **2**, *pp. 1-24* (1990).