Paradigmas de algoritmos na solução de problemas de busca multidimensional

Pedro J. de Rezende
Renato Fileto
{rezende|fileto}@dcc.unicamp.br

Relatório Técnico DCC–95-01

Janeiro de 1995

Paradigmas de algoritmos na solução de problemas de busca multidimensional*

Pedro J. de Rezende†
Renato Fileto
{rezende|fileto}@dcc.unicamp.br

Sumário

Neste trabalho, descrevemos alguns problemas de busca em subespaços (*range search*) e soluções encontradas na literatura, a partir das quais são identificados dois paradigmas de algoritmos: árvores de partição e linearização. Estes paradigmas levam a algumas das soluções assintoticamente ótimas e a soluções generalizadas para determinados tipos de problemas de busca em subespaços. A abordagem adotada possibilita uma visão geral e abrangente das técnicas envolvidas, encarando diversas soluções distintas como variações de uma mesma concepção básica. Realçamos ainda possibilidades de aplicação e questões abertas, levantando os possíveis impactos da evolução das pesquisas a respeito de buscas multidimensionais.

Abstract

In this paper, we describe some range search problems and solutions found in the literature, from which two algorithmic paradigms are identified: partition trees and linearization. These paradigms lead to some asymptotically optimal solutions and to generalized solutions for certain types of range search problems. The adopted approach propitiates a general view of the techniques, in such a way that several distinct solutions are perceived as variants of the same basic conception. We emphasize the applications and related subjects of research, giving a flavor of the possible effects that may result from the evolution of the methods for multidimensional search.

1 Introdução

Este artigo considera soluções para alguns problemas de busca em várias dimensões, os quais pertencem a uma classe mais ampla, conhecida como problemas de busca. A literatura a respeito, no enfoque de projeto de algoritmos e geometria computacional, é farta e inclui muitos resultados recentes. Diversas soluções têm sido apresentadas com diferentes relações entre as medidas de desempenho assintótico, cada vez mais próximas das cotas inferiores estabelecidas.

*Trabalho desenvolvido com o apoio financeiro do CNPq e da Fapesp.

†Departamento de Ciência da Computação, IMECC, Caixa Postal 6065, UNICAMP, 13081-970 Campinas SP.

O valor dos resultados relativos a problemas de busca multidimensional está tanto no suporte que oferecem à solução de outros problemas nas áreas de algoritmos, geometria computacional e computação gráfica, quanto na possibilidade de derivação de novos métodos de acesso multi-chave e a dados geométricos dispersos num espaço multidimensional (ou aperfeiçoamento dos métodos já existentes), os quais são de grande valia em aplicações a áreas como bancos de dados, geoprocessamento e projeto auxiliado por computador (*CAD*).

Analisando algumas das soluções de um ponto de vista geral, de modo a destacar aspectos em comum entre elas, pode-se identificar algumas abordagens gerais relacionadas à concepção básica dessas soluções, que classificamos como paradigmas de algoritmos. Alguns desses paradigmas são ilustrados nas soluções descritas neste trabalho. Eles permitem estabelecer uma visão unificadora e encarar diversas soluções como manifestações de um mesmo fundamento racional, propiciando uma certa sistematização.

Este artigo é organizado em 5 seções. Na seção 2, são fornecidos os conceitos e classificações básicos, a fim de localizar a área de estudo e prover os fundamentos necessários à compreensão dos resultados descritos posteriormente. Na seção 3 são descritas algumas soluções para os problemas considerados, de modo a ilustrar as técnicas envolvidas, evidenciando alguns paradigmas de algoritmos. Na seção 4 esses paradigmas são discutidos em maior grau de generalização, procurando formalizar as definições e apresentar o potencial de utilização. Finalmente, na seção 5 são resumidas as contribuições. Em todas as seções são oferecidas referências adicionais detalhando os assuntos e/ou estendendo o tema central.

2 Conceitos básicos

2.1 Problemas de busca

Problemas de busca aparecem freqüentemente em computação e diversos problemas podem ser reduzidos a busca ou vistos como instâncias desta classe de problemas. Definições para problemas de busca em geral podem ser encontradas em [Knu73, Ben79, Mel84, IRKV88]. Esses problemas envolvem uma base de dados a respeito dos quais deseja-se extrair alguma informação ou determinar se satisfazem determinadas propriedades.

Há dois tipos de consultas possíveis em problemas de busca:

Consultas unitárias: A consulta é feita *somente uma vez*, dispensando qualquer tipo de pré-processamento que pudesse agilizar as buscas.

Exemplos: calcular a dominância dos pontos de um conjunto, conjunto dominante e par mais próximo (enunciados destes problemas podem ser encontrados em [PS85]).

Consultas em modo repetitivo: São realizadas *repetidas consultas* sobre uma mesma coleção de dados, cada qual relativa a algum *objeto de consulta*, obtido de um espaço de objetos de consulta possivelmente infinito. Em cada consulta, deseja-se identificar (ou simplesmente contar) os objetos da base de dados satisfazendo a uma dada propriedade com relação ao objeto de consulta. Pode ser útil arranjar a informação em uma estrutura organizada e mantê-la armazenada, a fim de facilitar as buscas, o que, se

realizado, implica num custo de memória e pré-processamento (e, no caso de estruturas dinâmicas, de reorganização). Esses custos são compensados pela agilização dos tempos das buscas, que são realizadas repetidas vezes.

Exemplos: busca em subespaço (*range search*)[1], dominância de *um* ponto e k vizinhos mais próximos a *um* dado ponto.

Outra classificação para as consultas diz respeito ao tipo de resposta a ser retornado. Problemas de enumeração são aqueles onde a resposta consiste em enumerar os elementos da base de dados satisfazendo uma propriedade. Há outras possibilidades como, por exemplo, retornar simplesmente a quantidade desses elementos de dados.

A análise de desempenho dos métodos de busca com consultas em modo repetitivo para um conjunto P de N objetos de dados armazenados, usualmente leva em consideração as quatro (ou, muitas vezes apenas as três primeiras) medidas de custo abaixo, que são funções de N:

1. tempo de consulta: o tempo de processamento de uma consulta;
2. uso de memória: a quantidade de espaço de memória requerida pela estrutura pré-processada;
3. tempo de pré-processamento: o tempo necessário para montar a estrutura de dados de busca;
4. tempo de atualização: pode ser o tempo despendido para inserir um elemento, o tempo para remover um elemento ou o tempo para atualizar (reorganizar) a estrutura após um certo número de inserções e remoções.

As medidas de complexidade acima podem ser tanto de pior caso quanto de caso médio e neste trabalho, trataremos sempre de medidas de complexidade de pior caso.

O tempo de consulta no caso de problemas de enumeração (veja seção 2.2) costuma ser denotado por $(f(N)+k)$ onde $f(N)$, denominado tempo de busca, é o tempo despendido para procurar na estrutura pré-processada os elementos satisfazendo à propriedade em questão e k é o tamanho da saída (quantidade de elementos satisfazendo à propriedade), que é o tempo mínimo necessário para retorná-la. Esta separação permite contornar o limite inferior trivial *de pior caso* $O(N)$ para o tempo total de consulta, possibilitando uma comparação mais acurada entre os métodos nas situações em que $k \in o(N)$.

As duas medidas de complexidade mais importantes em grande parte das situações são o tempo de consulta e o uso de memória. Os diversos métodos existentes para um mesmo problema proporcionam diversas relações distintas entre essas grandezas. Assim, é comum referir-se aos métodos com tempo de consulta e uso de memória proporcionais a $f(N) + k$ e $g(N)$, respectivamente, como sendo da ordem de $O(f(N) + k, g(N))$. O tempo de pré-processamento algumas vezes pode ser desprezado, por ser efetuado uma única vez, ao passo que a memória dispensada para a estrutura pré-processada fica comprometida durante todo o tempo em que se desejar efetuar consultas.

[1] Como não há uma tradução estabelecida na literatura em português para o termo *range*, das traduções de *range search*: "busca em sub-região" e "busca em subespaços", preferimos esta última. Para uma definição do problema veja seção 2.2.

2.2 O problema de busca em subespaços

Um desafio persistente na área de problemas de busca é conseguir soluções eficientes para buscas em várias dimensões. Dois termos comumente empregados para designar busca em várias dimensões são busca multidimensional (*multidimensional search*) [DL76] e busca geométrica (*geometric retrieval*) [CY85, PS85].

Este trabalho trata especificamente de alguns problemas de busca em várias dimensões denominados problemas de busca em subespaços (*range search*). Na versão *consultas em modo repetitivo*, um problema de busca em subespaços, com objetos de dados pontuais pode ser formalmente enunciado da seguinte maneira:

Busca em subespaços:

Sejam $\gamma_1, \gamma_2, \ldots, \gamma_d$ espaços topológicos totalmente ordenados.

Seja $\Gamma = \gamma_1 \times \gamma_2 \times \ldots \times \gamma_d$ o produto cartesiano desses espaços (espaço multidimensional ou espaço d-dimensional).

Seja $P \subseteq \Gamma$; $P = \{p_1, p_2, \cdots, p_N\}$ um conjunto finito de pontos dispersos em Γ, constituindo a base de dados (conjunto de dados armazenados). Denota-se um ponto $p \in P$ por $p = (x_1, x_2, \ldots, x_d)$ com $x_i \in \gamma_i$ $(1 \leq i \leq d)$.

Seja $\Sigma \subseteq 2^{\Gamma}$ um conjunto de subespaços de busca, definidos através de alguma coleção de predicados.

Deseja-se pré-processar P de modo a responder eficientemente a consultas relativas a quais pontos de P estão contidos em um subespaço de busca $\sigma \in \Sigma$.

Os tipos das respostas a serem fornecidas permitem uma classificação das consultas em três tipos:

Consultas de enumeração: Enumerar os pontos em $P \cap \sigma$.

Consultas de contagem: Determinar o número de pontos em $P \cap \sigma$ ($|P \cap \sigma|$).

Consultas de vacuosidade: Determinar se algum ponto de P está contido em σ ($P \cap \sigma = \emptyset$?).

O conjunto Σ dos subespaços de busca permitidos é o principal determinante de um problema de busca em subespaços. Uma vez definido o par (P, Σ), pode-se pré-processar a base de dados P, produzindo uma estrutura de dados T para auxiliar a responder consultas para qualquer $\sigma \in \Sigma$.

Neste trabalho lida-se com o caso em que os objetos de dados são *pontos* em um espaço multidimensional e explora-se variações da formulação do problema caracterizadas pelos formatos de subespaços de busca admitidos nas consultas[2]. Entre os problemas clássicos

[2] Argumentos a favor desta abordagem, do ponto de vista da aplicabilidade, podem ser encontrados em [Fil94].

encontrados na literatura para $\gamma_i = \mathbb{R}$ $(1 \leq i \leq d)$ encontram-se (veja ilustrações dos subespaços para $d = 2$ na figura 1):

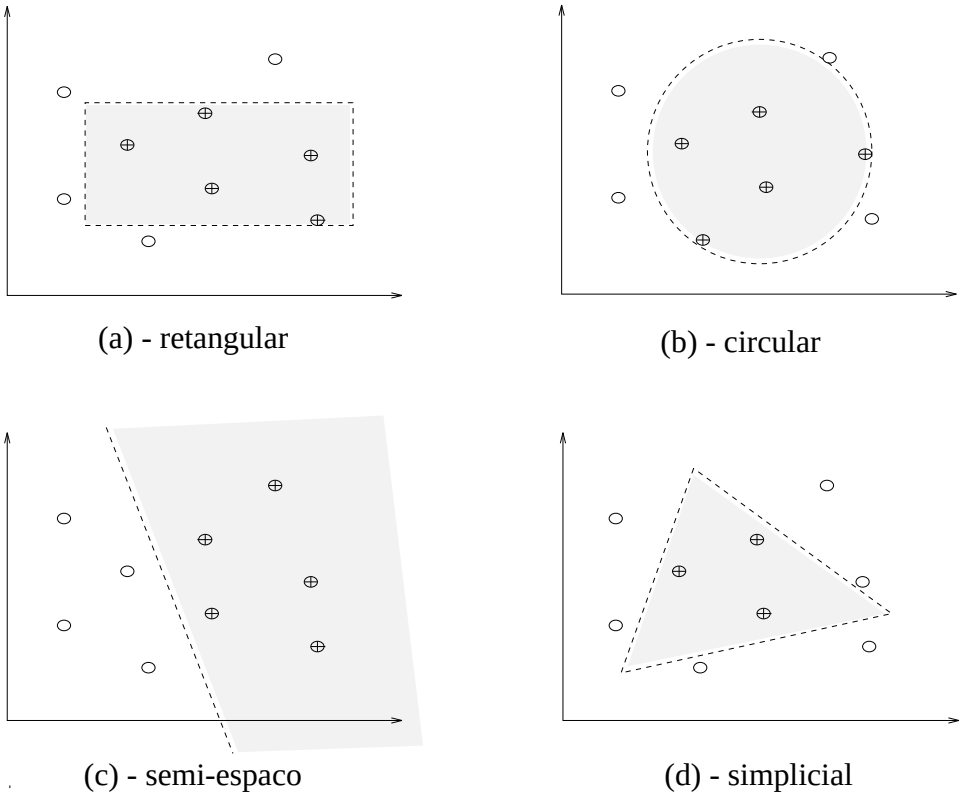

Figura 1: Alguns tipos de subespaços de busca no plano euclideano.

- busca em subespaço ortogonal (também conhecida por busca em subespaço hiper-retangular): O subespaço de busca é $\sigma = \sigma_1 \times \sigma_2 \times \ldots \times \sigma_d$, o produto cartesiano de intervalos $\sigma_i = [a_i, b_i] \subseteq \gamma_i$ $(1 \leq i \leq d)$, denotando um hiper-retângulo, com as arestas paralelas aos eixos coordenados (figura 1-a).

- busca em subespaço circular: O subespaço de busca é um hiper-disco $\sigma(o, r)$ de centro o e raio r arbitrários[3] (figura 1-b).

- busca em semi-espaço: O subespaço de busca é um semi-espaço $\sigma(h)$ do espaço Γ limitado por um hiperplano h (figura 1-c).

- busca em subespaço simplicial: O subespaço de busca é delimitado por um simplexo[4]. A figura 1-d ilustra um simplexo em $\mathbb{R}^2$ (um triângulo).

[3] Busca em subespaço circular usualmente refere-se a este problema em duas dimensões. Embora a maioria dos resultados existentes na literatura digam respeito ao caso bi-dimensional, neste trabalho, usamos este termo para designar o problema em dimensão arbitrária.

[4] Um δ-simplexo em $\mathbb{R}^d$ $(\delta \leq d)$ é um conjunto δ-dimensional formado pela envoltória convexa de $\delta + 1$ pontos $\{q_0, q_1, \ldots, q_\delta\}$, seus vértices, que definem $\delta + 1$ vetores linearmente independentes. O interior de

O subespaço de busca ortogonal constitui uma das formas mais simples pois neste caso o espaço Γ pode ser particionado em faixas para as quais as buscas podem ser realizadas mais facilmente. Busca em subespaço circular, por outro lado, é um problema mais difícil, pelo fato de sua fronteira ser não linear.

Os problemas de busca em semi-espaços e busca em subespaços simpliciais estão bastante relacionados desde sua definição até as técnicas de solução a eles empregadas. Freqüentemente, as mesmas soluções se aplicam a ambos os problemas. Eles são de fundamental importância, pois suas soluções resultam em melhoria de algoritmos para outros problemas em geometria computacional e teoria da computação como, por exemplo, *ray shooting* [dBHO+91, AM92b, Sch92] e programação linear [Mat93b].

Uma resenha das soluções descritas na literatura para esses formatos de subespaços de busca clássicos pode ser encontrada em [Fil94].

Um formato de subespaços de busca mais genérico, englobando os formatos descritos acima e que recentemente começou a ser considerado na literatura [AM92a, Mat93a] é o dos subespaços determinados por inequações polinomiais. Os subespaços polinomiais são aqueles determinados por uma única inequação polinomial, isto é:

Subespaços polinomiais:

Seja $f(x,a) = f(x_1,\cdots,x_d,a_1,\cdots,a_K)$ um polinômio de grau limitado em $d+K$ variáveis, onde $x = (x_1,x_2,\cdots,x_d)$ é um ponto no espaço $\mathbb{R}^d$ e $a = (a_1,a_2,\cdots,a_K)$ é um vetor de K parâmetros, sendo K uma constante.

O conjunto Σ_f de subespaços de busca polinomiais determinados por f é dado por $\Sigma_f = \{\sigma_f(a) \mid a \in \mathbb{R}^K\}$, onde $\sigma_f(a) = \{x \in \mathbb{R}^d \mid f(x,a) \geq 0\}$.

O polinômio f especifica os tipos de subespaços de busca a serem considerados (por exemplo, discos, cônicas, cilindros, etc.) e o vetor de parâmetros a determina um subespaço particular do tipo considerado.

Por meio de conjunções, disjunções e complementos de inequações polinomiais, obtém-se diversos formatos de subespaços, que são denominados subespaços semi-algébricos. O conjunto dos subespaços semi-algébricos admite obviamente subespaços com fronteira não linear, como é o caso de subespaços circulares. Portanto, uma solução para busca em subespaços semi-algébricos se aplica a uma ampla variedade de subespaços de busca. É importante que os subespaços tenham complexidade de descrição limitada, isto é, que possam ser descritos por um número limitado de polinômios de grau limitado.

Nas próximas seções, são analisadas algumas soluções para busca em semi-espaços e para buscas simpliciais e suas generalizações para busca em subespaços semi-algébricos, sob o enfoque de paradigmas de algoritmos.

---

um δ-simplexo é o conjunto de todos os pontos da forma $\sum_{i=0}^{\delta}(\alpha_i \cdot q_i)$; sendo $\sum_{i=0}^{\delta}\alpha_i = 1$ e $\alpha_i > 0$ para todo $0 \leq i \leq \delta$. Um δ-simplexo pode ser considerado como a interseção de $(\delta+1)$ semi-espaços onde os hiperplanos limitantes estão em posição geral. Quando $\delta = d$ chamamos o δ-simplexo simplesmente de simplexo.

Para exemplificar o potencial de aplicação das soluções para problemas de busca em subespaços em geral, basta notar que uma consulta a uma base de dados, expressa por um conjunto de restrições nos atributos pode ser vista como uma busca em subespaço, de formato determinado pela natureza das restrições, se convenientemente mapeada num sistema de coordenadas.

3 Algumas soluções para busca em subespaços

Nesta seção, são esboçadas algumas das soluções encontradas na literatura para busca em semi-espaços, simplexos e discos no plano, as quais são essencialmente de uma mesma natureza. O objetivo é apresentar as técnicas envolvidas, identificando alguns paradigmas de algoritmos, que são discutidos mais genericamente na seção 4.

3.1 Solução através da árvore de Willard

A árvore de Willard [Wil82] foi originalmente proposta na solução de busca em regiões poligonais no plano não necessariamente conexas ou limitadas. Entretanto, ela pode ser mais facilmente compreendida se analisada como um método de busca em semi-espaços no plano, conforme descrição apresentada por Matoušek em [Mat93a].

Seja P um conjunto de N pontos no plano. Por simplicidade, assuma os pontos de P em posição geral, isto é, nenhum subconjunto de P com três pontos é colinear. Seja r_1 uma reta de direção arbitrária dividindo o conjunto de pontos P ao meio. Pelo teorema do *ham-sandwich cut* [Ede87] existe uma reta r_2, tal que r_1 e r_2 dividem o plano em 4 regiões, $\mathcal{R}_1$, $\mathcal{R}_2$, $\mathcal{R}_3$ e $\mathcal{R}_4$, cada qual contendo $1/4$ dos pontos de P (por simplicidade de descrição, considere $N = 4^c$, com $c > 1$ inteiro), como ilustra a figura 2-a.

Seja $\sigma(h)$ um semi-plano limitado por uma reta h. É fácil ver que a reta h intercepta o interior de no máximo 3 regiões $\mathcal{R}_i$ $(1 \leq i \leq 4)$, ou seja uma das regiões fica totalmente dentro ou totalmente fora do semi-plano $\sigma(h)$, como ilustra a figura 2-b. No processo de busca pelos pontos contidos em $\sigma(h)$, esta região pode ser tratada diretamente. Para as demais regiões é necessário processamento adicional para determinar quais pontos contidos no interior das mesmas pertencem ao semi-plano de busca $\sigma(h)$.

Esta técnica de decomposição do problema permite ums economia de 25% do processamento de uma consulta e aplicando-a recursivamente nas regiões decompostas, tem-se a seguinte equação de recorrência para o tempo de consulta de pior caso:

$$T(N) = \overline{k} + 3T(N/4) \in O(N^{\log_4 3} + k) \subset O(N^{0,7925} + k) \qquad (1)$$

onde $\overline{k}$ é o número máximo de pontos contidos em uma região de uma partição e k é o tempo de enumeração da resposta.

Este método sugere uma estrutura de dados T em forma de árvore, armazenando a informação das partições aplicadas recursivamente, de modo a dar suporte ao processamento das consultas. A cada nó v não folha é associada uma região $\mathcal{R}(v)$ do espaço, um conjunto de pontos $\mathcal{P}(v) = P \cap \mathcal{R}(v)$ com os pontos de P nela contidos e uma partição $\Phi(v)$ de $\mathcal{R}(v)$ em sub-regiões de acordo com $\mathcal{P}(v)$. Para cada sub-região $\varphi \in \Phi(v)$ é atribuído um

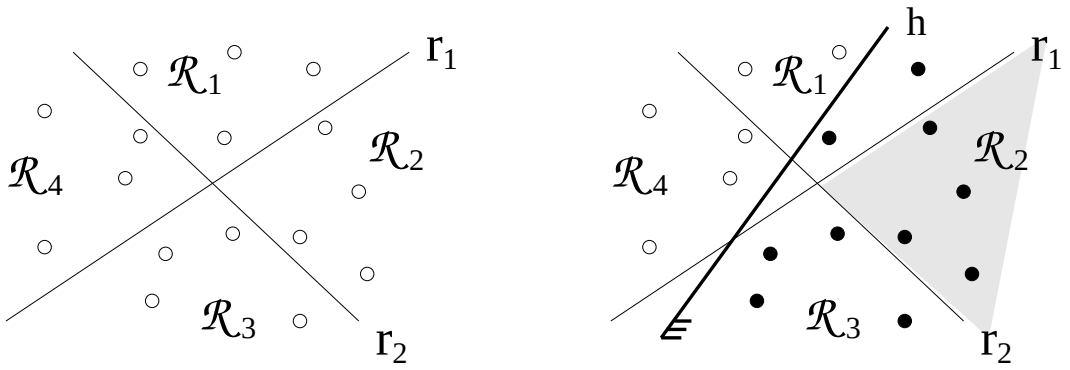

Figura 2: A decomposição de um conjunto de pontos no plano em 4 partes iguais (a) e uma região totalmente contida em um semi-espaço (b).

filho a v. Ao nó raiz é associado todo o espaço Γ e o conjunto de pontos P. Aos nós folha são associadas regiões contendo uma quantidade de pontos de P inferior a um valor pré-estabelecido. A figura 3 ilustra uma partição do plano e a árvore correspondente.

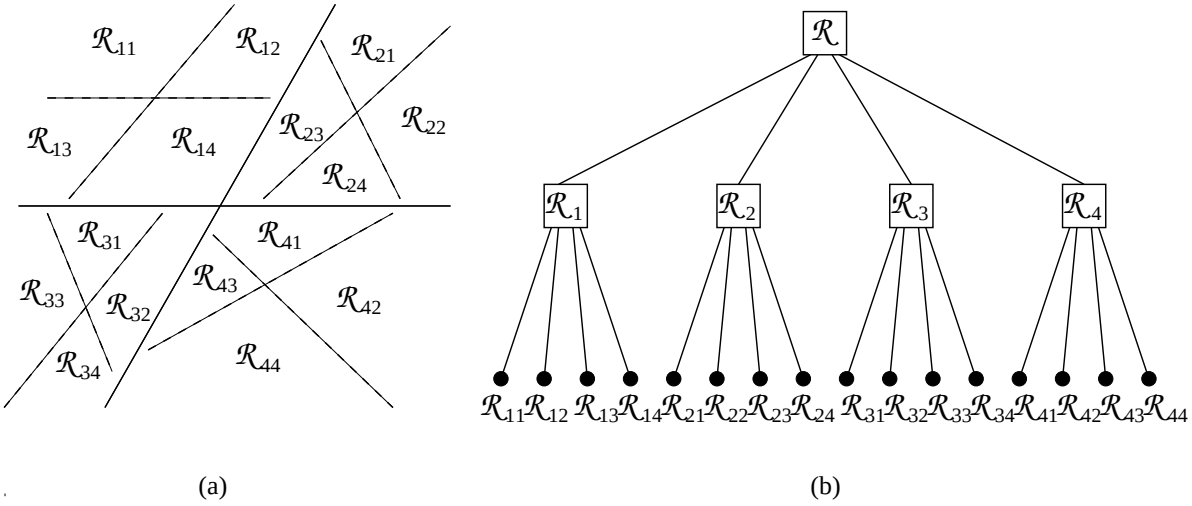

Figura 3: A solução de Willard: uma partição do plano (a) e a árvore correspondente (b).

O processo de busca começa pela raiz e descende recursivamente nos nós relativos às regiões interceptadas por h. Para problemas de contagem, pode-se armazenar em cada nó v o número de pontos contidos em $\mathcal{R}(v)$. No caso de problemas de enumeração os pontos de P ficam armazenados nas folhas de T associadas com as regiões que os contêm. Para enumerar os pontos contidos em uma região $\mathcal{R}(v)$ é necessário descer até as folhas descendentes de v, mas o tempo despendido neste processo é proporcional ao tamanho da saída, de modo que este processamento adicional não aumenta o tempo de busca assintótico (um processamento

de mesma ordem é necessário para enumerar a saída)[5].

Willard originalmente propôs uma partição determinada por duas retas mais um número configurável de semi-retas. Utilizando esta partição para construir uma estrutura de árvore conforme a descrita acima, tem-se as seguintes medidas de complexidade para busca em semi-espaços e regiões poligonais no plano (vide [Wil82]):

Consulta:	$O(N^{\log_6 4} + k) \subset O(N^{0,774} + k)$
Memória:	$O(N)$
Pré-processamento:	$O(N^2)$ (pior caso)
	$O(N \cdot \log^2 N)$ (caso médio)

3.2 Solução através da árvore de conjugação de Edelsbrunner e Welzl

A solução de Edelsbrunner e Welzl [EW86] é um aperfeiçoamento da árvore de Willard, diferindo pelo tipo de partição utilizada. A estrutura é denominada árvore de conjugação pelo fato das regiões relativas a nós adjacentes na árvore não serem particionadas independentemente, mas possuírem uma linha divisória em comum. Uma partição do plano segundo este critério e a correspondente árvore de conjugação são ilustradas na figura 4.

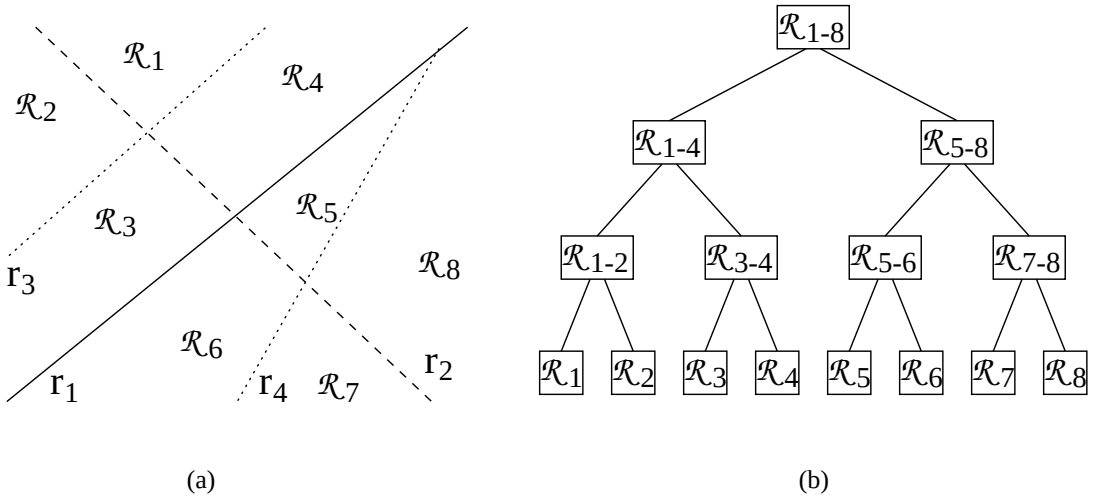

Figura 4: A solução de Edelbrunner e Welzl: a partição do plano (a) e a árvore de conjugação correspondente (b).

Com esta estrutura consegue-se reduzir os tempos de consulta e pré-processamento em relação à solução de Willard:

Consulta:	$O(N^{\log_2 (1+\sqrt{5})-1} + k) \subset O(N^{0,695} + k)$
Memória:	$O(N)$
Pré-processamento:	$O(N \cdot \log N)$ (pior caso)

[5] Observa-se aqui o princípio básico do paradigma de algoritmos denominado filtragens sucessivas (*filtering search*), descrito por Chazelle em [Cha86].

Note que a concepção básica da solução não mudou, embora se tenha empregado outro tipo de partição para determinar a conformação física da árvore (número de filhos por nó) e o método de construção da estrutura. Esta forma abstrata da solução é denominada árvore de partição e os tipos específicos de partições que podem ser empregados são chamados esquemas de partição. Existem também soluções baseadas em árvores de partição para algumas dimensões mais altas ($d = 2$, 3 e 4) e mesmo para dimensões arbitrárias onde a dimensão d é um parâmetro. Outras questões relativas ao paradigma de árvores de partição são discutidas na seção 4.1.

3.3 Soluções para busca em subespaço circular

Os métodos de busca em semi-espaços e buscas simpliciais, como os descritos acima, podem ser empregados para solucionar problemas de busca em subespaços semi-algébricos. Existem técnicas para se estabelecer correspondências entre problemas de busca em subespaços semi-algébricos e problemas de busca em semi-espaços ou simplexos em dimensões mais altas. Busca em subespaço circular é um dos problemas que podem ser solucionados por este caminho.

Um disco $\sigma(o, r)$ em $\mathbb{R}^2$, de centro em $o = (o_x, o_y)$ e raio r é o conjunto de pontos (x, y) satisfazendo a inequação polinomial $(x - o_x)^2 + (y - o_y)^2 \leq r^2$. Re-arranjando os termos desta inequação, temos:

$$\sigma(o, r) = \{(x, y) \in \mathbb{R}^2 \mid r^2 - {o_x}^2 - {o_y}^2 + 2o_x x + 2o_y y - x^2 - y^2 \geq 0\}. \tag{2}$$

Considere o hiperplano:

$$h = \{(t_1, t_2, t_3, t_4) \in \mathbb{R}^4 \mid r^2 - {o_x}^2 - {o_y}^2 + t_1 + t_2 + t_3 + t_4 = 0\}.$$

Mapeando cada ponto $(x, y) \in \mathbb{R}^2$ num ponto $(t_1, t_2, t_3, t_4) \in \mathbb{R}^4$ com $t_1 = 2o_x x$, $t_2 = 2o_y y$, $t_3 = -x^2$ e $t_4 = -y^2$ temos que:

$$(x, y) \in \sigma(o, r) \iff (t_1, t_2, t_3, t_4) \in \sigma(h) \tag{3}$$

onde:

$$\sigma(h) = \{(t_1, t_2, t_3, t_4) \in \mathbb{R}^4 \mid r^2 - {o_x}^2 - {o_y}^2 + t_1 + t_2 + t_3 + t_4 \geq 0\} \tag{4}$$

O problema de busca em subespaço circular no plano é portanto reduzido a busca em semi-espaços em 4 dimensões. Este é um exemplo do paradigma conhecido como linearização. A dimensão do espaço para onde é mapeado o problema é chamada de dimensão da linearização.

Note porém que a linearização descrita acima *não* é a de menor dimensão para discos em $\mathbb{R}^2$. A transformação de elevação ao parabolóide possibilita uma linearização de dimensão 3, mapeando os pontos de $\mathbb{R}^2$ em pontos do parabolóide $z = x^2 + y^2$. Seja $\zeta : \mathbb{R}^2 \to \mathbb{R}^3$ a transformação que leva o ponto (x, y) no ponto $\zeta(x, y) = (x, y, x^2 + y^2)$ em $\mathbb{R}^3$. O leitor pode verificar que pontos cocirculares são mapeados por ζ em pontos coplanares e que a imagem do interior do disco $\sigma(o, r)$ em $\mathbb{R}^2$ é a intersecção do parabolóide com um dos semi-espaços determinados pelo plano que contém a imagem da fronteira de $\sigma(o, r)$.

Portanto, determinar os pontos de P contidos no disco $\sigma(o, r)$ equivale a determinar os pontos de $\zeta(P)$ contidos nesse semi-espaço. Esta transformação é ilustrada na figura 5.

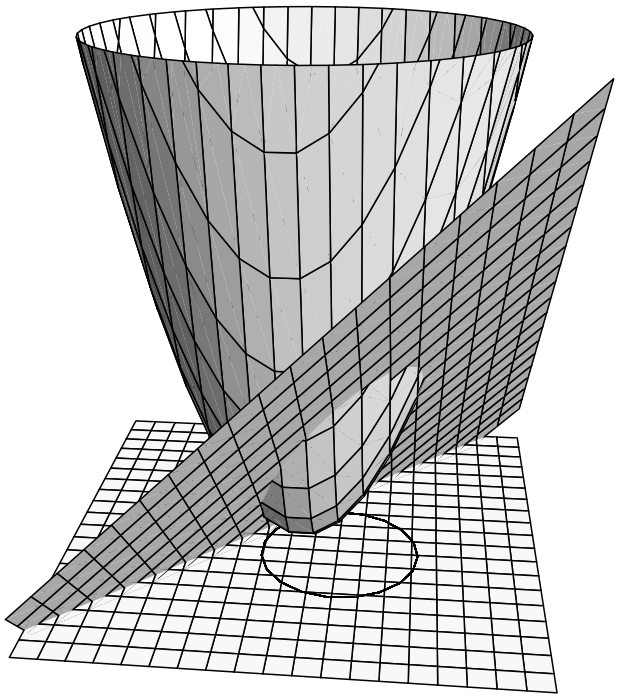

Figura 5: A transformação de elevação ao parabolóide.

Na seção 4.2 apresentamos uma definição mais formal e genérica do paradigma de linearização e relatamos alguns resultados provenientes de sua aplicação.

4 Paradigmas de algoritmos

A análise e o projeto de soluções baseando-se em paradigmas permite uma melhor compreensão dos problemas e suas soluções, promovendo a aplicação de técnicas sistemáticas e o aparecimento de soluções para problemas mais gerais.

Na seção anterior foram identificados os paradigmas de árvores de partição e linearização, os quais aparecem com freqüência (às vezes sem ser mencionado que se tratam de abordagens gerais) em soluções descritas na literatura para problemas de busca em subespaços. Nesta seção, discutimos algumas questões relativas aos paradigmas de árvores de partição e descrevemos mais formalmente e de maneira generalizada o paradigma de linearização.

4.1 Árvores de partição

Uma estrutura do tipo árvore de partição aparece pela primeira vez no trabalho de Willard [Wil82] e o conceito é formalizado por Matoušek [Mat91, Mat93a]. As soluções baseadas em árvores de partição propõem diversos esquemas de partição distintos. A eficiência de uma solução utilizando um determinado esquema de partição é determinada pelo número máximo de regiões da partição que podem ser cortadas pela fronteira de um subespaço de busca, assim como pela distribuição dos pontos da base de dados entre essas regiões.

Yao e Yao [YY85] provam a existência de esquemas de partição tais que qualquer hiperplano cruza um número sub-linear de suas regiões para cada dimensão fixa, o que tem motivado a pesquisa de soluções baseadas em árvores de partição para várias dimensões. Matoušek [Mat93a] oferece uma resenha dos resultados para semi-espaços e simplexos (incluindo não somente as soluções baseadas em árvores de partição). Observa-se que a abordagem de árvores de partição é dominante na quantidade de trabalhos publicados e nos desempenhos obtidos. As soluções baseadas neste paradigma estão entre as mais eficientes para o caso geral de busca em semi-espaços e buscas simpliciais com uso de memória quase linear[6]. Entretanto, elas são superadas por outras soluções de concepções distintas, em casos específicos como enumeração dos pontos contidos em semi-espaços e simplexos, para dimensões pequenas.

Recentemente, árvores de partição têm sido empregadas também para busca em subespaços semi-algébricos em geral, mas os resultados ainda são escassos [AM92a, Mat93a]. Dentre as questões abertas está a determinação de esquemas de partição que possibilitem soluções eficientes para estes tipos de subespaços.

4.2 Linearização

O paradigma de linearização [AM92a, Mat93a] consiste em estabelecer uma correspondência entre um problema de busca em subespaços de algum formato arbitrário em d dimensões e o problema de busca em *semi-espaços* num espaço de dimensão $D > d$. O termo linearização provém do fato da fronteira de um semi-espaço ser um hiperplano, dado por uma equação linear.

Aplicando este paradigma é possível tratar problemas de busca em subespaços onde a fronteira do subespaço de busca é constituída de variedades algébricas dadas por equações polinomiais de grau limitado. Os subespaços de busca são usualmente determinados por conjunções e/ou disjunções de inequações polinomiais do tipo[7] $f(x_1, x_2, \cdots, x_d) \geq 0$, onde $x_1, x_2, \cdots, x_d$ são as coordenadas de um ponto $x \in \Gamma$ (veja definição de subespaços de busca polinomiais no final da seção 2.2).

Considere um problema de busca em subespaço polinomial em $\mathbb{R}^d$. Para empregar o paradigma de linearização é necessário determinar uma função mapeando cada ponto $(x_1, \cdots, x_d) \in \mathbb{R}^d$ em algum ponto $(t_1, \cdots, t_D) \in \mathbb{R}^D$, de tal forma que buscas em subespaços semi-algébricos em $\mathbb{R}^d$ correspondam a buscas em *semi-espaços* em $\mathbb{R}^D$.

[6] Mais formalmente, "quase linear" significa da ordem de $O(N^{1+\varepsilon})$, onde o valor de ε depende da relação de compromisso estabelecida entre o uso de memória e o tempo de busca.

[7] As inequações também podem ser estritas ($f(x_1, x_2, \cdots, x_d) > 0$). Os subespaços por elas determinados recebem o mesmo tratamento.

Função de linearização:

Uma função $\zeta : \mathbb{R}^d \to \mathbb{R}^D$, com $D > d$ é chamada função de linearização se:

1. ζ é contínua;
2. dado um subespaço polinomial $\sigma_f \in \Sigma_f$, o subespaço afim gerado pela imagem de sua fronteira $\langle \zeta(\partial\sigma_f) \rangle$ é um hiperplano de dimensão $D-1$.

Decorre da continuidade de ζ que a imagem $\zeta(\sigma_f)$ de um subespaço de busca polinomial $\sigma_f \in \Sigma_f$ está inteiramente contida em um dos *semi-espaços* de $\mathbb{R}^D$ delimitados por $\langle \zeta(\partial\sigma_f) \rangle$, donde se pode estabelecer a correspondência do problema de busca em subespaços original em $\mathbb{R}^d$ com busca em *semi-espaços* em $\mathbb{R}^D$, isto é:

$$x \in \sigma_f \iff \zeta(x) \in \zeta(\sigma_f) \tag{5}$$

Agarwal e Matoušek [AM92a, Mat93a] concluíram, a partir do trabalho de Yao e Yao [YY85], que todo polinômio é linearizável, isto é, admite uma função de linearização.

Um método geral para se obter uma linearização é fazer corresponder a cada uma das funções coordenadas de $\zeta(x)$ um monômio nas coordenadas de x aparecendo no polinômio f. Porém, como foi ilustrado na seção 3.3, este método produz linearizações de alta dimensão (exponencial no grau de f), tornando tais soluções muito dispendiosas.

Na procura de uma função de linearização, pode-se agrupar os monômios de f de diversas maneiras distintas, para formar as funções coordenadas de $\zeta(x) = (\zeta_1(x), \cdots, \zeta_D(x))$. O fundamental é que para se ter uma linearização, a fronteira de σ_f determinada pelo polinômio f, quando mapeada por ζ em $\mathbb{R}^D$, deve determinar um hiperplano denotado por uma equação da forma:

$$\sum_{i=1}^{D} \alpha_i \zeta_i(x) = 0 \tag{6}$$

onde cada $\zeta_i(x)$ é um polinômio em x obtido de somas de monômios de f e os α_i's são coeficientes determinados a partir dos coeficientes de f.

Um problema em aberto é conseguir abordagens algorítmicas para se obter linearizações da menor dimensão possível (ou até mesmo de dimensão polinomial no grau de f).

Aplicação e extensões

A aplicação direta do paradigma de linearização possibilita as melhores soluções conhecidas para busca em alguns tipos de subespaços determinados por polinômios. Para outros tipos de subespaços, entretanto, são obtidos algoritmos mais eficientes generalizando métodos de busca em subespaços de fronteira linear (tipicamente semi-espaços e simplexos) para o caso

não linear. Por exemplo, para busca em subespaço circular no plano, a aplicação direta de linearização em 3 dimensões resulta em um algoritmo com tempo de busca $O(N^{2/3})$ com memória linear. Todavia, substituindo semi-espaços por círculos em algumas soluções para busca em semi-espaços (por exemplo [Mat91]) e realizando as generalizações necessárias, pode-se obter tempo de busca $O(\sqrt{N})$ com memória linear.

Algoritmos inspirados em linearização e baseados em generalização de soluções propostas para busca em semi-espaços e para buscas simpliciais são abordados por Agarwal e Matoušek [AM92a, Mat93a]. Eles propõem uma estrutura para busca em subespaços determinados por polinômios com uso de memória e tempo de pré-processamento linear, permitindo efetuar buscas em tempo $O(N^{1-1/b+\varepsilon})$, onde b é uma constante tal que $d \leq b \leq 2d-3$ e ε é uma constante arbitrária.

5 Conclusões

Foram abordados alguns problemas de busca em subespaços com dados pontuais discretos, segundo o enfoque de projeto de algoritmos eficientes e geometria computacional. Descreveram-se soluções da literatura, partindo de alguns dos primeiros métodos para solucionar busca em semi-espaços e buscas simpliciais e chegando até às abordagens recentemente propostas para busca em subespaços de natureza mais geral, com a fronteira determinada por polinômios de grau limitado.

Do estudo dessas soluções concluiu-se que elas se baseiam numa mesma forma abstrata de estrutura de dados denominada árvore de partição. A árvore de partição provê a conformação básica da estrutura de dados física e as diretrizes para o processamento das consultas, constituindo um paradigma de algoritmos. As soluções particulares diferem pelo esquema de partição utilizado e em alguns detalhes de seu funcionamento.

Outro paradigma de algoritmos identificado neste artigo é a linearização de subespaços de busca. Por meio de técnicas de linearização consegue-se solucionar buscas em subespaços com fronteira não linear, a partir de soluções concebidas para subespaços com fronteira linear.

Estes paradigmas constituem abordagens mais sistemáticas para os problemas e agem como agentes unificadores das soluções ou, pelo menos, conjuntos de soluções. Além desses paradigmas, podem ser identificados outros, nos diversos tipos de problemas de busca em subespaços. Em [Fil94] é realizado um tratamento mais extensivo de problemas de busca em subespaços e dos paradigmas de algoritmos aplicados nas soluções dos mesmos.

Uma das questões que podem ser levantadas é o quanto paradigmas de algoritmos podem contribuir na produção de soluções gerais para problemas de busca em subespaços. Cabe notar então que não existe uma solução boa para todos os tipos de subespaços. Em diversos casos particulares as soluções gerais perdem em eficiência e simplicidade para as soluções específicas e em alguns casos, existem inclusive soluções específicas mais eficientes que os limites inferiores estabelecidos para o caso geral. Porém, algumas soluções genéricas podem ser úteis e, para algumas subclasses de problemas, as primeiras soluções não triviais foram conseguidas com a aplicação (às vezes implícita) de paradigmas de algoritmos, como é o caso das soluções para busca em subespaços com a fronteira determinada por polinômios.

Os temas de pesquisa em aberto relacionados com a aplicação dos resultados obtidos para problemas de busca em subespaços incluem:

- generalização das soluções de modo a permitir o tratamento de objetos de dados não pontuais [Sam90, Cox91];
- dinamização [Ove83, CT92];
- adequação das estruturas de dados propostas aos mecanismos de armazenamento disponíveis [Sil81, IKO88, OSdBvK90, SO90];
- implementação e experimentação das soluções.

Referências

[AM92a] P. K. Agarwal and J. Matoušek. On range search with semialgebraic sets. In *Proc. 17th MFCS - Internat. Sympos. on Mathematical Found. of Computer Science*, volume 629 of *Lecture Notes in Computer Science*, pages 1–13. Springer-Verlag, 1992.

[AM92b] P. K. Agarwal and J. Matoušek. Ray shooting and parametric search. In *Proc. 24th Annu. ACM Sympos. Theory Comput.*, pages 517–526, 1992.

[Ben79] J. L. Bentley. Decomposable searching problems. *Inform. Process. Lett.*, 8:244–251, 1979.

[Cha86] B. Chazelle. Filtering search: a new approach to query-answering. *SIAM J. Comput.*, 15:703–724, 1986.

[Cox91] F. S. Cox Jr. Análise de métodos de acesso a dados espaciais aplicados a sistemas gerenciadores de bancos de dados. Master's thesis, Depto. de Ciência da Computação da Universidade Estadual de Campinas, Dezembro 1991.

[CT92] Y.-J. Chiang and R. Tamassia. Dynamic algorithms in computational geometry. *Proceedings of the IEEE*, 80(9):1412–1434, 1992.

[CY85] R. Cole and C. K. Yap. Geometric retrieval problems. *Inform. Control*, 63:39–57, 1985.

[dBHO+91] M. de Berg, D. Halperin, M. Overmars, J. Snoeyink, and M. van Kreveld. Efficient ray shooting and hidden surface removal. In *Proc. 7th Annu. ACM Sympos. Comput. Geom.*, pages 21–30, 1991.

[DL76] D. P. Dobkin and R. J. Lipton. Multidimensional searching problems. *SIAM J. Comput.*, 5:181–186, 1976.

[Ede87] H. Edelsbrunner. *Algorithms in Combinatorial Geometry*, volume 10 of *EATCS Monographs on Theoretical Computer Science*. Springer-Verlag, Heidelberg, West Germany, 1987.

[EW86] H. Edelsbrunner and E. Welzl. Halfplanar range search in linear space and $O(n^{0.695})$ query time. *Inform. Process. Lett.*, 23:289–293, 1986.

[Fil94] R. Fileto. Busca em subespaços em várias dimensões. Master's thesis, Depto. de Ciência da Computação da Universidade Estadual de Campinas, 1994. To appear.

[IKO88] C. Icking, R. Klein, and T. Ottmann. Priority search trees in secondary memory. In *Proc. Internat. Workshop Graph-Theoret. Concepts Comput. Sci. (WG '87)*, volume 314 of *Lecture Notes in Computer Science*, pages 84–93. Springer-Verlag, 1988.

[IRKV88] S. S. Iyengar, N. S. V. Rao, R. L. Kashyap, and V. K. Vaishnavi. Multidimensional data structures: Review and outlook. *Advances in Computers*, 27:69–119, 1988.

[Knu73] D. E. Knuth. *Sorting and Searching*, volume 3 of *The Art of Computer Programming*. Addison-Wesley, Reading, MA, 1973.

[Mat91] J. Matoušek. Efficient partition trees. In *Proc. 7th Annu. ACM Sympos. Comput. Geom.*, pages 1–9, 1991.

[Mat93a] J. Matoušek. Geometric range search. preprint, 1993.

[Mat93b] J. Matoušek. Linear optimization queries. *J. Algorithms*, 14:432–448, 1993. The results combined with results of O. Schwarzkopf also appear in *Proc. 8th ACM Sympos. Comput. Geom.*, 1992, pages 16–25.

[Mel84] K. Melhorn. *Data Structures an Algorithms 3: Multidimensional Searching and Computational Geometry*. Springer-Verlag, Berlin, 1984.

[OSdBvK90] M. H. Overmars, M. H. M. Smid, M. T. de Berg, and M. J. van Kreveld. Maintaining range trees in secondary memory, part I: partitions. *Acta Inform.*, 27:423–452, 1990.

[Ove83] M. H. Overmars. *The design of dynamic data structures*, volume 156 of *Lecture Notes in Computer Science*. Springer-Verlag, 1983.

[PS85] F. P. Preparata and M. I. Shamos. *Computational Geometry: an Introduction*. Springer-Verlag, New York, NY, 1985.

[Sam90] H. Samet. *The Design and Analysis of Spatial Data Structures*. Addison-Wesley Publishing Company, 1990.

[Sch92] O. Schwarzkopf. Ray shooting in convex polytopes. In *Proc. 8th Annu. ACM Sympos. Comput. Geom.*, pages 286–295, 1992.

[Sil81] Y. V. Silva Filho. Optimal choice of discriminators in a balanced k-d binary search tree. *Inform. Process. Lett.*, 13:67–70, 1981.

[SO90] M. H. M. Smid and M. H. Overmars. Maintaining range trees in secondary memory, part II: lower bounds. *Acta Inform.*, 27:453–480, 1990.

[Wil82] D. E. Willard. Polygon Retrieval. *SIAM Journal on Computing*, 11(1):149–165, 1982.

[YY85] A. C. Yao and F. F. Yao. A general approach to d-dimensional geometric queries. In *Proc. 17th Annu. ACM Sympos. Theory Comput.*, pages 163–168, 1985.

Relatórios Técnicos – 1992

92-01 **Applications of Finite Automata Representing Large Vocabularies,** *C. L. Lucchesi, T. Kowaltowski*

92-02 **Point Set Pattern Matching in d-Dimensions,** *P. J. de Rezende, D. T. Lee*

92-03 **On the Irrelevance of Edge Orientations on the Acyclic Directed Two Disjoint Paths Problem,** *C. L. Lucchesi, M. C. M. T. Giglio*

92-04 **A Note on Primitives for the Manipulation of General Subdivisions and the Computation of Voronoi Diagrams,** *W. Jacometti*

92-05 **An (l, u)-Transversal Theorem for Bipartite Graphs,** *C. L. Lucchesi, D. H. Younger*

92-06 **Implementing Integrity Control in Active Databases,** *C. B. Medeiros, M. J. Andrade*

92-07 **New Experimental Results For Bipartite Matching,** *J. C. Setubal*

92-08 **Maintaining Integrity Constraints across Versions in a Database,** *C. B. Medeiros, G. Jomier, W. Cellary*

92-09 **On Clique-Complete Graphs,** *C. L. Lucchesi, C. P. Mello, J. L. Szwarcfiter*

92-10 **Examples of Informal but Rigorous Correctness Proofs for Tree Traversing Algorithms,** *T. Kowaltowski*

92-11 **Debugging Aids for Statechart-Based Systems,** *V. G. S. Elias, H. Liesenberg*

92-12 **Browsing and Querying in Object-Oriented Databases,** *J. L. de Oliveira, R. de O. Anido*

Relatórios Técnicos – 1993

93-01 **Transforming Statecharts into Reactive Systems,** *Antonio G. Figueiredo Filho, Hans K. E. Liesenberg*

93-02 **The Hierarchical Ring Protocol: An Efficient Scheme for Reading Replicated Data,** *Nabor das C. Mendonça, Ricardo de O. Anido*

93-03 **Matching Algorithms for Bipartite Graphs,** *Herbert A. Baier Saip, Cláudio L. Lucchesi*

93-04 **A lexBFS Algorithm for Proper Interval Graph Recognition,** *Celina M. H. de Figueiredo, João Meidanis, Célia P. de Mello*

93-05 **Sistema Gerenciador de Processamento Cooperativo,** *Ivonne. M. Carrazana, Nelson. C. Machado, Célio. C. Guimarães*

93-06 **Implementação de um Banco de Dados Relacional Dotado de uma Interface Cooperativa,** *Nascif A. Abousalh Neto, Ariadne M. B. R. Carvalho*

93-07 **Estadogramas no Desenvolvimento de Interfaces,** *Fábio N. de Lucena, Hans K. E. Liesenberg*

93-08 **Introspection and Projection in Reasoning about Other Agents,** *Jacques Wainer*

93-09 **Codificação de Seqüências de Imagens com Quantização Vetorial,** *Carlos Antonio Reinaldo Costa, Paulo Lício de Geus*

93-10 **Minimização do Consumo de Energia em um Sistema para Aquisição de Dados Controlado por Microcomputador,** *Paulo Cesar Centoducatte, Nelson Castro Machado*

93-11 **An Implementation Structure for RM-OSI/ISO Transaction Processing Application Contexts,** *Flávio Morais de Assis Silva, Edmundo Roberto Mauro Madeira*

93-12 **Boole's conditions of possible experience and reasoning under uncertainty,** *Pierre Hansen, Brigitte Jaumard, Marcus Poggi de Aragão*

93-13 **Modelling Geographic Information Systems using an Object Oriented Framework,** *Fatima Pires, Claudia Bauzer Medeiros, Ardemiris Barros Silva*

93-14 **Managing Time in Object-Oriented Databases,** *Lincoln M. Oliveira, Claudia Bauzer Medeiros*

93-15 **Using Extended Hierarchical Quorum Consensus to Control Replicated Data: from Traditional Voting to Logical Structures,** *Nabor das Chagas Mendonça, Ricardo de Oliveira Anido*

93-16 $\mathcal{LL}$ – **An Object Oriented Library Language Reference Manual,** *Tomasz Kowaltowski, Evandro Bacarin*

93-17 **Metodologias para Conversão de Esquemas em Sistemas de Bancos de Dados Heterogêneos,** *Ronaldo Lopes de Oliveira, Geovane Cayres Magalhães*

93-18 **Rule Application in GIS – a Case Study,** *Claudia Bauzer Medeiros, Geovane Cayres Magalhães*

93-19 **Modelamento, Simulação e Síntese com VHDL,** *Carlos Geraldo Krüger e Mário Lúcio Côrtes*

93-20 **Reflections on Using Statecharts to Capture Human-Computer Interface Behaviour,** *Fábio Nogueira de Lucena e Hans Liesenberg*

93-21 **Applications of Finite Automata in Debugging Natural Language Vocabularies,** *Tomasz Kowaltowski, Cláudio Leonardo Lucchesi e Jorge Stolfi*

93-22 **Minimization of Binary Automata,** *Tomasz Kowaltowski, Cláudio Leonardo Lucchesi e Jorge Stolfi*

93-23 **Rethinking the DNA Fragment Assembly Problem,** *João Meidanis*

93-24 EGOLib — **Uma Biblioteca Orientada a Objetos Gráficos,** *Eduardo Aguiar Patrocínio, Pedro Jussieu de Rezende*

93-25 **Compreensão de Algoritmos através de Ambientes Dedicados a Animação,** *Rackel Valadares Amorim, Pedro Jussieu de Rezende*

93-26 GeoLab: **An Environment for Development of Algorithms in Computational Geometry,** *Pedro Jussieu de Rezende, Welson R. Jacometti*

93-27 **A Unified Characterization of Chordal, Interval, Indifference and Other Classes of Graphs,** *João Meidanis*

93-28 **Programming Dialogue Control of User Interfaces Using Statecharts,** *Fábio Nogueira de Lucena e Hans Liesenberg*

93-29 EGOLib – **Manual de Referência,** *Eduardo Aguiar Patrocínio e Pedro Jussieu de Rezende*

Relatórios Técnicos – 1994

94-01 **A Statechart Engine to Support Implementations of Complex Behaviour,** *Fábio Nogueira de Lucena, Hans K. E. Liesenberg*

94-02 **Incorporação do Tempo em um SGBD Orientado a Objetos,** *Ângelo Roncalli Alencar Brayner, Claudia Bauzer Medeiros*

94-03 **O Algoritmo KMP através de Autômatos,** *Marcus Vinícius A. Andrade e Cláudio L. Lucchesi*

94-04 **On Edge-Colouring Indifference Graphs,** *Celina M. H. de Figueiredo, João Meidanis, Célia Picinin de Mello*

94-05 **Using Versions in GIS,** *Claudia Bauzer Medeiros and Geneviève Jomier*

94-06 **Times Assíncronos: Uma Nova Técnica para o Flow Shop Problem,** *Hélvio Pereira Peixoto e Pedro Sérgio de Souza*

94-07 **Interfaces Homem-Computador: Uma Primeira Introdução,** *Fábio Nogueira de Lucena e Hans K. E. Liesenberg*

94-08 **Reasoning about another agent through empathy,** *Jacques Wainer*

94-09 **A Prolog morphological analyser for Portuguese,** *Jacques Wainer, Alexandre Farcic*

94-10 **Introdução aos Estadogramas,** *Fábio N. de Lucena, Hans K. E. Liesenberg*

94-11 **Matching Covered Graphs and Subdivisions of K_4 and $\overline{C_6}$,** *Marcelo H. de Carvalho and Cláudio L. Lucchesi*

94-12 **Uma Metodologia de Especificação de Times Assíncronos,** *Hélvio Pereira Peixoto, Pedro Sérgio de Souza*

Departamento de Ciência da Computação — IMECC
Caixa Postal 6065
Universidade Estadual de Campinas
13081-970 – Campinas – SP
BRASIL

reltec@dcc.unicamp.br