

versão enviada

# Armazenamento Distribuído de Mapas Vetoriais em Sistemas de Geoprocessamento

Claudia de Andrade Tocantins<sup>1,2</sup>

Marco Antonio Casanova<sup>2</sup>

<sup>1</sup>Pontifícia Universidade Católica - Rio  
R. Marquês de São Vicente, 225  
22453 Rio de Janeiro RJ - Brasil

<sup>2</sup>Centro Científico Rio - IBM Brasil  
Av. Presidente Vargas, 824/844 - 22º Andar  
20071 Rio de Janeiro RJ - Brasil

claudia@inf.puc-rio.br, casanova@vnet.ibm.com

## Abstract

This paper first presents a geographic object model and a general architecture for geographic object management. Then, it describes in detail concepts about vector maps, which are very important in the context of geoprocessing systems. Finally, it introduces distributed storage methods for vector maps as a way of parallelizing the execution of spatial operators and minimizing the cost of some geographic operations.

## 1 Introdução

Sistemas de Geoprocessamento, também chamados de Sistemas de Informação Geográfica, ou SIGs caracterizam-se por utilizar *dados georeferenciados*, ou seja, dados que descrevem objetos e fenômenos do mundo real associados a uma localização na superfície terrestre [Goo91]. Além do aspecto de georeferenciamento, os dados utilizados em SIGs caracterizam-se pelo grande volume, pela heterogeneidade e pelo fato de estarem potencialmente distribuídos em diversos locais. As operações típicas de SIGs, designadas no que se segue simplesmente por *operações geográficas*, são também muito mais complexas do que operações convencionais. Portanto, do ponto de vista de gerência e manipulação de dados, SIGs colocam desafios interessantes.

Diversos estudos têm sido desenvolvidos com o intuito de melhorar o desempenho dos SIGs. Eles abrangem áreas tais como a definição de arquiteturas e modelos de dados [WHM90, Goo92, Her92, RM92, FK93], o uso de novos métodos de acesso espacial [Sam90, BKSS90, Cox91, HS92, MCD94] e a síntese de algoritmos eficientes para a resolução de consultas espaciais [BKSS94, HS94, Wan93].

Este trabalho alinha-se com este esforço endereçando o problema de armazenamento distribuído de um grande volume de dados georeferenciados, organizados sob forma de mapas, de forma a paralelizar a execução de operadores espaciais e minimizar o custo de certas operações geográficas. Como suposições básicas, admite-se que: (1) os mapas estão representados vetorialmente; (2) o custo dos operadores espaciais domina o custo global das operações geográficas; (3) o custo de comunicação entre os processadores é

negligível, quando comparado com o custo de acesso a memória secundária e o custo das operações locais (o que é o caso por exemplo de um *cluster* de estações de trabalho).

O enfoque adotado é uma variante da estratégia genérica que consiste em distribuir, de forma balanceada, o armazenamento entre processadores, permitindo a paralelização das operações. As peculiaridades pertinentes a mapas e operações geográficas vem à tona em dois aspectos. Primeiro, o critério de distribuição entre processadores adotado obriga a divisão de mapas em fragmentos e leva em conta apenas a posição geográfica dos objetos. Isto significa que feições da superfície terrestre que estão geograficamente próximas tenderão a ter suas representações armazenadas no mesmo processador. Ou seja, proximidade geográfica induz proximidade física de armazenamento, o que representa uma heurística adequada para minimizar o custo das operações geográficas. Segundo, a compilação de operações geográficas sobre mapas em operações locais sobre os dados armazenados em cada processador exige uma análise cuidadosa da semântica das operações de forma a evitar que a distribuição dos objetos induza erros semânticos.

O trabalho está dividido da seguinte forma. A seção 2 apresenta um modelo para objetos geográficos e uma arquitetura genérica para gerenciadores desses objetos, incluindo variantes para o caso distribuído. A seção 3 define os conceitos de mapa e introduz a noção de critério de fragmentação para conjuntos de mapas. A seção 4 enfoca o tratamento de mapas vetoriais na arquitetura introduzida na seção 2 e discute métodos de armazenamento distribuído para mapas. Finalmente, a seção 5 contém as considerações finais do trabalho.

## 2 Preliminares

### 2.1 Um Modelo para Objetos Geográficos

Esta seção descreve brevemente o modelo para objetos geográficos adotado neste trabalho, seguindo em linhas gerais [CFS<sup>+</sup>94]. O modelo introduz os conceitos de geo-objeto, geo-campo, mapa de geo-objetos e representação de geo-campo.

Um *geo-objeto* corresponde a um elemento da realidade geográfica e tipicamente possui atributos *convencionais*, além de estar associado a uma ou mais *representações* geométricas geo-referenciadas, que são descrições da sua localização na superfície terrestre. Por exemplo, uma estrada é um geo-objeto com atributos convencionais indicando sua identificação, tipo, etc...

As representações de um geo-objeto não são suas componentes, mas de outros objetos, chamados de *mapas de geo-objetos*, conceito definido em detalhe na seção 3. Um exemplo pode ser visto na figura 1 que representa a hidrografia de uma região, um exemplo típico, onde as linhas *r1* e *r3* representam dois rios, a região *r2*, um rio e a região *l1*, um lago. Note que um mesmo rio pode ter várias representações, por exemplo, em mapas com escalas diferentes. Um mapa de geo-objetos pode, também, ter atributos convencionais, como escala, projeção cartográfica, etc...

Um *geo-campo* define a distribuição espacial de uma variável geográfica sobre uma região da superfície terrestre, como altitude ou tipo de solo. Quando a variável assume um conjunto finito de valores, o geo-campo normalmente é chamado um *campo temático*. Um geo-campo pode também ter atributos convencionais, como a data em que foi definido ou o órgão responsável pelo levantamento, e está associado a uma ou mais *representações* da distribuição da variável geográfica.

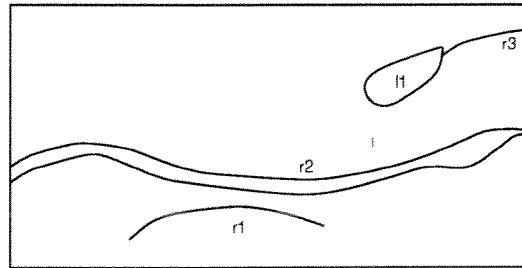


Figura 1: Um mapa de geo-objetos que representa a hidrografia de uma região.

Novamente, as representações de um geo-campo não são suas componentes, mas são tratadas como objetos separados de tipos especiais (ver [CFS+94]). Por exemplo, a altimetria de uma região pode ser representada por curvas de nível ou por uma imagem, onde a cor indica altitude. Em particular, um campo temático pode ser representado por um *mapa temático*, conceito definido em detalhe na seção 3.

Quando o contexto permitir, chamaremos mapas de geo-objetos e representações de geo-campos genericamente de *representações*.

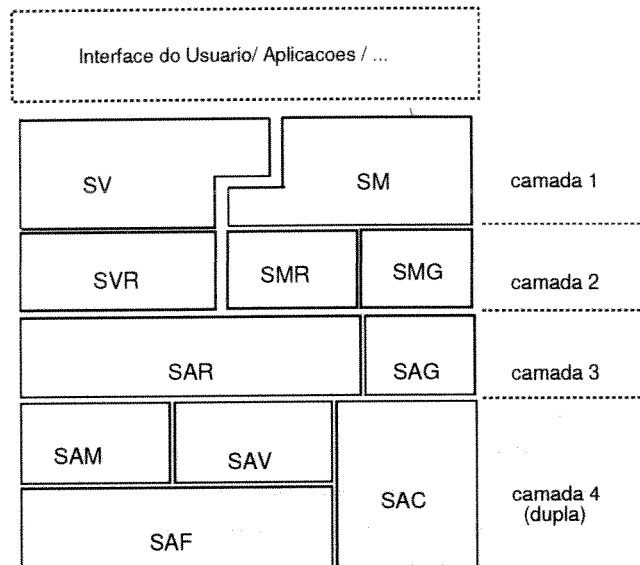
Note que há um relacionamento um-para-muitos entre um geo-campo e suas representações e um relacionamento muitos-para-muitos entre geo-objetos e mapas de geo-objetos (onde os geo-objetos possuem uma representação). Um *banco de dados geográfico* é, então, uma coleção de geo-objetos, geo-campos e representações e uma coleção de relacionamentos entre estes objetos.

Um banco de dados geográfico pode, em princípio, ter uma implementação distribuída do mesmo modo que um banco de dados convencional. Porém, implementações possivelmente mais eficientes poderão ser obtidas levando em conta duas características importantes de bancos de dados geográficos: (1) o armazenamento das representações tipicamente ocupa muito mais espaço do que o ocupado pelos atributos convencionais dos objetos; (2) as operações espaciais sobre estas representações são, em geral, muito mais complexas e dispendiosas do que operações envolvendo apenas os atributos convencionais.

O resto deste artigo discute principalmente a questão de armazenamento distribuído de bancos de dados geográficos, dando especial ênfase ao tratamento de mapas de geo-objetos e mapas temáticos.

## 2.2 Uma Arquitetura para um Gerenciador de Objetos Geográficos

Esta seção descreve brevemente uma arquitetura em camadas para um Gerenciador de Objetos Geográficos, ou brevemente GEOG, calcada nos conceitos introduzidos na seção 2.1. Uma descrição mais detalhada pode ser encontrada em [HMS<sup>+</sup>94]. A seção seguinte discute como adaptar a arquitetura para um ambiente distribuído.



SF	Subsistema de Armazenamento Físico	SM	Subsistema de Manipulação
SAM	Subsistema de Armazenamento Matricial	SMR	Subsistema de Manipulação de Representações
SAV	Subsistema de Armazenamento Vetorial	SMG	Subsistema de Manipulação de Geo-Objetos/Geo-Campos
SAC	Subsistema de Armazenamento Convencional	SV	Subsistema de Visualização
SAR	Subsistema de Armazenamento de Representações	SVR	Subsistema de Visualização de Representações

Figura 2: GeoG: arquitetura em camadas para um Gerenciador de Objetos Geográficos.

A figura 2 indica as camadas do GEOG e o nome por extenso dos subsistemas (não repetidos no texto por economia de espaço). A primeira camada do GEOG claramente separa as questões de visualização dos problemas de manipulação e oferece as abstrações de geo-objetos, geo-campos, e representações em geral (ou seja, lembrando, mapas de geo-objetos e representações de geo-campos). O SV oferece funções básicas para visualização de representações, que são tipicamente mais complexas do que nas aplicações tradicionais, enquanto que o SM oferece funções para definição e manipulação de geo-objetos, geo-campos e representações, indistintamente.

A segunda camada desce um nível na escala de abstração, separando a parte convencional da parte espacial. Ou seja, ela oferece separadamente serviços de visualização e manipulação de alto nível para representações, através do SVR e do SMR, respectivamente, e serviços de manipulação dos atributos convencionais dos objetos, através do

SMG.

A terceira camada cobre serviços de armazenamento e manipulação elementar para representações, via o SAR, e para atributos convencionais, através do SAG.

A última camada (dupla) implementa as formas de armazenamento - matricial, vetorial ou convencional - através respectivamente do SAM, SAV e SAC. Os dois primeiros subsistemas fazem uso do serviço de armazenamento de páginas físicas disponibilizada pelo SAF. O SAV trabalha com abstrações de *ponto*, *linha*, *polígono* e *região* implementadas através de estruturas espaciais, como por exemplo R-trees, V-trees e VR-trees [Med95].

## 2.3 Arquiteturas Distribuídas

Uma das formas de paralelizar as operações em um banco de dados, chamada de paralelismo particionado [DG92], consiste em particionar os dados e distribuí-los entre vários processadores. Na arquitetura clássica para sistemas de gerência de bancos de dados distribuídos, o armazenamento dos dados nos vários processadores é guiado por um *critério de distribuição*, que pode incluir replicação de fragmentos dos dados. O ciclo de execução de uma consulta ou atualização inclui operações locais, transferências de dados e composição de resultados parciais.

A adaptação desta idéia genérica à arquitetura em camadas descrita na seção anterior origina quatro possíveis arquiteturas distribuídas, dependendo do nível em que se introduz o critério de distribuição dos dados. A fronteira que as diferencia são sutis, variando o grau de abstração dos dados fragmentados e das operações paralelizáveis. A camada onde é aplicado o critério passa a ser responsável por mascarar para a camada superior o fato do armazenamento estar distribuído, controlando onde estão armazenados os fragmentos dos dados, bem como o processamento distribuído das operações oferecidas naquela camada.

Considere inicialmente que a distribuição se dá na primeira camada. Neste caso, podemos definir diversos critérios de distribuição para os geo-objetos, geo-campos e representações, de forma semelhante aos usados em bancos de dados convencionais. Porém, uma forma de distribuição baseada na fragmentação das representações oferece maior flexibilidade para paralelizar operações espaciais, já que permite armazenar fragmentos de representações em processadores distintos. O corte da arquitetura nesta camada resulta em um banco de dados distribuído fracamente acoplado, onde cada processador conhece a semântica dos dados e implementa operações de alto nível, tornando-os bastante independentes entre si.

A distribuição ao nível da segunda camada facilita a definição de critérios de distribuição para as representações de forma completamente independente dos critérios para a parte convencional dos objetos. Porém, cada um dos processadores ainda conhece a semântica das representações e pode operar sobre elas de forma independente. Esta arquitetura pode ser bastante adequada no contexto de um *cluster* de estações de trabalho, por exemplo, oferecendo um ambiente de armazenamento distribuído, interligado

por um sistema de comunicação de alto desempenho.

A distribuição dos dados na terceira camada (SAR e SAG) novamente é definida sobre representações e atributos convencionais, separadamente. Ela se distingue da situação anterior pelo nível de abstração dos operadores paralelizados, que é menor neste caso. Esse é o último ponto de corte em que os diversos processadores conhecerão a semântica das representações.

O último ponto de corte possível é a nível da camada de armazenamento elementar. Neste caso, apenas operações básicas sobre a geometria dos objetos vetoriais (no SAV), sobre estruturas matriciais (no SAM) ou sobre atributos convencionais (no SAC) são paralelizadas. Os processadores que armazenam os fragmentos destes objetos são completamente dependentes dos processos que implementam as camadas superiores.

### 3 Mapas Vetoriais

Dada a importância para sistemas de geoprocessamento, o restante deste trabalho aborda em mais detalhe mapas vetoriais, entendidos como a base para representações de mapas de geo-objetos e mapas temáticos. Esta seção define em detalhe o conceito de mapa vetorial, tomando como base [PD95], e critérios para fragmentação de mapas vetoriais. Contém, ainda, breves considerações a cerca de operações sobre mapas vetoriais. As relações topológicas entre pares de objetos usadas nesse texto estão definidas em [EF91] e em [CFO93].

Um *ponto* é um par ordenado  $(x, y) \in \mathbb{R}^2$ .

Uma *linha* é uma função contínua  $l : I \rightarrow \mathbb{R}^2$ , onde  $I = [0, 1]$ .

Uma linha é *fechada* se e somente se  $l(0) = l(1)$ ; caso contrário é *aberta*. Os pontos  $l(0)$  e  $l(1)$  são chamados *nós* de  $l$ .

Uma linha  $l$  é *simples* se e somente se  $l$  define uma função injetora em  $I$ , se  $l$  for aberta, ou se  $l$  define uma função injetora em  $[0, 1)$ , se  $l$  for fechada. Intuitivamente,  $l$  é simples se não cruza a si mesma em algum ponto.

Dada uma linha  $l$  e  $a, b \in [0, 1]$ , o *segmento de linha* de  $l$  entre  $a$  e  $b$ , denotado por  $l_{a,b}$ , é a função  $l$  restrita ao intervalo  $[a, b]$ .

Uma *cadeia* é uma seqüência de linhas  $c = (l_0 \dots l_{k-1})$  tal que, para todo  $i \in [1, k-1]$ , temos  $l_{i-1}(1) = l_i(0)$ . Note que, a partir de  $c$ , podemos definir uma linha  $l_c$ , chamada de linha *induzida* por  $c$ . Uma cadeia  $c$  é *simples*, *aberta* ou *fechada* se e somente se  $l_c$  o for.

Uma *região* é um par  $r = (c_0, F)$ , onde  $c_0$  é uma cadeia fechada simples, chamada *fronteira externa* da região, e  $F$  é um conjunto  $r_1, \dots, r_n$ , possivelmente vazio, de regiões, onde  $n \geq 0$ . As fronteiras externas de  $r_i$  são chamadas de *fronteiras internas* da região  $r$  e são tais que não se interceptam, estão contidas no interior de  $c_0$  e, para todo  $i, j \in [1, n]$  com  $i \neq j$ , temos que  $r_i$  está no exterior de  $r_j$ . Intuitivamente, as fronteiras internas  $r_1, \dots, r_n$  definem buracos na região  $r$ . De fato, a definição recursiva acima possibilita a

existência de buracos dentro de buracos, ou seja, a representação de uma ilha dentro de um lago no interior de um território, por exemplo.

Um *mapa vetorial*, ou simplesmente um *mapa*, é uma tripla  $M = (P, L, R)$ , onde  $P, L$  e  $R$  são respectivamente conjuntos de pontos, linhas e polígonos, os *elementos* do mapa, tais que: (1) dadas duas regiões  $r$  e  $s \in R$ , então  $r$  e  $s$  são disjuntas, ou  $r$  e  $s$  se tocam (em uma linha ou ponto); (2) dadas uma linha  $l \in L$  e uma região  $r \in R$ , então  $l$  e  $r$  são disjuntas, ou  $l$  e  $r$  se tocam em um ponto  $p$ , que deve ser um nó de  $l$  e pertencer à fronteira de  $r$ , ou  $l$  faz parte da fronteira de  $r$ , ou  $l$  é interior a  $r$ , ou  $l$  é coberta por  $r$  (ou seja,  $l$  é interior a  $r$ , mas pelo menos um nó de  $l$  faz parte da fronteira de  $r$ ); (3) dados um ponto  $p \in P$  e uma região  $r \in R$ , então  $p$  e  $r$  são disjuntos, ou  $p$  está na fronteira de  $r$  (ou seja  $p$  é nó de duas linhas consecutivas da fronteira de  $r$ ), ou  $p$  é interior a  $r$ ; (4) dadas duas linhas  $l$  e  $m \in L$ , então  $l$  e  $m$  são disjuntas, ou  $l$  e  $m$  se tocam em um nó comum; (5) dados um ponto  $p \in P$  e uma linha  $l \in L$ , então  $p$  e  $l$  são disjuntos, ou  $p$  é nó de  $l$ .

Um *mapa temático* é uma tripla  $m = (C, M, f)$  onde  $C$  é um conjunto finito de *temas*,  $M = (P, L, R)$  é um mapa vetorial e  $f : P \cup L \cup R \rightarrow C$  é uma função que associa cada elemento de  $M$  a um tema em  $C$ . Um *mapa de geo-objetos* é definido de forma idêntica, exceto que  $C$  é um conjunto finito de geo-objetos. Dado um geo-objeto  $c \in C$ , diz-se que  $f^{-1}(c)$  é a *representação* de  $c$  em  $m$ .

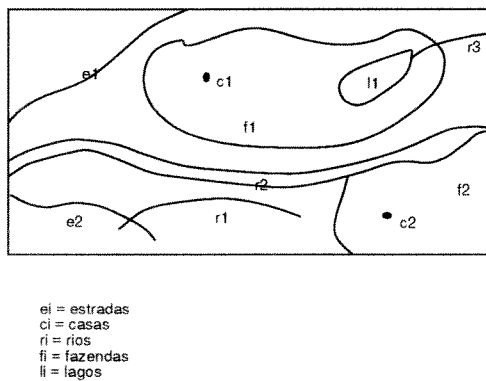


Figura 3: Exemplo de mapas.

A figura 3 exemplifica dois mapas de geo-objetos desenhados em conjunto. O primeiro deles é o mesmo da figura 1 e representa a hidrologia de uma região. O segundo mapa representa a localização espacial de objetos geográficos, onde as regiões  $f1$  e  $f2$  representam duas fazendas, as linhas  $e1$  e  $e2$ , duas estradas e os pontos  $c1$  e  $c2$ , duas casas.

Dado um mapa  $M = (P, L, R)$  e uma região  $u$ , o *fragmento de  $M$  induzido por  $u$*  é o mapa  $M' = (P', L', R')$  definido da seguinte forma:

- $P'$  é o subconjunto dos pontos em  $P$  no interior de  $u$ ;

- $l' \in L'$  se e somente se existe  $l \in L$  tal que
  - $l$  cruza  $u$  e  $l'$  corresponde ao fragmento de  $l$  que está no interior de  $u$ ;  $l'$  é chamado de *elemento fragmentado* de  $M'$ .
  - ou  $l$  está no interior de  $u$  e  $l = l'$ .
- $r' \in R'$  se e somente se existe  $r \in R$  tal que
  - $r$  e  $u$  se interceptam e  $r'$  corresponde ao fragmento de  $r$  que está no interior de  $u$ ;  $r'$  é chamado de *elemento fragmentado* de  $M'$ .
  - ou  $r$  está no interior de  $u$  e  $r = r'$ .

Um *critério de fragmentação de mapas* para uma região  $u$  de  $\mathbb{R}^2$  é um conjunto finito  $F = \{f_1, \dots, f_n\}$  de regiões que cobrem  $u$ . Dado um conjunto  $\mu$  de mapas,  $F$  induz então um segundo conjunto  $\mu'$  consistindo dos fragmentos dos mapas em  $\mu$  induzidos pelas regiões em  $F$ .

Por fim, consideramos como uma operação sobre mapas qualquer função que tenha como argumento ou contra-argumento um conjunto de mapas. Um exemplo importante são as chamadas *consultas por janela* (*window queries*) para mapas, que recebem como entrada um mapa  $M$  e uma região  $u$  (normalmente definida pelas extremidades de um retângulo) e retorna o fragmento de  $M$  induzido por  $u$ , conforme definido acima. Outro exemplo importante são as operações de junção espacial para mapas, que possuem como argumentos dois mapas  $M$  e  $M'$  e geram como resposta um terceiro mapa  $M''$  (ou um valor numérico  $v$ ), onde  $M''$  (ou  $v$ ) é construído utilizando-se elementos dos mapas de entrada que possuem entre si um dado relacionamento topológico ou geométrico (por exemplo interseção, cruzamento, pertinência ou proximidade).

## 4 Tratamento de Mapas Vetoriais

### 4.1 Subsistemas Responsáveis por Mapas Vetoriais

A tabela 1 indica os subsistemas descritos na seção 2.2 que são responsáveis pelo tratamento de mapas e indica o nível de abstração dos objetos manipulados por cada um deles.

O SAF armazena e recupera páginas em memória secundária, sem conhecer a semântica dos dados armazenados, enquanto que o SAV implementa as abstrações básicas necessárias à definição de mapas - pontos, linhas, polígonos, regiões e seus conjuntos. O SAR implementa mapas apenas como estruturas de dados com operações elementares, deixando a cargo do SMR a implementação das operações de mais alto nível. A camada superior da arquitetura, o SM, faz a ponte do nível de representação para o nível conceitual conhecido pelas aplicações.

SM	geo-objetos, geo-campos, representações
SMR	mapas temáticos / mapas de geo-objetos
SAR	mapas temáticos / mapas de geo-objetos
SAV	pontos, linhas, polígonos, regiões, conjuntos
SAF	páginas

Tabela 1: Subsistemas e objetos manipulados

O conceito de mapa é nativo, então, das três primeiras camadas do GEOG. Isto significa que a decisão sobre em qual delas será introduzido o armazenamento distribuído de mapas deve levar em consideração apenas o grau de abstração das operações a serem paralelizadas. No caso da seção 4.2, descreveremos métodos de armazenamento distribuído para mapas que são apropriados a uma implementação distribuída do SAR.

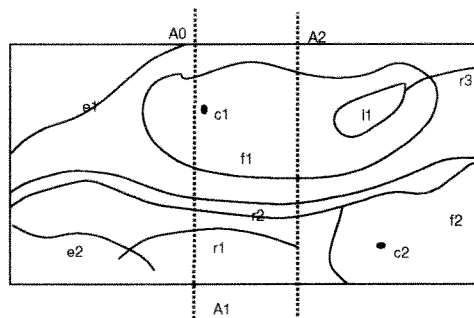


Figura 4: Mapa fragmentado em três partes:  $A_0$ ,  $A_1$  e  $A_2$ .

Independentemente de qual for a decisão, propomos trabalhar com critérios de distribuição para mapas definidos sempre a partir de critérios de fragmentação, ou seja, os processadores armazenarão localmente fragmentos de mapas. Esta forma de distribuição de mapas tem como consequência imediata que uma operação geográfica sobre um mapa será traduzida em operações geográficas semelhantes sobre seus fragmentos, de forma transparente para os usuários. Por exemplo, a figura 4 mostra os mapas da figura 3 particionados em três fragmentos  $A_0$ ,  $A_1$  e  $A_2$ , onde cada um é alocado a um processador distinto. Desta forma, o comprimento da linha  $r_1$  é dado pela soma dos comprimentos dos fragmentos de  $r_1$ . Cada processador local “percebe” que o valor que obtém não é final, pois “sabe” que está trabalhando com um elemento fragmentado. Situação semelhante ocorre no cálculo da área da região  $f_1$ .

## 4.2 Armazenamento Distribuído de Mapas Vetoriais

Esta seção discute duas famílias de métodos de armazenamento distribuído para mapas, criados através de variações e combinações de métodos conhecidos e que oferecem uma

base para a implementação distribuída do subsistema de armazenamento - SAR. A discussão não cobre todos os aspectos da sua implementação, deixando em aberto, por exemplo, o problema de gerenciar quais fragmentos correspondem a que mapas.

Independentemente do método, o critério de distribuição levará em conta apenas a posição geográfica dos objetos. Isto significa que feições da superfície terrestre que estão geograficamente próximas tenderão a ter suas representações armazenadas no mesmo processador. Ou seja, proximidade geográfica induzirá proximidade física de armazenamento, o que representa uma heurística adequada para minimizar o custo das operações geográficas.

Os métodos de armazenamento sugeridos adotam uma estratégia em dois níveis: no primeiro, um determinado método é usado para definir um critério de distribuição dos mapas entre os processadores; no segundo nível, um outro método é adotado para armazenar os fragmentos localmente em cada processador. A estrutura que define o critério de distribuição pode estar replicada em todos os processadores ou armazenada em apenas um ou alguns deles, dependendo da arquitetura global do sistema.

#### 4.2.1 Métodos Híbridos baseados em Grades

Esta seção descreve brevemente uma família de métodos de armazenamento distribuído para mapas que utilizam grades para definir o critério de distribuição. A escolha do método de armazenamento local é deixada livre e deve levar em conta principalmente a otimização da execução das operações locais.

A definição de uma grade  $G$  inclui uma *função de alocação*  $\mathcal{A}[G]$  que mapeia cada célula  $c$  da grade em um conjunto  $\mathcal{A}[G](c)$  de processadores, no caso mais geral em que há replicação de fragmentos de mapas. Dado um mapa  $M$ , o fragmento de  $M$  induzido por  $c$  será então armazenado de forma replicada em todos os processadores no conjunto  $\mathcal{A}[G](c)$  utilizando o método escolhido.

A utilização de grades para representar o critério de distribuição simplifica a localização de um objeto e facilita as operações espaciais entre mapas, já que força o armazenamento nos mesmos processadores dos fragmentos de todos os mapas que sejam induzidos pela mesma célula. Ou seja, a utilização de grades agrupa, em um mesmo processador, todos os fragmentos dos diversos mapas que cobrem a mesma região do espaço. O problema de balanceamento de carga, porém, não é resolvido apenas pelo uso de grades, sendo necessária a adoção, ainda, de uma função de alocação criteriosamente escolhida para espalhamento das células entre os processadores. A implementação das grades deve permitir que mais de uma célula seja alocada para o mesmo processador e que uma grade seja subdividida e as novas células alocadas a mais de um processador. Note que o processo de subdivisão de células pode gerar uma realocação dinâmica de estruturas entre processadores à medida em que o banco de dados cresce.

Pode ser definida, ainda, uma variação deste método em que as grades agem também como catálogo global do banco de dados. Ou seja, as grades guardam quais mapas estão efetivamente associados a quais células.

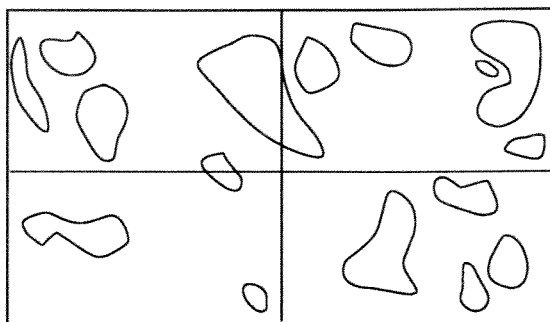


Figura 5: Critério de distribuição da estrutura: uma grade regular define partições.

A figura 5 mostra um mapa composto de diversas regiões (modelando, por exemplo, áreas desmatadas) dividido em quatro fragmentos a partir de uma grade regular. Cada uma das partes é alocada a um processador e armazenada, por exemplo, em árvores R locais.

#### 4.2.2 Métodos Uniformes baseados em Árvores Balanceadas

Esta seção delinea uma família de métodos de armazenamento distribuído em que, tanto o critério de distribuição, quanto o armazenamento local de fragmentos baseiam-se em árvores R. A idéia, grosseiramente falando, consiste em cortar as árvores R (utilizadas para armazenamento) em um dado nível e distribuir as diversas sub-árvores assim obtidas entre os processadores. Um exemplo esquemático da estrutura é mostrado na figura 6.

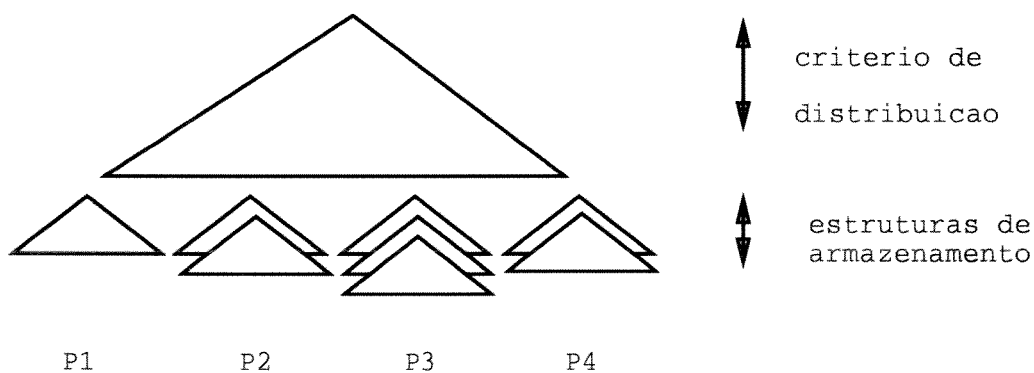


Figura 6: Esquema da estrutura em árvore. O critério de distribuição é dado por uma árvore R e os fragmentos dos mapas são armazenados em árvores R locais.

Brevemente, um método de armazenamento distribuído desta família opera da seguinte forma. O critério de distribuição é definido por uma árvore R (ou uma de suas variantes), chamada *árvore de distribuição* e denotada por  $D$ . Cada folha desta árvore

conterá um par da forma  $(P, B)$ , onde  $P$  é um conjunto de processadores, contendo réplicas dos mesmos fragmentos e  $B$  é o retângulo envolvente destes fragmentos. O total de pares armazenados nas folhas de  $D$  é limitado, levando-se em conta o número de processadores e a função de alocação, parte integrante da definição do método. A árvore é construída, inicialmente, com a inserção direta de pares referentes aos fragmentos dos mapas. Quando o número máximo de pares nas folhas é atingido, dizemos que  $D$  está *cheia*. A partir daí, a inserção de novos mapas induzirá rearranjo dos pares já armazenados, conforme descrito abaixo.

Ao se inserir um novo mapa no banco de dados, digamos  $M$ , uma árvore  $R$  (do tipo escolhido para armazenamento de fragmentos) é gerada para armazená-la  $M$  (possivelmente de forma centralizada em algum processador onde  $M$  foi criado ou editado). A árvore resultante é cortada no *nível de corte* especificado pelo método. As sub-árvores podadas induzem, então, fragmentos de  $M$  (os retângulos envolventes destes fragmentos correspondem aos retângulos das raízes das sub-árvores). Note que a escolha do nível de corte é bastante importante, pois ele determina o processo de fragmentação dos mapas.

Para cada fragmento  $N$  do mapa  $M$ , a árvore de distribuição  $D$  é alterada da seguinte forma. Se  $D$  não estiver cheia, um novo par  $(Q, C)$ , onde  $Q$  é um conjunto de processadores dado pela função de alocação e  $C$  é o retângulo envolvente de  $N$ , é inserido diretamente como folha de  $D$ . Réplicas do fragmento  $N$  são, então, armazenadas em todos os processadores em  $Q$ . Caso a árvore  $D$  esteja cheia, ela é percorrida para determinar todos os pares, contidos nas folhas, cujo retângulo envolvente  $B$  intercepta  $C$ , o retângulo envolvente de  $N$ . O par  $(P, B)$  cujos processadores em  $P$  estiverem menos carregados é escolhido, privilegiando um critério de balanceamento de carga do sistema. O fragmento  $N$  é armazenado nos processadores em  $P$  e o retângulo envolvente  $B$  é ajustado para cobrir o retângulo envolvente de  $N$ . Caso todos os processadores associados a todos estes pares estiverem lotados, um par  $(P, B)$  é escolhido (segundo heurísticas de agrupamento espacial) e parte dos fragmentos armazenados nos processadores em  $P$  são realocados para outros processadores (uma operação - semelhante à de divisão de nós de uma R-tree convencional - cuja descrição detalhada está fora do escopo deste trabalho). No caso de não existirem pares  $(P, B)$  em que  $B$  e  $C$  se interceptem, novamente uma heurística de agrupamento espacial é usada para escolher a folha de  $D$  em que o novo par  $(Q, C)$  será inserido. Essa inserção pode gerar, recursivamente, um processo de rearranjo da estrutura com a propagação de divisões e reagrupamentos de nós pela estrutura.

A realocação de fragmentos não inviabiliza o uso destes métodos, já que um banco de dados geográfico tende a ser estático e a realocação pode ser feita concorrentemente com outras operações, adaptando-se, por exemplo, o procedimento de rearranjo proposto em [ELS95]. Além disto, se assumirmos que os processadores formam um *cluster*, a transferência de dados entre eles será rápida.

Por fim, observamos que o acesso aos dados tende a ser eficiente, pois depende do acesso a duas árvores  $R$ , uma para localização dos processadores que são candidatos a conter os dados, e outra local, implementada da forma convencional. Operações entre conjuntos são facilitadas se a política de manutenção dos nós da árvore de distribuição

induzir agrupamento espacial dos objetos. Um novo problema, porém, deve ser equacionado: como árvores R permitem a existência de nós cujos retângulos envolventes não são disjuntos, cuidado especial deve ser tomado no processamento de fragmentos em áreas geográficas cobertas por mais de um nó da estrutura.

Novamente, uma variação deste método pode ser definida onde cada folha da árvore de distribuição indica também quais mapas possuem fragmentos armazenados nos processadores por ela representados. Ou seja, a estrutura  $D$  passa a funcionar também como catálogo global de mapas.

## 5 Considerações Finais

Partindo de um modelo e uma arquitetura genéricos, este trabalho enfatizou o tratamento de mapas vetoriais, principalmente a questão de armazenamento distribuído destes objetos, como forma de paralelizar a execução de operadores espaciais e minimizar o custo de certas operações geográficas.

As contribuições principais centraram-se em duas famílias de métodos de armazenamento distribuído para mapas, que oferecem uma base para a implementação distribuída do subsistema de armazenamento. A discussão apresentada foi bastante resumida, encontrando-se uma versão completa em [Toc95].

### Agradecimentos

Os resultados reportados nesse trabalho foram obtidos no âmbito do projeto GEOTECH, executado dentro do Programa Temático em Ciência da Computação do CNPq / MCT.

Os autores agradecem a Andréa S. Hemerly, Maurício R. Mediano, Marcelo P.C. Duarte e Paulo M. Souza pelas valiosas sugestões que contribuíram para a execução do trabalho. A primeira autora também agradece a Capes pelo financiamento parcial do projeto através de sua bolsa de estudos.

### Referências

- [BKSS90] Nobert Beckmann, Hans P. Kriegel, Ralf Schneider, and Benhard Serger. **The R\*-tree: an Efficient and Robust Access Method for Points and Rectangles.** In *Proceedings of the 1990 ACM Sigmod International Conference on Management of Data*, 1990. Vol.19, No.2.
- [BKSS94] Thomas Brinkhoff, Hans P. Kriegel, Ralf Schneider, and Benhard Serger. **Multi-Step Processing of Spatial Joins.** In *Proceedings of the 1994 ACM Sigmod International Conference on Management of Data*, May 1994.

- [CFO93] Eliseo Clementini, Paolino Felice, and Peter Ooesterom. **A Small Set of Formal Topological Relationships Suitable for End-User Interfaces.** In *Proceedings of Third International Symposium of Advances in Spatial Databases*, June 1993.
- [CFS<sup>+</sup>94] Gilberto Camara, Ubirajara Freitas, Ricardo C.M. Souza, Marco A. Casanova, and Andréa S. Hemerly. **Object-Oriented Data Modelling and Interface Design for GIS.** In *International Journal of Geographic Information System*, 1994.
- [Cox91] Frederico Sidney Cox. **Análise de Métodos de Acesso a Dados Espaciais Aplicados a Sistemas Gerenciadores de Banco de Dados.** Master's thesis, UNICAMP, December 1991.
- [DG92] David DeWitt and Jim Gray. **Parallel Database Systems: The Future of High Performance Database Systems.** *Communications of the ACM*, 35(6), June 1992.
- [EF91] Max Egenhofer and Robert Franzosa. **Point-set Topological Spatial Relations.** *International Journal of GIS*, 5(2), 1991.
- [ELS95] Georgios Evangelidis, David Lomet, and Betty Salzberg. **The hB<sup>II</sup>-tree: A Modified hB-tree Supporting Concurrency, Recovery and Node Consolidation,** 1995. Accepted to VLDB'95 Conference.
- [FK93] Christos Faloutsos and Ibrahim Kamel. **High Performance R-tree.** Technical report, University of Maryland, 1993.
- [Goo91] M. Goodchild. **Integrating GIS and Environmental Modeling at Global Scales.** In *Proc GIS/LIS 91*, volume 1, pages 117–127, 1991.
- [Goo92] Michael F. Goodrich. **Geographical Data Modeling.** *Computers & Geosciences*, 18(4), 1992.
- [Her92] John Herring. **Tigris: a Data Model for an Object-Oriented GIS.** *Computer & Geosciences*, 18(4), 1992.
- [HMS<sup>+</sup>94] Andréa S. Hemerly, Maurício Mediano, Paulo Souza, Claudia A. Tocantins, and Marcelo Duarte. **Uma Arquitetura para um Gerenciador de Objetos Geográficos.** Documento Interno - Centro Científico - Rio, 1994.
- [HS92] Erik G. Hoel and Hanan Samet. **A Qualitative Comparison Study of Data Structures for Large Line Segments.** In *Proceedings of ACM SIGMOD Conference*, 1992.
- [HS94] Erik G. Hoel and Hanan Samet. **Performance of Data-Parallel Spatial Operations.** In *Proceedings of the 20th VLDB Conference*, Santiago, Chile, 1994.

- [MCD94] Maurício Mediano, Marco A. Casanova, and Marcelo Dreux. **V-Trees - A Storage Method for Long Vector Data**. In *Proceedings on 20th VLDB Conference*, September 1994.
- [Med95] Maurício Riguette Mediano. **V-trees: Um Método de Armazenamento para Objetos Vetoriais Longos**. Master's thesis, Pontifícia Universidade Católica do Rio de Janeiro, January 1995.
- [PD95] Enrico Puppo and Giuliana Dettori. **Towards a Formal Model for Multiresolution Spatial Maps**, 1995. Accepted to the Fourth International of Symposium of Large Spatial Databases - SSD'95.
- [RM92] Jonathan Raper and David Maguire. **Design Models and Functionality in GIS**. *Computers & Geosciences*, 18(4), 1992.
- [Sam90] Hanan Samet. **The Design and Analysis of Spatial Data Structures**. Addison-Wesley, 1990.
- [Toc95] Claudia Andrade Tocantins. **Uma Arquitetura Distribuída para a Resolução de Operações Espaciais em um Sistema de Informações Geográficas**. Master's thesis, Pontifícia Universidade Católica do Rio de Janeiro, 1995.
- [Wan93] Fangju Wang. **A Parallel Intersection Algorithm for Vector Polygon Overlay**. *IEEE Computer Graphics & Applications*, 1993.
- [WHM90] M. Worboys, H. Hearnshaw, and D. Maguire. **Object-Oriented Data Modeling for Spatial Databases**. *International Journal of Geographical Information Systems*, 4(4), 1990.