Um *benchmark* voltado à análise de desempenho de sistemas de informações geográficas

Este exemplar corresponde à redação final da tese devidamente corrigida e defendida pelo Sr. Ricardo Rodrigues Ciferri e aprovada pela Comissão Julgadora.

Campinas, 19 de junho de 1995.

Prof. Dr. Geovane Cayres Magalhães

Dissertação apresentada ao Instituto de Matemática, Estatística e Ciência da Computação, UNICAMP, como requisito parcial para a obtenção do título de Mestre em Ciência da Computação.

FICHA CATALOGRAFICA ELABORADA PELA

BIBLIOTECA DO IMECC DA UNICAMP

Ciferri, Ricardo Rodrigues

C487b Um benchmark voltado a analise de desempenho de sistemas de informacoes geograficas / Ricardo Rodrigues Ciferri. -- Campinas, [SP : s.n.], 1995.

Orientador : Geovane Cayres Magalhaes

Dissertacao (mestrado) - Universidade Estadual de Campinas, Instituto de Matematica, Estatistica e Ciencia da Computacao.

1. Banco de dados. I. Magalhaes, Geovane Cayres II. Universidade Estadual de Campinas. Instituto de Matematica, Estatistica e Ciencia da Computacao. III. titulo.

Um *benchmark* voltado à análise de desempenho de sistemas de informações geográficas

Ricardo Rodrigues Ciferri[1]

Departamento de Ciência da Computação
IMECC – UNICAMP

Banca Examinadora:

- Geovane Cayres Magalhães (Orientador)[2]
- Claudia Maria Bauzer Medeiros [3]
- Marco Antônio Casanova[4]
- Cecília M. Rubira (Suplente)[5]

---

[1] O autor é formado em Ciência da Computação pela Universidade Federal de São Carlos.
[2] Pesquisador em Telecomunicações do Centro de Pesquisas e Desenvolvimento da TELEBRÁS. Professor em tempo parcial no Departamento de Ciência da Computação – IMECC – UNICAMP.
[3] Professora do Departamento de Ciência da Computação – IMECC – UNICAMP.
[4] Pesquisador do Centro Científico Rio – IBM Brasil.
[5] Professora do Departamento de Ciência da Computação – IMECC – UNICAMP.

Universidade Estadual de Campinas
Instituto de Matemática, Estatística e Ciência da Computação
Departamento de Ciência da Computação

Um *benchmark* voltado à análise de desempenho de sistemas de informações geográficas

por

Ricardo Rodrigues Ciferri

Orientador: Prof. Dr. Geovane Cayres Magalhães

Campinas
19 de junho de 1995

Dedico

Aos meus pais

Domingos e Telma

À minha noiva

Cristina

Aos meus avós

Sebastião, Pedro, Zélia e Lúcia

Agradecimentos

Ao professor e amigo Geovane pela orientação geral do trabalho.

Aos meus pais pelo apoio e estímulo constantes.

A Cristina por seu amor e companheirismo.

A Ivor e Maria Helena pelo incentivo.

A Ardemírio de Barros Silva (Instituto de Geociências - Unicamp), Claudia Bauzer Medeiros (DCC - Unicamp) e Mateus Batistella (Embrapa - NMA) pelas críticas e sugestões a este trabalho.

A Flávio Sammarco Rosa (Departamento de Geografia – USP) pela cessão do cadastro de usuários e empresas de geoprocessamento.

A todas as instituições que responderam o questionário.

Ao CNPq pela concessão da bolsa de mestrado. Parte deste trabalho foi desenvolvido no contexto do projeto GEOTEC do PROTEM – CNPq.

A FAEP pela concessão do auxílio ponte para finalização da escrita da dissertação.

Ao funcionário Luiz Cláudio Rosa da secretaria da pós-graduação da Unicamp.

A todos aqueles que direta ou indiretamente contribuíram para a realização deste trabalho, em especial aos colegas do DCC – Unicamp.

Sumário

A enorme quantidade e a natureza dos dados armazenados por aplicações que utilizam sistemas de informações geográficas (SIGs) implicam em alterações ou extensões nos métodos de acesso, otimizadores de consulta e linguagens de consulta estabelecidos para sistemas gerenciadores de banco de dados (SGBDs) convencionais. Com isto, diferentes soluções têm sido apresentadas, tornando-se imprescindível a criação de algum mecanismo que possa medir a eficiência destas soluções para auxiliar o direcionamento de futuros trabalhos de pesquisas. Para tal propósito é utilizada, nesta dissertação, a técnica experimental de *benchmark*.

Esta dissertação propõe a carga de trabalho e caracteriza os dados de um *benchmark* voltado à análise de desempenho de SIGs. A carga de trabalho do *benchmark* é composta por um conjunto de transações primitivas, especificadas em alto nível, que podem ser utilizadas para a formação de transações mais complexas. Estas transações primitivas são predominantemente orientadas aos dados espaciais, sendo, a priori, independentes do formato de dados utilizado (*raster* ou vetorial). A caracterização dos dados do *benchmark* foi efetuada em termos dos tipos de dados necessários para a representação de aplicações georeferenciadas, e adicionalmente procedimentos para se realizar a geração de dados sintéticos. Finalmente, uma aplicação alvo utilizando dados sintéticos foi definida com a finalidade de validar o *benchmark* proposto.

Abstract

Geographical Information Systems (GIS) deal with data that are special in nature and size. Thus, the technologies developed for conventional data base systems such as access methods, query optimizers and languages, have to be modified in order to satisfy the needs of a GIS. These modifications, embedded in several GIS, or being proposed by research projects, need to be evaluated. This thesis proposes mechanisms for evaluating GIS based on benchmarks.

The benchmark is composed of a workload to be submitted to the GIS being analysed and data characterizing the information. The workload is made of a set of primitive transactions that can be combined in order to derive transactions of any degree of complexity. These primitive transactions are oriented to spatial data but not dependent on the way they are represented (vector or raster). The benchmark data base characterization was defined in terms of the types of data required by applications that use georeferencing, and by the need to generate complex and controlled artificial data. The proposed technique and methods were used to show how to create the transactions and the data for a given application.

Conteúdo

Lista de Figuras

Lista de Tabelas

Lista de Equações

Capítulo 1

Motivação e estruturação da dissertação

Esta dissertação tem por objetivo propor um conjunto de transações primitivas que representem a carga de trabalho de um *benchmark* voltado à análise de desempenho de sistemas de informações geográficas, além de caracterizar aspectos relacionados aos dados.

Um sistema de informação geográfica (SIG) é caracterizado como um *software* composto por vários subsistemas integrados, os quais são voltados à geração de mapas e à extração de informações sobre os objetos geográficos representados nestes mapas, com o auxílio de um sistema gerenciador de banco de dados (SGBD) não convencional e de um conjunto de funções analíticas [PM90, Tim94]. O principal componente de um SIG é o SGBD não convencional. Este SGBD é responsável por permitir o uso conjunto de uma enorme quantidade de dados espaciais e convencionais, através de estruturas de armazenamento de dados, linguagens e otimizadores de consultas específicos [Ooi90, Cox91, Güt94, MCD94].

Devido à natureza dos dados espaciais, uma linguagem de consulta para SIGs é mais complexa do que as linguagens de consultas dos SGBDs convencionais [Sou+93]. Além das facilidades encontradas nas linguagens convencionais, uma linguagem de consulta para SIGs também deve se preocupar com relacionamentos geométricos e topológicos. A manipulação de dados espaciais também implica em uma extensão nos níveis mais internos de um SGBD convencional, tais como otimizador de consultas, métodos de acesso e armazenamento físico dos dados. Atualmente, graças ao crescente desenvolvimento de SIGs, diferentes metodologias têm sido propostas para solucionar os problemas acima citados. Desta forma, torna-se imprescindível a criação de um mecanismo que compare a eficiência destas soluções, visando auxiliar o direcionamento de futuros trabalhos de pesquisa. Para tal propósito é utilizada, em geral, a técnica experimental de *benchmark*.

Genericamente, a técnica de *benchmark* quando aplicada em SGBDs consiste na execução de um conjunto conhecido de transações, ou carga de trabalho como é

comumente denominada, sobre um banco de dados também conhecido [Fer78, Per90, Gra91]. Tanto a carga de trabalho quanto o banco de dados podem ser reais ou sintéticos (artificiais). Os resultados gerados por esta técnica são altamente confiáveis, uma vez que o próprio sistema sendo analisado é utilizado para a obtenção dos resultados de desempenho.

Os *benchmarks* padrões disponíveis atualmente são inapropriados para analisar o desempenho de SIGs. A maioria destes *benchmarks* ou são voltados especificamente à análise de desempenho de sistemas convencionais (*TP1* [Ano+85], *TPC-A* [Tra89a, Tra89b], *TPC-B* [CB92], *TPC-C* [KSR91, Tra92], *Wisconsin* [Dew85, BT88, Gra91], *City Benchmark* [RY94]) ou são voltados à análise de desempenho de sistemas específicos fora do escopo de geoprocessamento (*HiperModel Benchmark* [And+90], *Benchmarking Simple Database Operations* [RKC87], *Object Operations Benchmark – OO1* [CS92], *OO7 Benchmark* [CDN93]). Já os *benchmarks* (protótipos) voltados à análise de desempenho de sistemas georeferenciados [GR87, Bou88, Mcl90, Sto+93] são restritos, uma vez que suas respectivas cargas de trabalho consideram apenas uma quantidade reduzida de transações. Em adição, para estes *benchmarks* os dados utilizados são exclusivamente reais, não havendo menção a respeito da geração de dados sintéticos.

Com base nestas considerações, esta dissertação propõe a carga de trabalho e caracteriza os dados de um *benchmark* voltado à análise de desempenho de SIGs. Visando a criação de um *benchmark* representativo, foi elaborado um questionário que serviu de base para a determinação das transações e do conjunto de dados utilizado por aplicações georeferenciadas reais. Este questionário foi de propósito geral e englobou aspectos relacionados ao *hardware*, *software* e *liveware*, servindo não somente para o levantamento das características espaciais de SIGs, mas também para caracterizar o segmento de geoprocessamento nas principais instituições brasileiras. A criação do questionário foi efetuada em parceria com o Instituto de Geociências da Unicamp e o CPqD Telebrás.

A carga de trabalho do *benchmark* é composta por um conjunto de transações primitivas, especificadas em alto nível, que podem ser utilizadas para a formação de transações mais complexas. Estas transações primitivas propostas são predominantemente orientadas aos dados espaciais, sendo, a priori, independentes do formato de dados utilizado (*raster* ou vetorial). No entanto, algumas transações definidas são voltadas especificamente para o formato *raster*, ou ainda utilizam ambos formatos conjuntamente. A carga de trabalho proposta é baseada em transações realizadas durante a fase de execução do sistema. Em outras palavras, não são consideradas, por exemplo, operações realizadas durante a fase de captura de dados.

A caracterização dos dados do *benchmark* foi efetuada em termos dos tipos de dados necessários para a representação de aplicações georeferenciadas, e adicionalmente procedimentos para se realizar a geração de dados sintéticos. Esta caracterização é independente da forma de organização dos dados (estrutura de arquivos, quantidade de dados armazenados, entre outros), de modo a permitir a execução da carga de trabalho do *benchmark* segundo aplicações georeferenciadas específicas.

Finalmente, para permitir a comparação de desempenho entre SIGs distintos, foi definida uma aplicação alvo na qual a carga de trabalho do *benchmark* proposto pode ser executada. Esta aplicação é composta de dados sintéticos, sendo voltada para representar uma aplicação de grande porte. Assim, a comparação de desempenho entre SIGs distintos é possível, uma vez que foram definidos com rigor tanto aspectos relacionados às transações ("*bufferização*", seletividade e arquivos sobre os quais cada transação irá atuar) quanto aspectos relacionados aos dados (estrutura do banco de dados, tipos de dados, quantidade, controle de seletividade e distribuição).

Além deste capítulo introdutório, a dissertação é composta por mais seis capítulos.

O capítulo 2 caracteriza os principais conceitos envolvendo sistemas de informações geográficas, desde sua definição até a descrição das principais aplicações atualmente existentes.

Alguns conceitos relacionados à análise de desempenho são descritos no capítulo 3. Este capítulo discute, em especial, a técnica experimental de *benchmark*, apresentando as vantagens de sua utilização e descrevendo características sobre a carga de trabalho, dados, medidas e formas de comparação de resultados de desempenho.

O capítulo 4 descreve inicialmente os principais *benchmarks* voltados à análise de desempenho de SIGs. A seguir, são descritos os métodos utilizados para a identificação de características de aplicações georeferenciadas, com ênfase no questionário desenvolvido para tal propósito. Por fim, é realizada uma análise das respostas dos questionários distribuídos.

O conjunto de transações primitivas que representam a carga de trabalho do *benchmark* e aspectos relacionados à composição dos dados são apresentados no capítulo 5. Visando verificar a abrangência e o potencial de portabilidade da carga de trabalho do *benchmark* proposto, também são apresentadas neste capítulo algumas consultas comumente encontradas em aplicações georeferenciadas a partir das transações primitivas.

O capítulo 6 descreve uma aplicação alvo composta de dados sintéticos, na qual a carga de trabalho do *benchmark* pode ser executada para comparação de desempenho de SIGs.

A dissertação é concluída no capítulo 7 com as contribuições esperadas e as extensões futuras.

Capítulo 2

Sistemas de informações geográficas

2.1 Introdução

Em geral, qualquer indivíduo está acostumado a interagir com características espaciais em sua rotina diária. Isto ocorre porque todos nós vivemos em um mundo caracterizado por ser de natureza basicamente espacial. Nós moramos em um local, trabalhamos em outro, interagimos com outras pessoas, estabelecimentos comerciais e instituições, todos localizados em um determinado lugar. Decisões envolvendo distância, direção, adjacência, localização relativa, e outros conceitos espaciais complexos são manipulados por nós regularmente, porém de maneira intuitiva. Para nos ajudar a tomar tais decisões tem-se desenvolvido, através de muitos séculos, um dispositivo para armazenamento de informações espaciais. Este dispositivo é conhecido como mapa [1].

Um mapa pode ser definido como sendo uma representação em superfície plana e em escala menor, de um terreno, país, território, etc [Fer86]. Para [SE90] um mapa é apenas um tipo particular de sistema de informação, o qual possui como principal funcionalidade o fato de ser uma ferramenta básica de interpretação espacial.

Mapas são constituídos, em sua totalidade, de objetos geográficos. Um objeto geográfico representa uma entidade estática do mundo real, e portanto possui uma localização fixa em relação à superfície terrestre, descrita por uma geometria. Os objetos geográficos representados em um mapa variam de acordo com a granularidade deste, a qual pode ser classificada em regional, nacional e global, dependendo do nível de abstração requerida. Por exemplo, um mapa regional descrevendo um conjunto restrito de ruas da região central de uma cidade certamente conterá objetos como ruas, escolas, hospitais, praças, bancos, padarias, cinemas e lojas (figura 2.1).

[1] o termo mapa se refere a mapa analógico, salvo aviso explícito no texto.

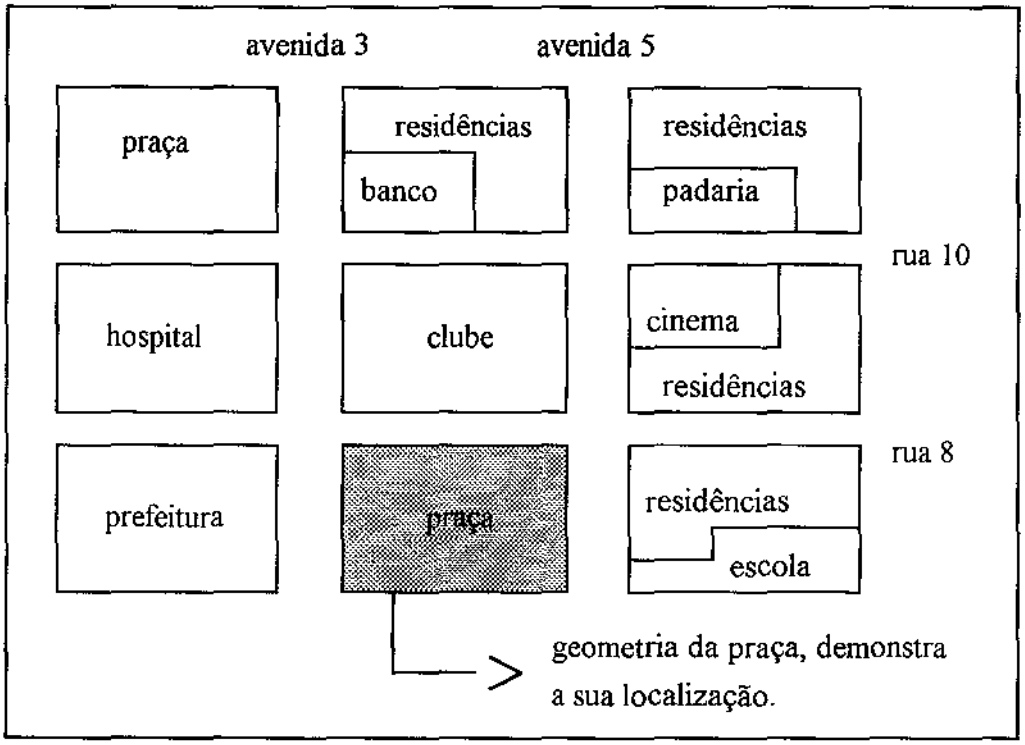


Figura 2.1 Mapa de uma porção específica de um bairro.

Já em um mapa voltado para a representação de um estado da federação (figura 2.2), os objetos geográficos serão cidades, rodovias, ferrovias, rios, lagos, regiões, entre outros.

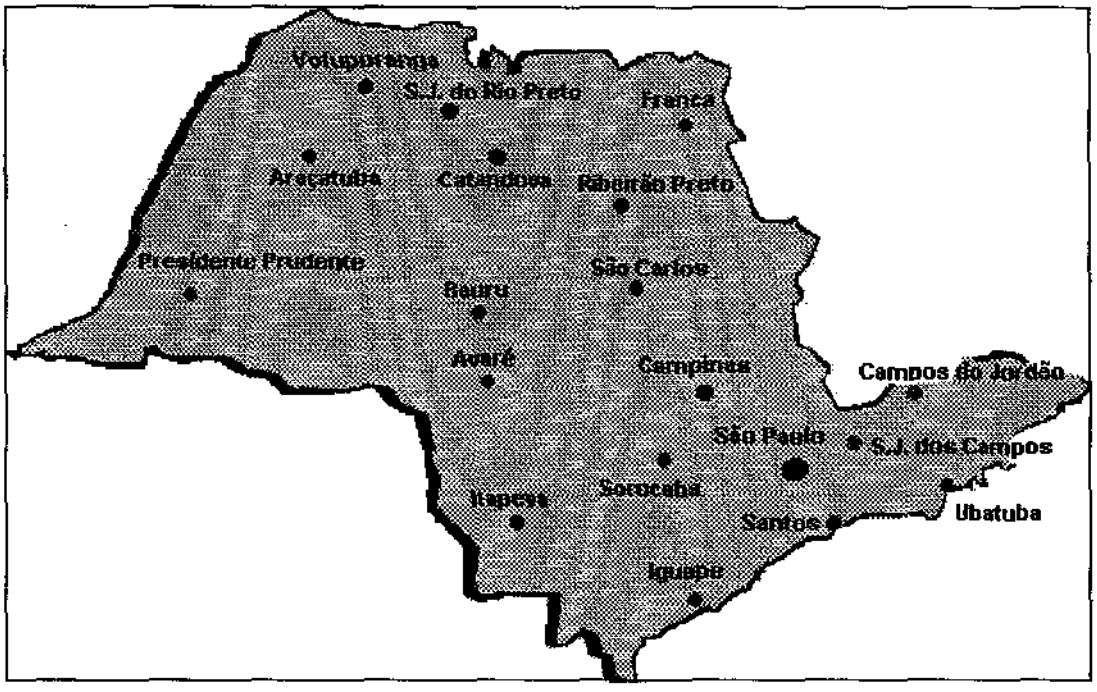


Figura 2.2 Mapa do estado de São Paulo com algumas cidades.

A manipulação manual de mapas, entretanto, gera alguns problemas. Estes problemas são decorrentes principalmente da demora de execução de tarefas manuais e da alta taxa de erros decorrentes da execução destas tarefas.

O primeiro problema é chamado de competição por localização. Um mapa geralmente agrupa somente uma classe particular de objetos geográficos, tais como estradas, rede de esgoto, florestas, entre outros. Quando é necessário juntar mais de uma classe de objetos geográficos em um mesmo mapa, pode ocorrer uma indefinição na representação, caso exista para uma mesma localização vários objetos geográficos. Para contornar este problema foram desenvolvidas complexas técnicas de simbolização cartográfica baseadas em convenções de cores e estilos.

A junção de mapas também é complicada, desde que estes mapas podem estar em escalas diferentes e portanto devem ser transformados para uma escala padrão. Uma escala é definida como a razão entre a distância representada no mapa e o seu verdadeiro tamanho na terra. Para representar uma escala utiliza-se um par de números x : y, na mesma unidade de medida. Desta forma, uma escala 1 : 25.000 indica que 1 unidade de medida no mapa corresponde a 25.000 unidades da mesma medida na terra.

Um terceiro problema está relacionado com a recuperação de informações em mapas. A informação armazenada em mapas é freqüentemente de importância crítica, mas experiências demostraram que enquanto recuperar pequenas quantidades de informações é um processo fácil, a recuperação de grandes quantidades de informações ou a determinação de relacionamentos existentes entre os vários objetos geográficos é um processo muito vagaroso.

A atualização de mapas consiste em um quarto problema. A falta de mecanismos que garantam a propagação de atualizações de objetos geográficos pode gerar inconsistência nas informações contidas nos diversos mapas de uma base cartográfica. Por exemplo, um rio pode estar representado em um mapa contendo a rede hidrográfica estadual, como pode estar representado em um mapa territorial. A modificação de percurso deste rio pode ser alterada no mapa hidrográfico, mas nenhum mecanismo garante que a mudança seja feita no mapa territorial ou em qualquer outro mapa que contenha este rio como um objeto geográfico.

Os problemas citados anteriormente tratam da visualização, mudança de escala, consulta, inserção, remoção e modificação de objetos geográficos em mapas. Sistemas de informações geográficas visam à resolução destes problemas, de maneira fácil, rápida, segura e automatizada.

2.2 Caracterização de SIG

Devido à rápida e recente evolução de SIGs, não existe ainda uma definição padrão aceita por toda comunidade científica que caracterize estes sistemas. A seguir serão apresentadas algumas destas definições. Classificações baseadas em funcionalidade, custo, plataforma, área de aplicação e modelo de dados são as mais comuns. Algumas classificações tentam distinguir um SIG em termos de *hardware* e *software*, enquanto outras classificações tentam caracterizar um SIG como sendo um tipo particular de sistema de informação.

2.2.1 Abordagem *toolbox*

Esta classificação caracteriza um SIG como um sistema que incorpora um sofisticado conjunto de algoritmos, baseado em computador, para a manipulação de objetos espaciais [Ope91]. Um objeto espacial representa uma entidade do mundo real que possui uma geometria associada. Um carro e uma casa são exemplos de objetos espaciais. Seguidores desta classificação asseguram que um SIG não define um campo por si próprio e pode ser considerado como a combinação de processamento de informações com técnicas de análise espacial encontradas em muitos outros ramos da ciência.

Críticas a esta classificação são encontradas em [Car89, Cow90]. O principal ponto criticado está relacionado com o termo objeto espacial. O conceito de objeto espacial generaliza o conceito de objeto geográfico, e assim pode englobar vários tipos de objetos, como exemplo uma molécula de DNA (*deoxyribonucleic acid*). Em um SIG todos os objetos espaciais devem possuir uma localização fixa em relação à superfície terrestre, ou seja, devem ser georeferenciados. Desta forma, um sistema baseado em computador que lide com moléculas de DNA para o reconhecimento de novos padrões de vírus não poderia ser caracterizado como um SIG, apesar de possivelmente oferecer um conjunto de algoritmos para a manipulação do objeto espacial DNA.

2.2.2 Abordagem de banco de dados

Segundo esta classificação, um SIG é formado por um SGBD não convencional[2] (figura 6.1), o qual é usado para armazenar dados relativos a objetos geográficos, e por um conjunto de processos especializado no tratamento de dados espaciais [Smi+89, Ooi90, Sou+93, Sil94].

Três tipos de dados podem ser associados a objetos geográficos [MM93, Tim94]: dados espaciais, dados convencionais e dados gráficos, os quais são conjuntamente denominados de dados geográficos.

[2] também chamado de SGBD estendido ou SGBD não convencional.

- **dados espaciais:** um dado espacial consiste em um conjunto de coordenadas que descreve a geometria de um objeto geográfico. Estas coordenadas são referentes a um sistema de projeção cartográfica, tal como latitude e longitude. Este tipo de dado é usado muitas vezes como índice espacial para localizar geograficamente objetos geográficos no banco de dados. Isto não é equivalente, por exemplo, a um endereço, o qual é considerado apenas como um dado convencional, uma vez que este descreve a localização discreta de um objeto geográfico, como uma residência, não descrevendo os limites de um objeto geográfico, ou seja, do terreno que a residência ocupa. Nas figuras 2.1 e 2.2, a geometria de uma praça e a localização de uma cidade, respectivamente, são caracterizados como dados espaciais.

  Para representar a geometria de objetos geográficos são usados basicamente os conceitos de ponto, linha e polígono (figura 2.3). Um ponto é a menor unidade possível para representar um objeto geográfico. Um objeto geográfico constituído de apenas um ponto não possui área. Um ponto pode estar associado a um ou mais objetos geográficos (isto é possível quando representa-se diferentes tipos de conjuntos de dados (temas) em um mesmo mapa), sendo que cada um destes objetos geográficos pode ter uma seqüência arbitrária de dados convencionais associados a ele. Uma linha é uma seqüência de pontos conectados retilineamente, onde cada par de pontos conectados corresponde a um segmento de linha. Um polígono é formado por uma linha fechada. Pontos, linhas e polígonos são estruturas 0-dimensionais, 1-dimensionais e 2-dimensionais, respectivamente. Em geral[3], pontos são usados para representar localizações discretas como uma cidade em um mapa; linhas são usadas para representar objetos geográficos lineares, tais como rios, estradas e limites; e polígonos são usados para representar objetos geográficos bidimensionais, tais como a área ocupada por um bairro, cidade ou país.

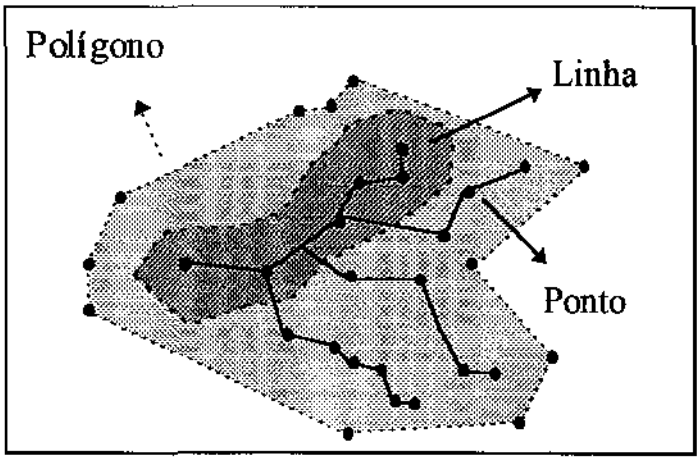


Figura 2.3 Dados espaciais – Conceitos de ponto, linha e polígono.

[3] dependendo da resolução espacial.

- **dados convencionais:** refletem atributos associados a um objeto geográfico, tais como nome de uma cidade, população de um país, tipo de água de um lago, número de leitos de um hospital, proprietário de uma casa, entre outros (figura 2.4).

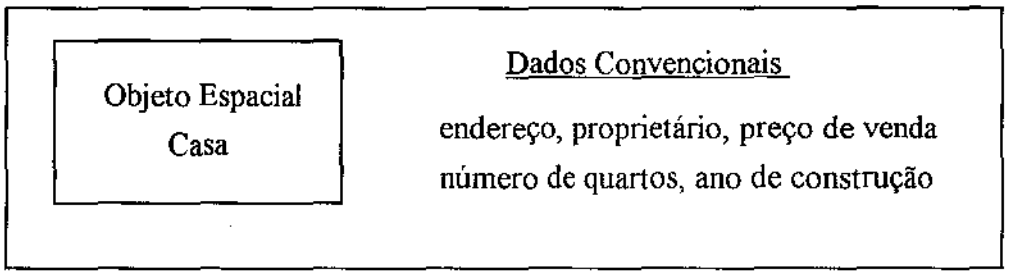


Figura 2.4 Dados convencionais do objeto espacial Casa.

- **dados gráficos:** também chamados dados pictóricos, são formados por *pixels*. Um *pixel* corresponde à menor unidade de exibição de uma imagem. Dados gráficos são usados para representar dados espaciais e convencionais na tela de um computador, de acordo com convenções de cor, estilo, forma e tamanho [Tel91]. Geralmente, um SIG possui tipos de dados gráficos predefinidos para cada um de seus tipos de dado espacial e convencional, sendo também possível a derivação de dados gráficos a partir de dados convencionais e espaciais. Dados gráficos, adicionalmente, podem corresponder a uma fotografia digitalizada ou a uma imagem de satélite. Deve-se diferenciar o conceito de célula (seção 2.3), encontrado no formato varredura, de dados gráficos. Estes últimos, embora possam visualmente representar a geometria de um objeto geográfico, não oferecem meios para a extração de coordenadas geográficas, nem permitem a associação de categorias. O termo *pixel* é comumente encontrado na literatura SIG erroneamente como sinônimo de célula. Dados gráficos são manuseados por funções de processamento de imagens (figura 2.5).

Figura 2.5 Dado gráfico – Fotografia de uma casa.

Já [Bou88] classifica os dados encontrados em SIGs urbanos em quatro tipos, segundo sua origem e utilização: dados cadastrais, dados topográficos, redes e dados temáticos. Dados cadastrais preocupam-se com a localização geográfica, identificação e caracterização de lotes (*land parcels*), tais como tipo, geometria, presença de construção, proprietário, entre outros. Dados topológicos incluem a descrição de rodovias, ruas, avenidas, entre outras, as quais são divididas em "pedaços". Redes correspondem basicamente às redes de gás, eletricidade, telecomunicações, água e esgoto. Dados temáticos são dados relativos a um assunto ou tema específico, tais como uso do solo, poluição e divisão política. A divisão dos dados em temas específicos facilita a visualização de fenômenos, assim como diretamente deriva as transações de análise de ponderação, superposição e reclassificação (seções 5.2 e 5.3), necessárias para análise conjunta de temas distintos. A figura 2.6 descreve uma mesma área geográfica contendo dados temáticos distintos.

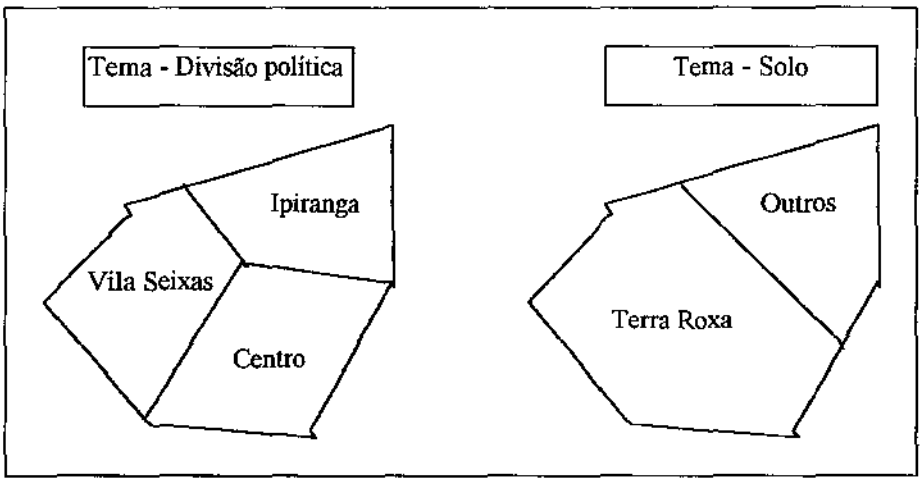


Figura 2.6 Dados temáticos.

O uso apenas de SGBDs convencionais para dar suporte às necessidades de armazenamento de objetos geográficos em um SIG gera vários inconvenientes. A inabilidade das linguagens de consultas convencionais em oferecer operadores espaciais, geométricos e topológicos, para a realização de consultas em SIGs, constitui um grave problema. Em um SGBD convencional não seria possível a realização direta de consultas do tipo: "Quais as cidades com população em idade escolar superior a *40.000* habitantes e que estão localizadas no máximo a *120* km do rio Pardo".

Operadores geométricos retornam um valor escalar, explorando assim, a existência de alguma métrica, como distâncias e direções. Como exemplo de operadores desta classe pode-se citar os operadores de distância (entre dois objetos geográficos), área (de um objeto geográfico) e perímetro (de um objeto geográfico). Operadores topológicos retornam um valor *booleano*, explorando os conceitos de disjunção, interseção, inclusão, etc, entre objetos geográficos. Operadores que retornam um objeto geográfico a partir de dois ou mais objetos geográficos são chamados de operadores espaciais.

Mesmo que a linguagem de consulta de um SGBD convencional seja estendida para suportar estes operadores, a consulta não será efetuada eficientemente, pois os métodos de acesso, otimizador de consultas e tipos de dados deste SGBD são inapropriados para a manipulação de dados espaciais e gráficos, sendo projetados somente para suportar dados convencionais.

Métodos de acesso são caracterizados por uma estrutura de dados e por um conjunto de algoritmos de pesquisa para recuperação de dados. A principal funcionalidade de um método de acesso é fornecer um caminho otimizado aos dados com base em um conjunto definido de predicados sobre os atributos. SIGs devem possuir métodos de acesso que otimizem tanto o acesso aos dados espaciais, chamados de métodos de acesso espaciais, quanto aos dados convencionais, chamados de métodos de acesso convencionais. Um método de acesso espacial deve ser capaz de identificar relacionamentos espaciais existentes entre vários objetos geográficos. Alguns métodos de acesso espaciais encontrados na literatura são: *R-Tree, R-Tree Greene, R^+-Tree, R*-Tree, R*-Tree R, Cell Tree, K-d Tree, Spatial K-d Tree, Grid File, Quadtree, MX-CIF Quadtree e Point Quadtree* [Cox91].

A função de um otimizador de consultas é avaliar, para todos os predicados envolvidos em uma consulta, a melhor estratégia a ser utilizada para satisfazê-la. Como consultas em um SIG podem envolver predicados espaciais e convencionais, o otimizador de consultas deve ser estendido para interpretar predicados espaciais, os quais são inexistentes em uma arquitetura convencional. A necessidade de otimizadores de consultas que tratem de ambos os dados espaciais e convencionais torna-se mais aparente quando a consulta envolve estes dois tipos de dados. Como exemplo, uma consulta do tipo "mostre todas as cidades cujo nome começa com a letra A e que esteja no máximo a 83,71 km de São Paulo" requer do otimizador de consultas a capacidade de determinar qual tipo de predicado deve ser primeiramente pesquisado a fim de restringir o escopo da pesquisa. Para este exemplo, o otimizador deve decidir se restringe a pesquisa efetuando primeiramente a pesquisa com o predicado convencional (nome da cidade) ou efetuando a pesquisa com o predicado espacial (localização + cálculo da distância). A ordem em que os predicados são processados têm significado decisivo para a eficiência da resposta à consulta, uma vez que os predicados atuarão sobre diferentes tamanhos de conjunto de dados.

A representação física dos dados espaciais, os quais são constituídos logicamente por pontos, linhas e polígonos, influenciam diretamente o desempenho de um SIG. O tamanho utilizado pelo tipo de dado que representa o dado espacial pode ser muito grande, tornando a leitura e gravação destes dados ineficiente. SGBDs convencionais não oferecem tipos de dados específicos para dados espaciais, especialmente para polígonos. Desta forma, a representação destes dados deve ser feita utilizando-se apenas os tipos básicos convencionais disponíveis, os quais são inapropriados para esta tarefa. Um SIG deve, portanto, fornecer tipos especiais para a representação de dados espaciais.

Por último, deve-se incluir um modelo de dados que represente tanto os dados espaciais quanto os dados convencionais, além de relacionamentos entre os vários objetos geográficos e suporte temporal [PMS93].

Segundo esta classificação, um SIG pode ser visto como uma camada sobre um SGBD convencional, tal como o relacional, que oferece rotinas especializadas na manipulação de dados espaciais. O principal objetivo dos seguidores desta classificação é o desenvolvimento de SGBDs eficientes no tratamento da ligação de dados espaciais e convencionais. Outros aspectos tais como captura de dados, visualização de dados e suporte a funções analíticas, apesar de necessárias, não constituem uma preocupação maior.

[Smi+89] define um SIG como um sistema de banco de dados contendo uma grande quantidade de dados indexados espacialmente, e sobre os quais um conjunto de procedimentos opera para responder consultas sobre os objetos geográficos armazenados.

[Sil94] diz que um SIG necessita usar o meio magnético, neste deve existir uma base de dados integrada, os dados precisam estar georeferenciados, e deve haver funções de análise destes dados que variem de algébrica cumulativa (operações aritméticas) até algébricas não-cumulativas (operações booleanas).

[Ooi90, Sou+93] definem um SIG como um sistema de banco de dados que permite a manipulação, armazenamento, recuperação e análise de objetos geográficos, além de mostrar estes objetos na forma de mapas.

2.2.3 Abordagem orientada a processos

Esta classificação baseia-se na premissa que um sistema de informação geográfica constitui-se apenas em um tipo específico de sistema de informação [Aro89, Rip89, SE90, Tim94]. Um sistema de informação pode ser definido como uma cadeia de operações que envolve desde o planejamento e coleta de dados, posterior armazenamento e futuro uso destas informações para gerar outras informações em algum processo de decisão. Desta forma, um SIG é caracterizado como uma coleção de subsistemas integrados que ajudam a converter dados geográficos em futuras informações úteis. O sistema completo deve fornecer rotinas para entrada de dados, armazenamento, recuperação, análise e geração de saídas.

[Car89] identifica alguns problemas presentes em qualquer tipo de sistema de informação, inclusive em um SIG. Dentre estes problemas destaca-se a necessidade de proteger as informações, a necessidade freqüente de melhorar o sistema para mantê-lo em um apropriado nível de tecnologia, o que é muito custoso para SIGs, e a necessidade de formar um grupo de profissionais especializados, que além de conhecer muito bem o sistema de informação, deve educar usuários alertando-os sobre as reais funcionalidades e limitações do sistema de informação implantado.

Esta classificação não requer a obrigatoriedade do uso de computadores em SIGs. Alguns autores [Aro89, SE90] afirmam que um SIG pode ser tanto manual como baseado em computador, sendo que um SIG manual providencia as mesmas informações que um SIG baseado em computador. Já [Mag91] reconhece a existência de SIGs manuais, porém destaca que todos os recentes SIGs são baseados em computador. A obrigatoriedade ou não do uso de computadores em SIGs tem gerado grande polêmica. Isto surge devido à necessidade mínima de desempenho de algumas aplicações georeferenciadas na execução de determinadas funções analíticas. Desta forma, o computador torna-se o único meio para a realização eficiente destas funções [PM90].

[SE90] define um SIG baseado em computador como um sistema de informação que é projetado para trabalhar com dados referenciados por coordenadas espaciais ou geográficas, sendo que este sistema deve embutir um SGBD não convencional para manipular dados espaciais, além de funções analíticas.

A seguir são descritas cada uma das cinco fases que formam um SIG baseado em computador.

- **entrada de dados:** também chamada de coleta de dados, consiste na obtenção de dados geográficos através de alguma técnica de captura, tal como fotografia, sensoriamento remoto, GPS (*Global Positioning System*), digitalização de mapas existentes, documentos arquivados, ou ainda simples observação. Esta fase é considerada crítica, pois além de ser muito custosa, devido à enorme quantidade de dados comumente coletada, deve gerar dados geográficos com precisão. Esta precisão varia de uma aplicação para outra. Quanto maior a precisão necessária, mais custosa é a obtenção dos dados. Mapas existentes devem ser questionados principalmente em relação ao estado de conservação e à data de criação. Mapas mal conservados podem prejudicar o processo de digitalização, gerando dados imprecisos. Já a data de criação pode indicar que uma mapa está obsoleto, uma vez que a região representada por este pode ter sofrido um processo de transformação que a modificou significativamente.

- **pré-processamento:** esta fase é responsável pela conversão dos dados coletados para um formato apropriado utilizado pelo SIG.

- **gerenciamento de dados:** esta fase é caracterizada principalmente por suportar mecanismos para modificação, remoção, consulta e inserção de novos dados, geralmente com o auxílio de um SGBD. Modernos SGBDs escondem do usuário o modo como os dados são organizados fisicamente, permitindo que o usuário se preocupe apenas com a organização lógica destes dados. Além disto, devem ser providos outros mecanismos comumente encontrados em SGBDs convencionais, sendo alguns adaptados para tratarem com dados espaciais. Dentre estes mecanismos pode-se citar: processador de consulta, otimizador de consulta, gerenciamento de

transações, gerenciamento de *buffer*, controle de acesso e segurança, gerenciamento de recuperação, indexação e modelo de dados.

- **análise:** é responsável por gerar novas informações baseadas nas informações armazenadas. Para isto ela deve ser provida de um conjunto de funções analíticas. Por exemplo, de acordo com as informações a respeito da elevação de uma região pode-se derivar a elevação média e verificar se com esta média é possível a construção de prédios com segurança. Desde que nenhum sistema tem condições de fornecer todos os tipos imagináveis de funções analíticas, um SIG deve ser organizado modularmente para permitir um fácil acoplamento de futuras funções analíticas.

- **geração de saída:** é a fase onde as operações realizadas por um SIG são visualizadas. Isto pode ser efetuado tanto via *softcopy*[4] como via *hardcopy*[5]. Aqui são criadas listagens estatísticas, mapas e gráficos de vários tipos. A edição final de mapas também é realizada nesta fase.

2.2.4 Abordagem orientada a aplicação

Esta classificação refina a abordagem orientada a processos classificando um SIG de acordo com o tipo de informação sendo manuseada. Termos como sistemas de informações urbanas, sistemas de informação naturais, sistemas multipropósitos, sistemas cadastrais, entre outros, podem ser vistos como subáreas específicas de SIGs. [Jey88], citado em [Rip89], sugere a necessidade de dois tipos de definição sobre SIG: uma para o público leigo e outra para o público especializado na área. Desta forma, cada um dos termos citados poderia ser considerado como definições voltadas para o público especializado em subáreas de SIGs. Entretanto, apesar deste tipo de definição ajudar a ilustrar o escopo de atuação de SIGs, esta é considerada evasiva, tornando-se imprópria para a caracterização precisa destes sistemas. Um SIG deve ser independente de escala e de assunto.

2.2.5 Classificação baseada no tipo de consultas efetuadas

Um tipo de classificação de SIGs muito interessante é baseado no tipo de consultas efetuadas. Para um sistema ser considerado um SIG ele deve ser capaz de responder um conjunto de consultas. Estas consultas envolvem a análise de dados convencionais e espaciais, além de requerer a determinação de relacionamentos entre os vários objetos geográficos. [Car86], citado em [Cow90], define um SIG como um sistema baseado em computador que é capaz de determinar quais parcelas de terrenos satisfazem seis critérios básicos para a construção de um certo tipo de indústria. Estes critérios são: ter pelo menos 5 acres de área, estar situado em zona comercial, estar à venda, não estar sujeito à

[4] *softcopy* = qualquer tipo de imagem gerada em um monitor de computador.
[5] *hardcopy* = qualquer tipo de impressão de dados em meios duradouros, como o papel.

inundação, ser distante no máximo 1 km de uma estrada asfaltada e possuir uma inclinação média inferior a 10%. É importante notar que se as informações sobre estes critérios, para cada terreno, estiverem armazenados em um SGBD convencional, na forma de dados convencionais, este SGBD poderá responder rapidamente esta consulta. Entretanto, a coleta de dados seria muito custosa, o que implica que o sistema não deve somente consultar os dados convencionais e sim gerar novos dados a partir dos dados espaciais existentes. Dados espaciais em um SIG são mais úteis para um processo de decisão em uma consulta do que para uma simples geração de mapas.

2.2.6 Outras classificações

Um outro tipo de classificação para SIG é apresentado por [DC88], baseado em três elementos básicos: tecnologia, banco de dados e infra-estrutura. Tecnologia engloba tanto aspectos de *hardware* quanto de *software*. Hoje quase qualquer tipo de plataforma de *hardware* pode ser usada para dar suporte a um SIG, desde computadores pessoais até *mainframes* [Fra91, Bat+93]. Entretanto, para facilitar o trabalho de entrada e saída de dados em um SIG, vários periféricos especializados são necessários, tais como *scanner*, mesa digitalizadora, *plotter*, entre outros. Aspectos de *software* envolvem técnicas de programação (procedural, orientada a objetos, etc), tipos de organização e modelagem. O segundo elemento básico, o banco de dados, representa uma importante parte de um SIG e é responsável direto pelo seu desempenho, uma vez que armazena enormes quantidades de dados geográficos. O último elemento básico, a infra-estrutura[6], engloba todas as pessoas responsáveis pelo projeto, implementação, suporte e uso de um SIG.

[Mag91] também segue a classificação proposta por [DC88], porém destacando o aspecto institucional. Para este autor, um SIG é melhor descrito como uma coleção integrada de *hardware/software*, dados e *liveware*, os quais operam em um contexto institucional.

Já [Car89], outro simpatizante do acréscimo do aspecto institucional para definir um SIG, descreve um SIG como sendo uma unidade institucional, refletindo uma estrutura operacional que integra tecnologia com banco de dados, conhecimento e contínuo suporte financeiro; e em particular, lida com problemas espaciais relacionados com parcelas da superfície terrestre, adicionando as dimensões espaciais e temporais ao banco de dados.

[Mul85], citado em [Cow90], diz o seguinte: "SIGs são vistos freqüentemente como operações em alta-escala com alto custo inicial, usualmente financiados por órgãos governamentais, tanto em nível federal, quanto estadual e municipal. O principal propósito destes SIGs é ajudar políticos e burocratas a tomarem decisões em relação ao gerenciamento de recursos naturais e humanos". Esta definição destaca as funcionalidades de instituição e de tomada de decisão de um SIG.

---

[6] também chamado de *liveware* e *peopleware*.

Outros autores, tais como [Cow90] também ressaltam a funcionalidade de um SIG como um sistema de suporte à decisão. Um sistema de informação pode ser dividido em dois tipos básicos: sistemas de processamento transacionais e sistemas de suporte à decisão. De maneira simplista, pode-se caracterizar um sistema de processamento transacional como sendo um sistema voltado para alterações em registros, enquanto um sistema de suporte à decisão é voltado para a geração de novos dados a partir da análise dos dados armazenados. Maiores detalhes são encontrados em [Mag91]. [Cow90] define um SIG como um sistema de suporte à decisão envolvendo a integração de dados referenciados espacialmente em um ambiente de solução de problemas. Entretanto, alguns autores, tal como [Rhi88], possuem sérias dúvidas sobre quão bem SIGs podem ser usados para o propósito de tomada de decisão.

2.2.7 Uma definição final sobre SIGs

Para tentar ser o mais rigoroso possível na definição de um SIG, este trabalho não se baseará exclusivamente em nenhuma das definições apresentadas anteriormente.

A caracterização de um SIG será feita a partir de três componentes básicos, cada um dos quais encontrados em várias definições já apresentadas, porém de forma integrada. O primeiro componente enfoca a produção de mapas, ou seja, é baseado em aspectos cartográficos. Alguns representantes são [Ber87, Tom91]. Já o segundo enfatiza o uso de um SGBD, constituindo uma visão voltada para sistemas de banco de dados. Finalmente, o terceiro componente enfatiza a importância da presença de análise espacial, ou seja, de um conjunto de funções analíticas que manipula objetos espaciais (geográficos).

Estes três componentes serão agrupadas conjuntamente com o conceito de sistema de informação para definir detalhadamente um SIG. Desta forma, um SIG pode ser definido como um sistema com:

- **capacidade cartográfica:** digitalização (conversão de mapas analógicos para a forma digital), visualização gráfica de mapas, manipulação gráfica de mapas (inserção, alteração, remoção, *zoom*, entre outros, de objetos geográficos) e impressão de mapas (através, por exemplo, de um *plotter*), entre outras. Abrange as fases de captura de dados e geração de saídas da abordagem orientada a processos, porém restringindo-se para a produção de mapas com o auxílio de computador.

- **capacidade de gerenciamento de dados:** deve englobar os conceitos descritos na seção 2.2.2. Desta forma, métodos de armazenamento e consulta de objetos geográficos eficientes devem ser suportados de forma transparente por um SGBD não convencional.

- **capacidades analíticas:** o sistema deve ter a habilidade de interpretar os dados espaciais armazenados no SGBD. Algumas funções analíticas requeridas são[7]:

  - **computação de operações escalares:** tais como distância entre dois pontos e cálculo da área que um objeto geográfico ocupa (seção 5.2.9).

  - **superposição de polígonos:** polígonos representando um tema (por exemplo: tipo de solo) são superpostos por polígonos representando um outro tema (por exemplo: limites geográficos regionais) para gerar novos polígonos. Isto facilita a geração de novos mapas, além de satisfazer determinados tipos de consulta espacial (seção 5.2.2).

  - **análise de proximidade:** também conhecida como zona de *buffer*. Esta função analítica consiste em gerar uma área (objeto geográfico bidimensional), na forma de um "corredor", ao redor de um objeto geográfico fonte (seção 5.2.4).

  - **busca espacial:** esta função retorna um conjunto de objetos geográficos que satisfaz um certo relacionamento topológico em relação a um objeto geográfico fonte (seção 5.2.7).

Para este trabalho um SIG é caracterizado como um *software* composto por vários subsistemas integrados, os quais são voltados para a geração de mapas e para a extração de informações sobre os objetos geográficos representados nestes mapas, com o auxílio de um SGBD não convencional e de um conjunto de funções analíticas (figura 2.7). Aspectos institucionais, temporais e relacionados com o *hardware* e o *liveware*, apesar de reconhecidamente importantes para muitas aplicações que usam SIGs, não serão considerados obrigatórios para a sua caracterização.

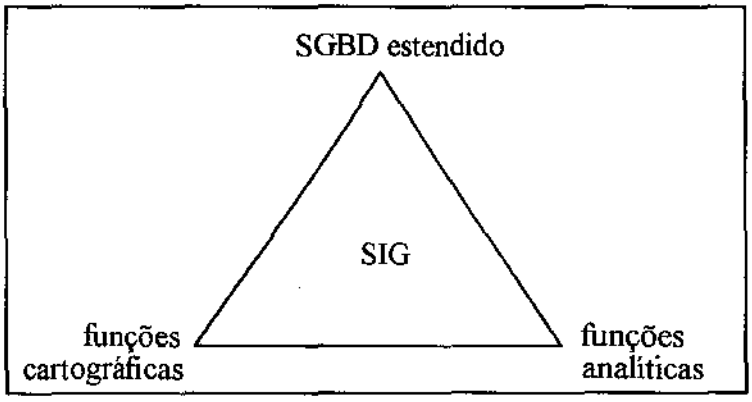


Figura 2.7 Componentes de um SIG.

[7] alguns autores, tal como [Cow90], consideram basicamente busca espacial e superposição como as principais funções analíticas de um SIG.

2.3 Modelos de dados geográficos

Um modelo de dados é uma coleção de ferramentas conceituais para descrição dos dados, relacionamentos entre os dados, semântica dos dados e restrições de consistência. O propósito de um modelo de dados é providenciar uma maneira formal de representar informações, além de meios para manipular tal representação. Para representar as informações contidas no mundo real, os sistemas de banco de dados (SBDs) precisam recorrer a modelos lógicos que consigam, através de construtores específicos, capturar, da melhor maneira possível, a semântica das informações e torná-la visível aos usuários. Assim, um modelo de dados pode ser visto como uma abstração do mundo real que incorpora somente aquelas propriedades que são relevantes para a aplicação em questão, uma concepção humana da realidade [Cod81, EN89, Per90, Peu90, Oli93, KS94, Tim94].

Modelos de dados podem ser estruturados em níveis. Estes níveis vão desde os mais abstratos até aqueles que procuram representar mais proximamente as características físicas do sistema, tais como as estruturas de dados e o *hardware* no qual o sistema está instalado. Uma das vantagens em se utilizar uma representação estruturada em níveis advém da facilidade de se permitir a escolha de determinadas alternativas sem que sejam afetadas as escolhas já realizadas em níveis superiores.

Os primeiros modelos de dados utilizados foram orientados a registros, isto é, modelos que estavam fortemente relacionados à estrutura de registro, na qual os dados são observados como uma seqüência fixa de campos valorados. Os dados nestes modelos assumem uma homogeneidade horizontal e vertical, ou seja, cada tipo de registro tem os mesmos tipos de campos (atributos) e cada campo tem o mesmo tipo de informação em todos os registros. O modelo hierárquico, o modelo de rede e o modelo relacional são exemplos de modelos orientados a registros.

O modelo relacional atualmente é o modelo de dados mais utilizado comercialmente. Neste modelo, o banco de dados é representado como um conjunto de tabelas (relações), onde cada tabela é composta de linhas (tuplas) e de colunas (atributos). O conceito de uma tabela envolve noções bastante simples e intuitivas, que facilitam o usuário a definir o banco de dados. Além disto, existe uma correspondência direta entre o conceito de tabela e o conceito matemático de relação. Além da possibilidade de implementação de tabelas, o modelo relacional possui um mecanismo bastante poderoso para a representação de visões dos usuários. Uma visão, do ponto de vista conceitual, é uma relação virtual, isto é, uma relação que não existe realmente, mas, do ponto de vista do usuário, comporta-se como se existisse, e sobre a qual pode ser definida qualquer operação relacional. Na realidade, as visões não são diretamente suportadas pelo SGBD, elas são apenas definidas em termos de outras relações ou ainda de outras visões já definidas anteriormente. No momento da sua utilização, a definição é processada e a visão se materializa em tempo de execução pelo próprio SGBD. De acordo com sua definição, qualquer operação válida sobre uma relação também poderia ser realizada, teoricamente, sobre uma visão. Na prática, entretanto, os SGBDs não conseguem implementar esta característica genericamente e, em muitos casos,

existem restrições quanto às operações que realizam atualizações e remoções de tuplas em visões.

Apesar de serem utilizados de forma eficiente em uma enorme gama de aplicações, os modelos orientados a registros apresentam certas limitações do ponto de vista semântico. Para tentar superar estas limitações foram propostos modelos que possuem construtores para modelar abstrações mais poderosas dos relacionamentos entre entidades, permitindo uma visão mais natural e realista. Os exemplos mais conhecidos de modelos semânticos são o modelo entidade-relacionamento (ER), o modelo entidade-categoria-relacionamento (ECR), o modelo funcional e o modelo SDM (*Semantic Data Model*).

Um terceiro tipo de modelo que tem merecido atenção especial dos pesquisadores em banco de dados são os modelos orientados a objetos (modelos O.O.). Na verdade, assim como os modelos semânticos, os modelos O.O. pretendem representar de uma forma mais apropriada o mundo real, principalmente no caso de aplicações não convencionais, como CAD/CAM e SIG, que não são suportadas eficientemente pelos modelos convencionais. A grande diferença entre os modelos semânticos e os modelos O.O. é que os primeiros enfatizam aspectos estruturais dos objetos, enquanto os últimos se preocupam principalmente com os aspectos comportamentais destes objetos. Alguns conceitos importantes dentro do contexto O.O. são: objetos complexos, identificadores, encapsulamento, tipos e classes de objetos, hierarquia de tipo e classe e polimorfismo. A descrição destes conceitos pode ser encontrada em [US90].

A modelagem de dados geográfica consiste na formulação de um conjunto adequado de abstrações para a representação da realidade geográfica no banco de dados, e na definição de critérios de manipulação e regras de integridade. Esta modelagem baseia-se em duas diferentes percepções do mundo real, as quais correspondem respectivamente ao modelo baseado em campos e ao modelo baseado em objetos, descritos posteriormente. Segundo [MP94], técnicas convencionais de modelagem de dados, tais como modelagem orientada a registro e modelagem semântica, não são adequadas para o tratamento de dados geográficos. A dificuldade consiste no fato de que a maior parte destes dados são validados em termos de sua localização geográfica, do tempo e da confiabilidade de sua coleta. Para superar estas dificuldades, modelos orientados a objetos que incorporam os conceitos de modelagem dos modelos baseados em campos e/ou baseados em objetos estão sendo propostos na literatura. Alguns exemplos são: Mgeo [Tim94], IFO [SR87], modelo de objetos genérico [Wor92], modelo de Alves [Alv89] e modelo MODGEO$_2$ [Pir95]. Atualmente, entretanto, os modelos baseado em campos e objetos utilizam principalmente SGBDs relacionais para materializar seus conceitos, e com menos freqüência são baseados em implementações específicas em termos de sistema de arquivos. A figura 2.8 visualiza os vários níveis de modelagem de uma aplicação geográfica.

O modelo baseado em campos (*field model*) foi criado com o intuito de representar simplificadamente dados temáticos. Este modelo representa a realidade como uma superfície contínua, denominada camada. Cada camada corresponde apenas a um tipo de tema, tal como solo, vegetação ou declividade. Cada tipo específico de elemento de um

tema, tal como solo argiloso e basáltico no tema solo, é denominado uma categoria. As categorias de um tema são representadas neste modelo através de células. Uma célula corresponde a uma área específica do espaço geográfico. Geralmente, divide-se uma camada em células de mesmo formato, as quais formam uma malha regular. Assim, para cada célula é associada apenas uma categoria. Os principais formatos regulares de células, em ordem crescente de importância, são: hexagonais, triangulares, quadráticas e retangulares (figura 2.9). Células quadráticas são comumente denominadas na literatura SIG como *pixel*. Este termo se confunde com o termo *pixel* encontrado em processamento de imagens e portanto deve ser usado com cuidado (seção 2.2.2). Já o termo *raster* (formato *raster*) é usado para designar células retangulares/quadráticas, sendo em geral também usado como sinônimo de célula, independente de seu formato. Uma discussão das vantagens e desvantagens de cada formato de célula é encontrado em [Peu90].

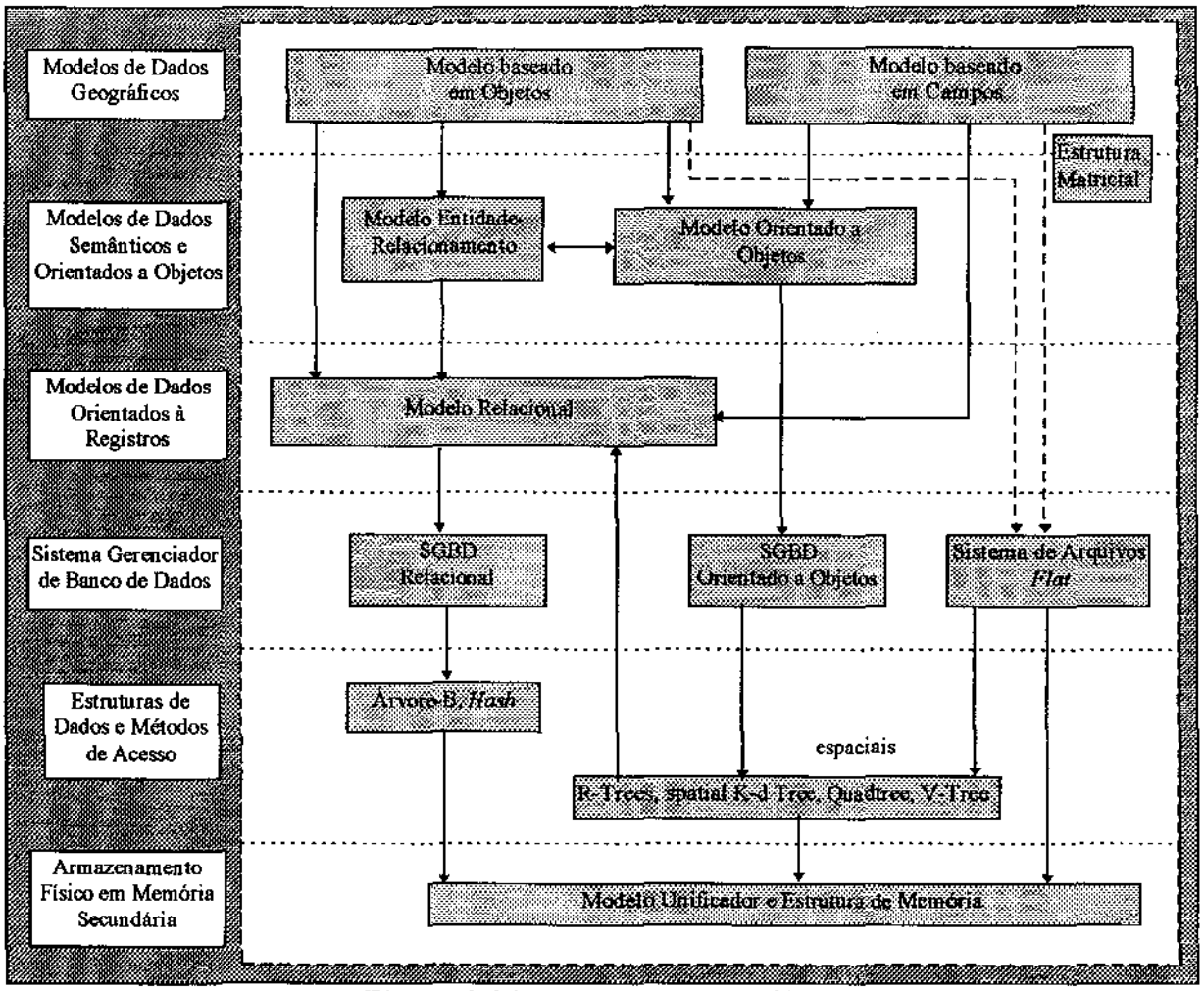


Figura 2.8 Modelagem geográfica.

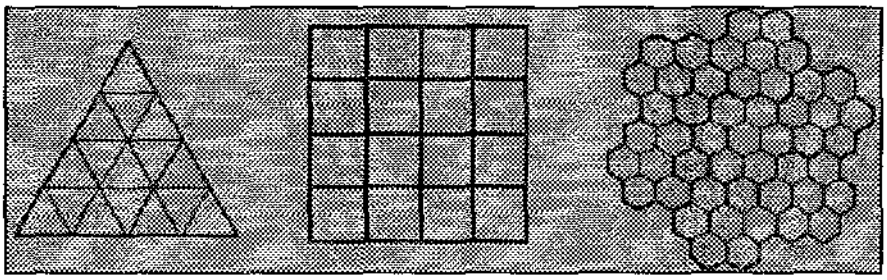


Figura 2.9 Principais formatos regulares de células (triangular, quadrático e hexagonal, da esquerda para a direita).

A estrutura de dados básica utilizada para o armazenamento de dados no modelo baseado em campos, principalmente *raster*, é a estrutura matricial, também conhecida como *array* (figura 2.10). Deste modo, cada posição da matriz corresponde a uma célula, a qual possui um número inteiro associado representando a categoria desta célula. Uma camada, portanto, é armazenada em uma matriz de forma discreta. Dados temáticos representados em matrizes de células são ditos estarem no formato varredura (*tesseral format*). A derivação de coordenadas geográficas neste modelo é efetuada de acordo com a posição das células na matriz, as quais possuem implicitamente associada à sua posição uma localização geográfica. Os principal problema desta representação é a precisão. Qualquer entidade estática do mundo real deve ser representada neste modelo por um conjunto de células. Assim, a representação da geometria desta entidade fica limitada aos limites das células, sendo efetuada de maneira imprecisa, a não ser que a geometria corresponda exatamente à geometria formada por um conjunto de células. Funções analíticas, tais como análise de proximidade, são profundamente afetadas por causa desta limitação. Em geral, dados temáticos não requerem uma alta precisão, sendo geralmente usados para representar mapas nas escalas 1 : 250.000 e 1 : 1.000.000. Quanto maior o tamanho da célula, mais impreciso é o processo de derivação de coordenadas geográficas.

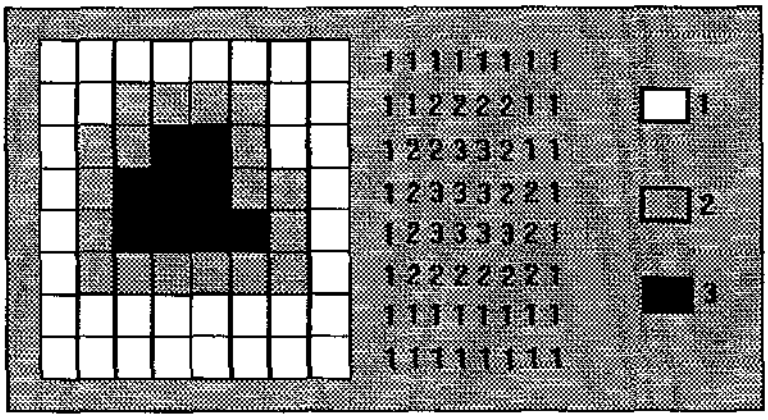


Figura 2.10 Representação de dados *raster* em uma matriz de células.

A passagem de uma representação contínua, em camada, para uma representação discreta, matriz de células, pode gerar alguns problemas. O primeiro problema consiste em determinar a categoria de uma célula cuja geometria e localização abrangem várias categorias no mundo real. Neste caso, deve ser associado somente o valor da categoria dominante, o que gera uma perda de dados. O conceito de categoria dominante varia de uma aplicação para outra, sendo que esta pode ser a categoria que ocupe a maior área dentro da célula, a categoria que ocupe o centro da célula ou simplesmente a categoria de maior valor, indicando assim uma ordem crescente de importância. Dependendo do tamanho da célula, a perda pode ser significativa (figura 2.11). Nesta figura, uma região geográfica contendo três tipos de vegetação, numeradas de 1 a 3, é armazenada em duas matrizes de células, uma envolvendo um tamanho de célula relativamente pequeno, matriz A, e outro envolvendo um tamanho de célula relativamente grande, matriz B. A categoria dominante para este exemplo é a categoria que ocupa a maior área dentro da célula. A categoria 1, quando mapeada na matriz A, consegue ser representada por três células. Já esta mesma categoria, quando mapeada na matriz B, não consegue ser representada em nenhuma célula. Assim, a matriz de células B derivada desta camada ignora totalmente a existência da categoria 1. Para minimizar situações como esta deve-se ter um tamanho de célula no mínimo igual à metade do tamanho do menor objeto geográfico que se deseja representar. Um segundo problema inerente à passagem de uma representação contínua, camada, para uma representação discreta, matriz de células, consiste na impossibilidade de se representar um valor de categoria em uma localização geográfica findada entre duas células. Isto é verdade, uma vez que a linha que separa duas células no formato varredura é considerada indefinidamente estreita.

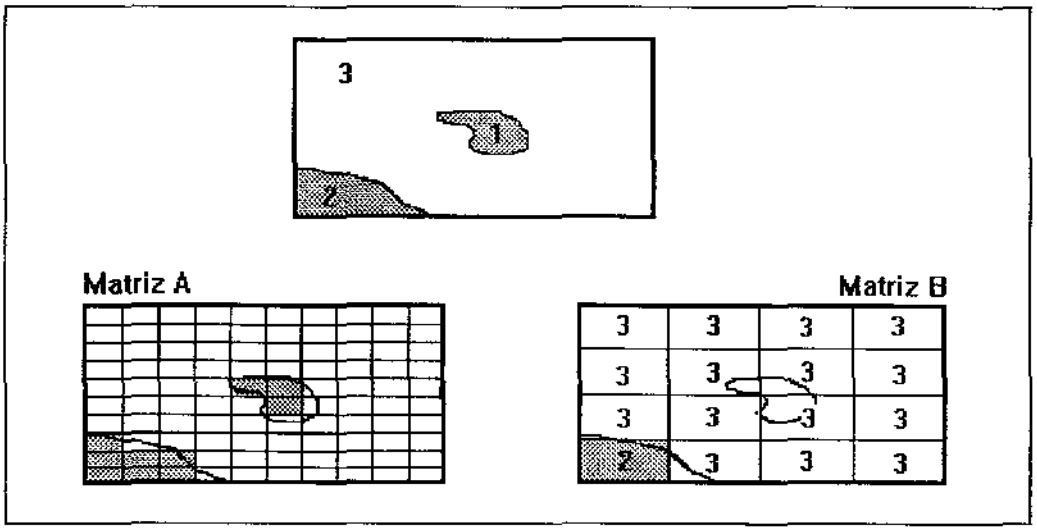


Figura 2.11 Tamanho da célula x precisão.

Atualmente, dados no formato varredura são obtidos principalmente por sensoriamento remoto. A quantidade de dados enviada por um satélite para cobrir uma dada região é enorme. [Sto+93] calcula em 1 *Gbyte*, 17 *Gbytes* e 2 *Tbytes* a quantidade de dados *raster* necessárias para mapear um estado americano como a Califórnia, o Estados Unidos e a superfície terrestre inteira, somente de alguns temas, com base nos satélites NOAA com

sensores AVHRR (*Advanced Very High Resolution Radiometer*). Estes satélites decompõem a terra em células quadráticas de 1 km de largura. O projeto Sequoia 2000 [SD92, Sto94] espera armazenar em torno de 100 *Tbytes* de dados (10^{14} *bytes*) nos seus primeiros anos, sendo que quando completado o projeto deve-se ter armazenado algo em torno de 10 *Petabytes* de dados (10^{16} *bytes*), os quais em sua maioria serão dados no formato varredura. Para minimizar a quantidade de dados armazenados são usados algoritmos de compactação de dados, tais como *run-length encoding, value point encoding* e *chain codes* [SE90, Sil94]. Entretanto, o uso de algoritmos de compactação prejudica a determinação de relações topológicas. Na literatura SIG o termo codificação de células corresponde ao processo de compactação de dados.

Devido à estrutura matricial de armazenamento de dados no formato varredura, algoritmos de análise são basicamente seqüenciais, onde lê-se os valores célula por célula em cada uma das matrizes envolvidas, efetuando-se para cada célula lida alguma operação simples que resulta também na escrita seqüencial de uma matriz de saída. Entretanto, como citado anteriormente, o grande volume de dados, o uso de técnicas de compactação, a imprecisão na representação de localizações geográficas, além da necessidade de derivação destas localizações a partir da posição em que a célula ocupa na matriz, incrementa a complexidade destes algoritmos.

O principal método de acesso utilizado para indexar dados no formato varredura é a *quadtree* [Peu90, SE90, Cox91]. Este método de acesso consiste em uma estrutura hierárquica, tipo piramidal, na qual uma dada camada é representada em vários níveis. Cada nível contém um tamanho de célula, sendo que cada nível imediatamente inferior possui um tamanho de célula menor e múltiplo do tamanho de célula do nível imediatamente superior. Assim, a medida em que se desce os níveis, mais detalhes se obtém sobre uma dada camada.

O modelo baseado em objetos (*object model*) representa a superfície terrestre como um conjunto de objetos geográficos, os quais existem independentemente. Estes objetos não estão necessariamente associados a um tema específico e podem inclusive ocupar a mesma localização geográfica. A ênfase deste modelo é a representação precisa da geometria de objetos, a qual é efetuada através do armazenamento de coordenadas geográficas de pontos, linhas e polígonos. Existem várias maneiras de se estruturar o armazenamento de coordenadas geográficas de pontos, linhas e polígonos, derivando vários modelos específicos a partir do modelo baseado em objetos. Os principais modelos são o modelo total (*Whole Polygon Structure*), o modelo *spaghetti*, o modelo topológico, o modelo DIME (*Dual Independent Map Encoding*), o modelo Arc-Node e o modelo de objetos relacional. Dados modelados nestes modelos são ditos estarem no formato vetorial (*vector format*).

O modelo total é a forma mais simples de estruturação de dados no formato vetorial (figura 2.12). Polígonos constituem a estrutura básica na qual são armazenadas coordenadas geográficas. Para cada polígono é armazenado um conjunto de coordenadas x,y indicando seus vértices. Dados convencionais e gráficos são armazenados

individualmente para cada polígono. Apesar de simples, este modelo apresenta alguns problemas. Neste modelo ocorre redundância no armazenamento de coordenadas geográficas, uma vez que vértices de polígonos adjacentes são armazenados várias vezes. Além disto, mantendo cada polígono separado esconde-se relacionamentos topológicos, principalmente adjacência.

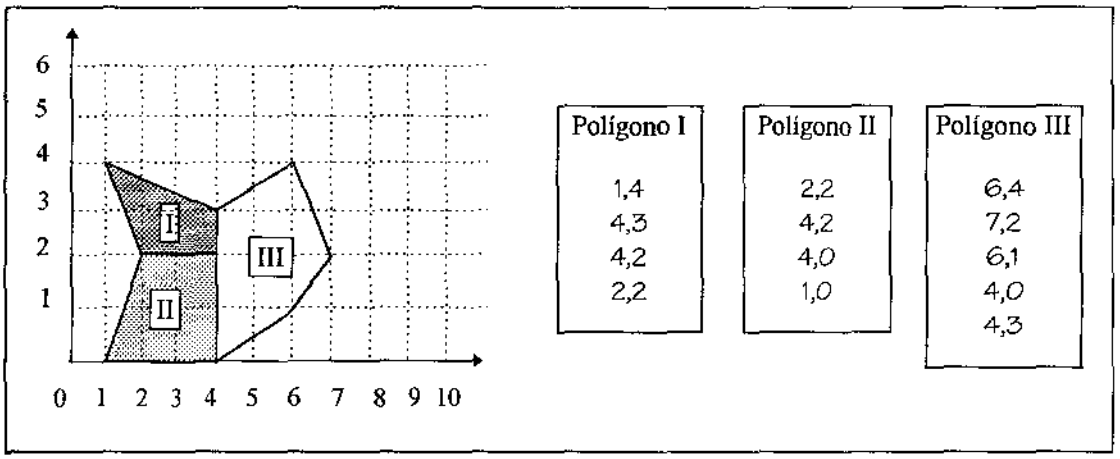


Figura 2.12 Modelo total.

O modelo *spaghetti* armazena separadamente para cada ponto, linha e polígono suas coordenadas geográficas, sem nenhuma estruturação. Dados convencionais e gráficos também são armazenados individualmente para cada um destes elementos. Da mesma maneira que o modelo total, este modelo apresenta redundância no armazenamento de coordenadas, além de esconder relacionamentos topológicos. Seu uso deve-se principalmente à facilidade de "*plottagem*", uma vez que os dados são impressos na mesma seqüência em que estão armazenados (figura 2.13).

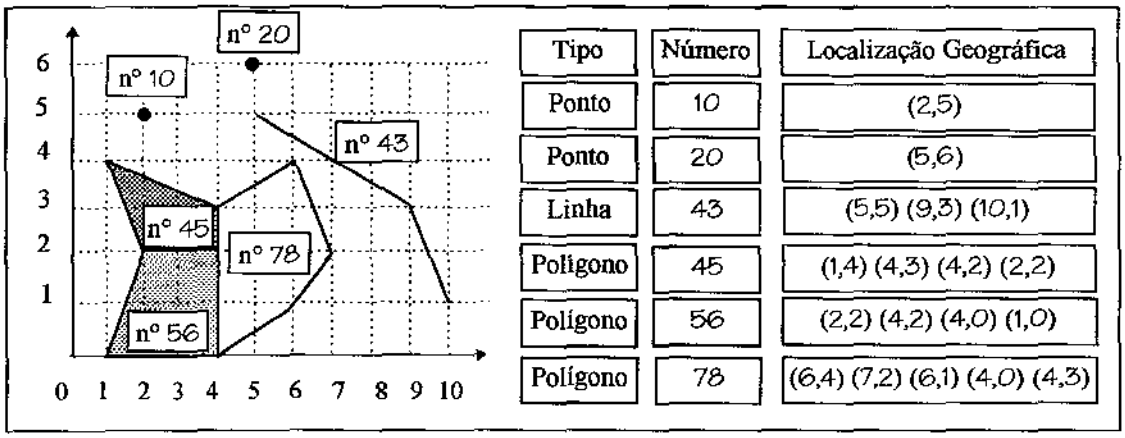
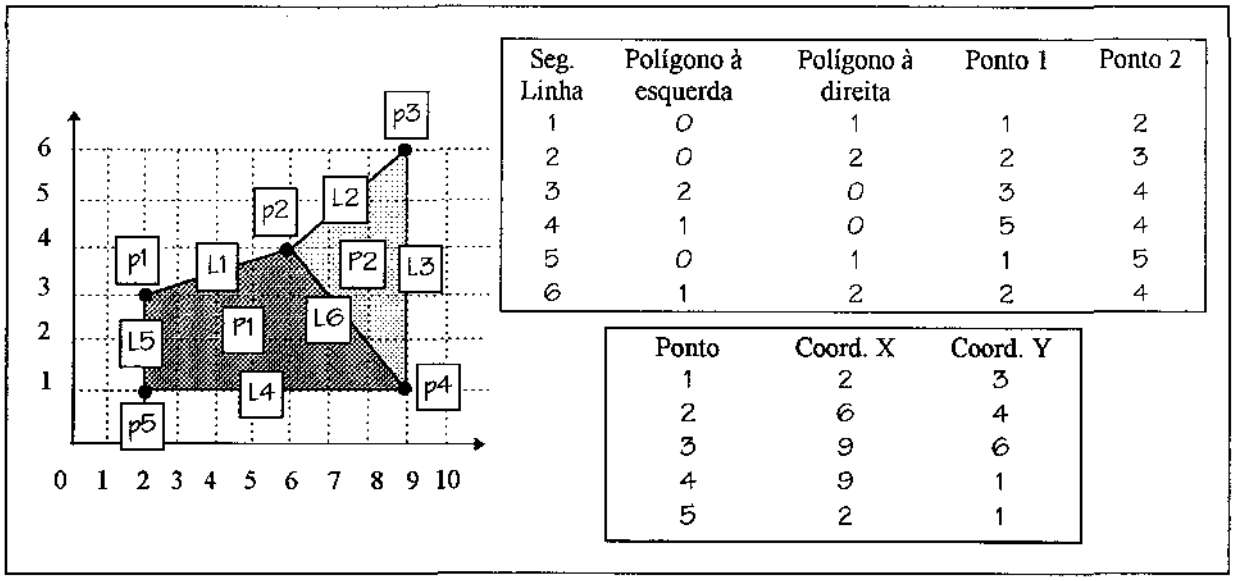


Figura 2.13 Modelo *spaghetti*.

O modelo topológico armazena apenas localizações geográficas relativas a pontos, em uma matriz contendo o número do ponto, e as coordenadas relativas aos eixos x e y. Em

outra matriz, este modelo armazena dados topológicos relativos a segmentos de linhas. São armazenados o número dos pontos que constitui cada segmento, permitindo desta forma a obtenção das coordenadas geográficas, e os polígonos que estão à esquerda e à direita, para se obter explicitamente relacionamentos topológicos de adjacência entre polígonos. As coordenadas dos polígonos são obtidas pesquisando-se esta tabela para se obter os segmentos que possuem este polígono como polígono à esquerda ou à direita, sendo que esta busca por ser demorada constitui a sua principal desvantagem. Por outro lado, este modelo não possui redundância de armazenamento, assim como representa explicitamente o relacionamento de adjacência entre polígonos (figura 2.14).

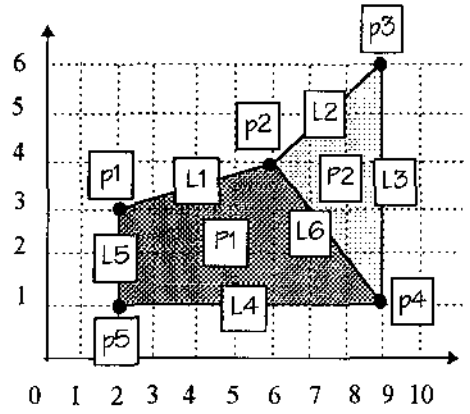


Seg. Linha	Polígono à esquerda	Polígono à direita	Ponto 1	Ponto 2
1	0	1	1	2
2	0	2	2	3
3	2	0	3	4
4	1	0	5	4
5	0	1	1	5
6	1	2	2	4

Ponto	Coord. X	Coord. Y
1	2	3
2	6	4
3	9	6
4	9	1
5	2	1

Figura 2.14 Modelo topológico.

O modelo DIME (*Dual Independent Map Encoding*) foi desenvolvido para ser usado pelo *U.S. Bureau of the Census* em 1967, sendo concebido para incorporar informações topológicas de áreas urbanas para o uso em análises demográficas. Embora os arquivos tipo DIME geralmente não correspondam à organização interna de um SIG, eles são comumente usados como formato comum de troca de dados. O componente básico deste modelo são os segmentos de linhas. Cada segmento é armazenado possuindo três componentes essenciais: um nome que o identifica (nome da rua), os nomes dos nós onde o segmento inicia e termina (orientado), e finalmente, um outro componente indicativo de polígonos, mostrando se eles estão do lado esquerdo ou direito de um determinado segmento. Adicionalmente, vários dados convencionais podem ser armazenados conjuntamente a estes três componentes, como exemplo numeração superior e inferior de cada segmento. Atributos comuns a vários segmentos podem ser armazenados separadamente, tal como CEP, os quais são referenciados através de um campo adicional que possui um ponteiro ou índice onde estes dados estão armazenados. Neste modelo, a localização geográfica de cada nó é armazenado em uma estrutura separada, geralmente em UTM (*Universal Transverse Mercator*), com o propósito de se evitar redundância no armazenamento (figura 2.15).



Ponto	Coord. X	Coord. Y
1	2	3
2	6	4
3	9	6
4	9	1
5	2	1

Header	Cep	Código de Área
1000	14033-234	13b
1001	12345-908	43a

Seg. Linha	Polígono à esquerda	Polígono à direita	Ponto 1	Ponto 2	Numeração Inferior	Numeração Superior	Header
1	0	1	1	2	42	70	1000
2	0	2	2	3	73	98	1000
3	2	0	3	4	567	656	1000
4	1	0	4	5	234	456	1001
5	0	1	5	1	112	178	1001
6	1	2	2	4	12	67	1000

Figura 2.15 Modelo DIME.

O modelo Arc-Node armazena apenas localizações geográficas relativas a pontos, em uma organização hierárquica de pontos, linhas e polígonos. Polígonos são armazenados separadamente contendo além do número de identificação, os segmentos de linhas que o formam, e dados convencionais quaisquer. Arcos, conotando segmentos de linhas, também são armazenados separadamente, contendo o número de identificação do arco, os pontos de início e fim, além de dados convencionais. Por último, pontos são armazenados contendo o número de identificação, coordenada x, coordenada y e dados convencionais (figura 2.16).

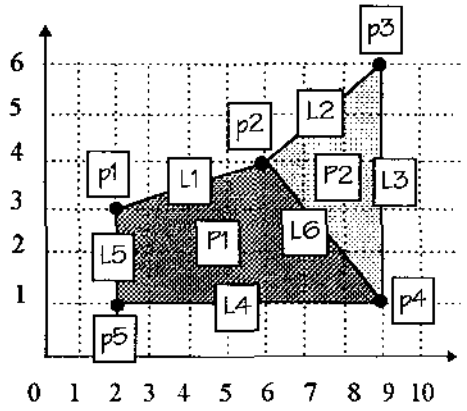
Já o modelo de objetos relacional possui estrutura semelhante ao modelo Arc-Node, porém adaptado ao modelo relacional (figura 2.17). Este modelo tem se mostrado muito popular em vários SIGs comerciais.

Devido à representação compacta de coordenadas geográficas através de pontos, linhas e polígonos, principalmente nos modelos Arc-Node e de objetos relacional, o volume de dados armazenados no formato vetorial é sensivelmente reduzido, quando comparado com a quantidade necessária para armazenar as mesmas informações no formato varredura.

O modelo baseado em objetos e o modelo baseado em campos são ditos serem duais. Enquanto o modelo baseado em objetos considera objetos geográficos como unidade básica nas quais dados geográficos são associados, o modelo baseado em campos dá ênfase no conteúdo de áreas geográficas mais que nos seus limites. O uso destes dois modelos tornou-se muito difundido em aplicações georeferenciadas, tornando necessário algoritmos de conversão de dados entre os dois formatos, especialmente entre o formato *raster* e os formatos vetoriais relacional e Arc-Node. O conceito de objetos também tem sido implementado em SIGs que utilizam o modelo baseado em campos. Para isto, células são agrupadas logicamente para ilustrar os conceitos de ponto, linha e polígono. Assim, uma linha no formato varredura é constituída de uma seqüência de células, as quais se aproximam do conceito de linha. De maneira similar é formado um polígono. Já um ponto geralmente é representado por uma única célula. Obviamente, a representação destes conceitos no formato vetorial é feita de maneira mais precisa, uma vez que isto é a sua base. Por outro lado, sistemas de informações geográficas que utilizam dados no formato vetorial têm implementado o conceito de categoria para este formato. Isto é feito associando um dado convencional, o qual representa a categoria, a cada polígono de um dado tema. Assim, polígonos são estruturados de acordo com o tema a que pertencem.

Polígono	Proprietário	Arcos	Perímetro	Área	Zona
1	Estado	1,6,4,5	478.2	11562	B3
2	João Saldanha	2,3,6	322.7	8765	B4

Arco	Ponto Inicial	Ponto Final	Comprimento	Estado Pavimento
1	1	2	3,16	bom
2	2	3	3,60	regular
3	3	4	5,00	bom
4	4	5	7,00	péssimo
5	5	1	2,00	bom
6	2	4	4,24	bom



Ponto	Coord. X	Coord. Y
1	2	3
2	6	4
3	9	6
4	9	1
5	2	1

Figura 2.16 Modelo Arc-Node.

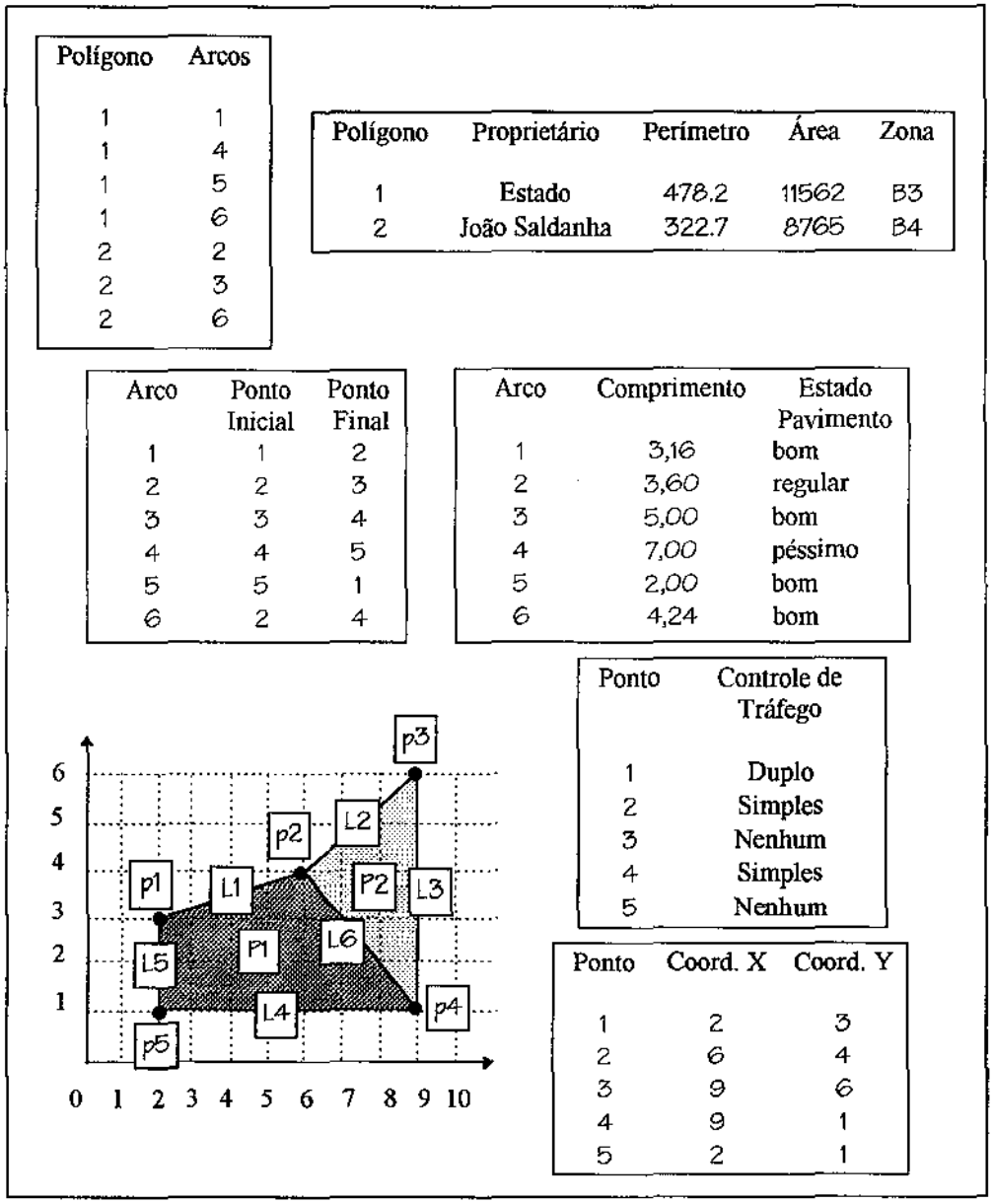


Figura 2.17 Modelo de objetos relacional.

2.4 Aplicações

Sistemas de informações geográficas são usados principalmente em aplicações de planejamento e administração, os quais podem ser divididos em dois grupos distintos: cadastral (urbano e rural) e ambiental. O primeiro grupo visa manter informações sobre os limites de ruas, avenidas, praças, propriedades particulares, escolas, hospitais, córregos, entre outros, para que se possa desenvolver atividades tais como controle de cobrança do

IPTU (Imposto Predial e Territorial Urbano), serviços de correio, planejamento viário, etc. Para este grupo os dados em geral são predominantemente no formato vetorial. Já o segundo grupo visa manter informações sobre os recursos naturais disponíveis, para que seja possível a exploração racional destes recursos, bem como garantir a qualidade do meio ambiente. Para este grupo os dados em geral são predominantemente no formato varredura.

2.4.1 Aplicações urbanas e rurais

A seguir serão descritas as principais aplicações envolvendo o planejamento e a administração cadastral urbano e rural. Para cada uma delas, uma lista não exaustiva de dados necessários é apresentada. Para facilitar a visualização destas aplicações, estas são separadas de acordo com as atividades que estão ligadas.

- **mapeamento urbano básico:** como citado em [Agu95], qualquer aplicação de planejamento e administração urbana e rural necessita de dados relativos a lotes, logradouros, linhas centrais, quadras, divisas, entre outros, os quais são denominados elementos do mapeamento urbano básico. O armazenamento e manutenção destes elementos constitui-se, portanto, de uma tarefa estritamente necessária. Para as aplicações descritas a seguir, considera-se que os dados relativos a estes elementos existam e sejam bem gerenciados. Um exemplo de aplicação que utiliza os elementos do mapeamento urbano básico é descrita em [AS94], a qual consiste na análise, utilizando-se SIGs, dos efeitos de desapropriação de terrenos em virtude da construção de uma auto-estrada.

- **cadastro imobiliário:** uma das principais fontes de renda de um município é o imposto predial e territorial urbano. Para agilizar a cobrança deste imposto deve-se ter dados relativos aos lotes (geometria, valor venal, valor de aluguel, setor, entre outros) e as construções (geometria, área construída, estilo arquitetônico, idade da construção, altura, entre outros), além de dados do proprietário (nome, CPF, RG, endereço, entre outros) destes lotes. [Rod+94] descreve a especificação de um sistema para atender as necessidades de gestão espacial e administrativa de imóveis em uma empresa pública.

- **transporte:** uma das principais preocupações em cidades de médio e grande porte é garantir o funcionamento eficiente dos meios de transporte, possibilitando o deslocamento de um grande número de pessoas em um tempo relativamente curto, com um custo mínimo. Para planejar este tipo de atividade são necessários dados sobre as redes de transporte (viária, ferroviária, fluvial, etc). Também é necessário dados populacionais (densidade demográfica/setor). Como exemplo deste tipo de planejamento pode-se citar: distribuição de linhas de ônibus para os diversos setores de uma cidade [AA94, RC94], determinação de sinalização, orientação do trânsito,

roteamento turístico [GC94, Mag94], entre outros. A partir das redes de transporte pode-se também efetuar o roteamento da coleta de lixo.

- **educação:** com a crescente demanda pelo ensino público, devido à crônica crise econômica, torna-se necessário um melhor gerenciamento da rede de ensino. Para isto, deve-se planejar a expansão da rede de forma que esta atenda toda a população, e ao mesmo tempo não se torne ociosa. Este tipo de planejamento se baseia em dados referentes à distribuição da população em idade escolar, localização, tipo e capacidade das escolas. Um exemplo real é efetuado em Belo Horizonte, onde a prefeitura realiza a matrícula de alunos através do correio, eliminando filas e ociosidade [DF94].

- **zoneamento:** este tipo de planejamento envolve a divisão da cidade em setores de maneira que cada setor seja utilizado para a atividade mais indicada na região em que ele se localiza. Assim, têm-se setores propícios para a instalação de indústrias, de comércio, de residências, de zonas de lazer, entre outras. Para realizar o zoneamento da cidade deve-se dispor de dados referentes à ocupação do solo, distribuição populacional, áreas de conservação, rede hídrica, geologia e geofísica do terreno, para citar alguns.

- **saúde:** a distribuição de hospitais e postos de saúde de modo a atender a todos os seguimentos populacionais também representa uma tarefa essencial. Um SIG pode auxiliar no planejamento desta tarefa indicando os pontos onde estes hospitais e postos de saúde devem ser localizados. Para isto é preciso dados sobre distribuição populacional, dados socio-econômicos, rede de transporte, localização, capacidade e especialidade dos centros de saúde existentes, entre outras coisas. Similarmente, um SIG pode ajudar no acomodamento de doentes em hospitais, desde que este seja alimentado com alguns dados relativos a hospitais, tais como leitos ociosos, especialidades e custo. Assim, quando um paciente necessitar de um leito, o SIG pode fornecer o hospital geograficamente mais perto que atenda os requisitos médicos deste paciente. Adicionalmente, pode-se monitorar áreas onde estão ocorrendo epidemias de modo a controlá-las mais facilmente. Em especial, pode-se também controlar a mortalidade infantil.

- **habitação:** a criação de novos loteamentos para suportar o aumento populacional, evitando-se assim o crescimento de favelas, é uma questão a ser destacada no planejamento urbano. É preciso garantir que estes novos loteamentos se situem em áreas adequadas para a expansão urbana. Estas áreas são determinadas com base nos dados sobre uso do solo, rede hídrica, áreas de conservação, geologia e relevo do terreno, entre outros.

- **água e esgoto:** o saneamento básico é uma questão fundamental, desde que este evita a disseminação de doenças entre a população, principalmente as mais carentes. Para realizar o seu planejamento é necessário dados sobre a rede hídrica, densidade e distribuição populacional, geologia, uso do solo, entre outros. [Sie94] descreve o gerenciamento da rede de distribuição de água da Sanepar (Companhia de Saneamento do Paraná S/A).

- **energia elétrica e telecomunicações:** no mundo atual, onde a globalização torna-se cada vez mais importante, o gerenciamento eficiente destes recursos é fundamental. Para que isto seja feito, um SIG deve dispor de dados relativos às redes de distribuição existentes, e dados relativos aos consumidores. [Arg+93, Mag93, Mag+94] descrevem o gerenciamento da rede de telefonia da Telebrás (Telecomunicações Brasileiras S/A), através do projeto SAGRE (Sistema Automatizado de Gerência de Rede Externa). Similarmente, [Web94a, Web94b] e [CSV94], descrevem o gerenciamento da rede de energia elétrica, da empresa estatal Eletropaulo (Eletricidade de São Paulo S/A) e da Celpe (Companhia Energética de Pernambuco S/A), respectivamente.

- **controle populacional:** o controle das contínuas mudanças demográficas de uma cidade, através do registro da distribuição espacial da população ao longo dos anos, ajuda a subsidiar o planejamento de planos e programas de desenvolvimento. Para isto dados censitários, tais como número de componentes de uma família, renda média familiar, faixas etárias e percentual de população masculina e feminina precisam estar georeferenciados.

- **segurança:** o crescente aumento da criminalidade, principalmente em médias e grandes cidades, torna necessário o monitoramento das áreas mais afetadas. Desta forma, pode-se associar crimes às características populacionais, criando-se condições para o estabelecimento de um política de combate aos crimes, preferencialmente sem o uso de violência. [AX94] descreve o uso de SIGs para a identificação de modelos de crimes no estado da Carolina do Norte, Estados Unidos.

2.4.2 Aplicações ambientais

A seguir é descrito, para cada atividade ambiental, as principais aplicações e as informações georeferenciadas que seriam necessárias para atendê-las, com ênfase na exploração e conservação de recursos:

- **recursos hídricos:** o interesse é explorar o potencial energético (energia hidroelétrica), de transporte (rios navegáveis), de abastecimento de água, e de pesca. É importante que a exploração destes recursos mantenha um nível de poluição aceitável no que se refere ao esgoto, lixo industrial e agrotóxicos. Portanto seriam necessários dados georeferenciados sobre cada um destes fatores para que seja

possível o planejamento desejado. [SC94] descreve um estudo ambiental do assoreamento da bacia hidrográfica do rio Manso apoiado no uso de SIGs.

- **fauna e flora:** os aspectos relativos ao uso e conservação destes recursos envolve a determinação de áreas propícias à agricultura [Roc94], ao reflorestamento, à exploração de madeira, à conservação, à criação de reservas biológicas, etc. Os dados necessários para a realização destas atividades envolve dados de vegetação, de solo (relevo e geologia) toponímia (localização de indústrias e centros consumidores), rede hídrica, rede de transporte, distribuição da fauna, etc. Em adição, pode-se monitorar e controlar pragas e doenças. Através do levantamento e monitoramento de áreas infectadas e dados de recobrimento florestal, consegue-se avaliar o comportamento das infestações e estabelecer medidas de combate rápidas e eficientes [Cor94]. A previsão de safras também pode ser efetuada com a ajuda de SIGs, conforme descrito em [Mor+94].

- **minerais:** é preciso planejar a extração e o beneficiamento dos recursos minerais evitando os problemas causados pela poluição resultante da exploração mineral. Este planejamento envolve informações relativas ao potencial mineral da região, áreas de conservação, rede hídrica, população, uso do solo, nível e tipo de poluição gerada pelas atividades minerais, entre outros. A COMIG (Companhia Mineradora de Minas Gerais) está atualmente implantando, com o auxílio do Instituto de Geociências da Unicamp, um SIG para suportar estas informações [SP94].

- **monitoramento de catástrofes naturais:** exemplos são o monitoramento de queimadas [Bat+94], o monitoramento de furacões e tornados [Dym94], o monitoramento de terremotos e o monitoramento de nevascas.

Capítulo 3

Análise de desempenho

3.1 Introdução

Com o desenvolvimento da indústria de informática diversos produtos de *hardware* e de *software* surgiram. Muitos destes produtos, lançados por companhias distintas, passaram a oferecer as mesmas ou semelhantes funcionalidades. Com isto, o surgimento de mecanismos que pudessem medir e comparar estes produtos, sobre determinados aspectos, tornou-se necessário. A análise de desempenho é um dos mecanismos mais utilizados para tal propósito. Esta técnica consiste em medir a eficiência com a qual produtos efetuam uma determinada funcionalidade. A eficiência é reportada através de alguma medida escalar, tais como tempo e distância. Assim, a comparação de vários produtos similares pode ser efetuada facilmente através do uso dos operadores relacionais >, < e = para a ordenação dos resultados de desempenho produzidos a partir de uma dada medida escalar.

A análise de desempenho é efetuada em quase todos os ramos da ciência. Um exemplo típico de seu uso, fora da área da ciência da computação, é encontrada fartamente em revistas especializadas na área de veículos automotores. Estas revistas freqüentemente analisam o desempenho dos veículos no tocante a velocidade máxima, retomada de velocidade, aceleração, aceleração lateral, espaço de frenagem, nível de ruído, *mileage* (número de milhas percorridas em média com um litro de combustível), entre outros.

Na área da ciência da computação, a análise de desempenho é utilizada tanto para a análise de componentes de *hardware* quanto de *software*. Dentre as subáreas da ciência da computação que utilizam esta técnica pode-se citar: sistemas operacionais, arquitetura de computadores, teoria da computação e banco de dados. Para este trabalho, a área de banco de dados será detalhadamente abordada, desde que um dos componentes principais de SIGs é um SGBD não convencional, o qual é comprovadamente uma das maiores fontes de gargalo destes sistemas. As demais áreas serão abordadas apenas superficialmente, o suficiente para ter-se alguma idéia da influência que geram no desempenho de SIGs.

Em geral a análise de desempenho ajuda a resolver quatro questões principais:

- **avaliação de capacidade máxima:** a análise de desempenho pode predizer quão rápido, no máximo, um determinado produto efetua uma dada funcionalidade. Este é o objetivo original para o desenvolvimento do mecanismo de análise de desempenho. Entretanto, como será discutido na seção 3.5, o desempenho pode ser reportado por vários tipos de medidas, os quais utilizam o fator tempo de modo diferente, assim como pode envolver medidas que não utilizam o fator tempo para reportar o desempenho.

- **comparação entre diferentes tecnologias:** o uso de resultados de desempenho para efeitos comparativos ajuda a identificar restrições de desempenho em determinados produtos. Estas restrições indicam a necessidade de restruturação do produto para que este se torne comercializável. Muitas vezes, porém, a tecnologia envolvida com o produto está obsoleta, e nestes casos deve-se abandonar a linha de pesquisa empregada até então e empregar uma nova linha de pesquisa que produza tecnologia com resultados de desempenho no mínimo aceitáveis. Resultados de desempenho também são usados para a identificação de produtos que devem ser melhor pesquisados, e conseqüentemente melhorados, devido ao alto desempenho proporcionado pelas tecnologias empregadas nestes produtos.

- **avaliação de capacidade específica:** a análise de desempenho pode ser necessária para avaliar se uma determinada alternativa possui um nível de desempenho aceitável para uma determinada aplicação. Uma questão que sempre surge para um gerente de sistema é saber se a aplicação desejada pode ser executada convenientemente em um determinado sistema computacional. Um sistema computacional é formado por componentes de *hardware* e de *software*. Em relação ao *hardware*, os principais são: processador, memória principal, memória secundária e dispositivos de entrada/saída (também chamados de recursos computacionais). Em relação ao *software*, os principais são: sistema operacional[1], *software* básico (compilador, interpretador, SGBD, SIG, etc) e programas aplicativos.

- **avaliação da relação** *custo x benefício*: esta relação é fundamental no processo de seleção de produtos. A partir desta, o gerente de sistema tem condições de optar por um sistema computacional que se ajuste o mais próximo possível tanto às necessidades de desempenho de sua aplicação quanto às restrições de gastos imposta por sua empresa. Na seção 3.5 são apresentados outros fatores relevantes no processo de seleção de produtos.

Para se efetuar a análise de desempenho utiliza-se os seguintes modelos:

[1] o sistema operacional também é classificado como um *software* básico. Entretanto, devido à sua importância e particularidade, este será citado separadamente.

- **modelo analítico:** baseia-se na obtenção de um conjunto de equações matemáticas juntamente com os algoritmos para resolvê-las, que relacionam medidas de desempenho à parâmetros do sistema sendo analisado. Usualmente estes modelos empregam várias hipóteses teóricas a respeito do sistema de modo a simplificar o modelo obtido. Desta forma, os resultados gerados tendem a ser imprecisos e com isto seu uso em sistemas reais torna-se reduzido. Este modelo é freqüentemente utilizado durante a fase de projeto, onde ainda não se tem o sistema implementado e já quer-se fazer uma estimativa do seu comportamento. O custo relativamente baixo da implantação deste modelo é outro fator a ser destacado.

- **modelo de simulação:** segundo [Ber+91], a simulação baseada em computador implica na formalização de um fenômeno, de um processo, de um sistema ou de qualquer outra coisa do mundo real e na sua posterior implementação em um programa de computador. Este modelo pode possuir aspectos quantitativos e/ou qualitativos. Na simulação quantitativa, variáveis independentes, variáveis dependentes e parâmetros são combinados dentro de um modelo numérico. Na simulação qualitativa, o modelo não é puramente numérico, sendo que os componentes do modelo e as relações entre estes componentes são representados simbolicamente ou estruturalmente. Modelos de simulação consideram a entrada (*input*) e a saída (*output*) de dados como parte do mundo real, e portanto devem ser modelados. Assim, este modelo deve ser capaz de gerar dados de saída a partir de um processo de cálculo ou de inferência sobre os dados de entrada. Desta forma, o modelo de simulação procura reproduzir as atividades do sistema sendo analisado de acordo com um conjunto de hipóteses e condições, eliminando a necessidade de experimentação no próprio sistema.

- **modelo experimental:** também chamado de modelo empírico, utiliza o próprio sistema sendo analisado para a obtenção dos resultados de desempenho. Desta forma, os resultados obtidos na análise de desempenho de sistemas são altamente confiáveis. Duas técnicas principais fazem parte deste modelo: monitoração e *benchmark* [Mag81]. A técnica de monitoração consiste em utilizar ferramentas próprias de avaliação estatística presentes no sistema sendo analisado. Devido à ausência de padronização destas ferramentas, seu uso torna-se específico e portanto limitado. Segundo [BD84, Dew85], a falta de padronização na técnica de monitoração inibe a comparação dos resultados gerados em diferentes sistemas. Já a técnica de *benchmark* consiste na execução de um conjunto fixo de testes sobre um sistema para avaliar o seu desempenho. Para a análise de desempenho de sistemas computacionais, esta técnica consiste na execução de um conjunto fixo de programas sobre um determinado sistema computacional. A técnica de *benchmark*, ao contrário da técnica de monitoração, pode ser aplicada na comparação de diferentes sistemas, uma vez que é padronizada, sendo os testes invariáveis e bem definidos.

Modelos experimentais dependem da maturidade da tecnologia envolvida. Ao contrário, os modelos analíticos e de simulação podem ser empregados em sistemas nascentes, onde ainda não se têm dados operacionais disponíveis. Dentro do contexto de qualquer

tecnologia, o modelo experimental deve ser o último a poder ser utilizado, já que ele necessita que todos os componentes do sistema já tenham sido implementados.

Em alguns casos é possível utilizar modelos híbridos, envolvendo mais de um dos modelos citados anteriormente. Estes modelos híbridos procuram associar as vantagens de cada um dos modelos individuais de forma a obter uma representação do sistema mais realista ou, ainda, procurando facilitar a execução das ferramentas criadas. Um típico exemplo de modelo híbrido é o modelo descrito em [EH84], onde utiliza-se o modelo experimental de monitoração para se obter os dados necessários para a criação de um modelo de simulação que avaliará as possíveis alternativas de gerência de *buffers-pool*.

3.2 Técnica de *benchmark*

A técnica de *benchmark*, conforme descrito anteriormente, consiste em um modelo de análise experimental onde é executado um conjunto fixo de testes sobre um sistema para avaliar o seu desempenho. Em particular, esta técnica é largamente utilizada para avaliação de desempenho de sistemas computacionais. Neste caso, um conjunto fixo de programas é executado nestes sistemas para gerar resultados de desempenho [Per90, CB92]. Para sistemas que possuam como um de seus principais componentes um SGBD, utiliza-se a técnica de *benchmark* de banco de dados, uma especialização da técnica genérica de *benchmark*, onde transações são definidas e posteriormente executadas para a obtenção de resultados de desempenho.

Segundo [Mcl90], um *benchmark* é composto por uma série representativa de testes funcionais e de desempenho, os quais são efetuados em um determinado subconjunto de dados, simulando assim o ambiente da aplicação alvo. O fato desta técnica utilizar o próprio sistema sendo analisado para a obtenção dos resultados de desempenho torna os resultados gerados altamente confiáveis. Por outro lado, a técnica de *benchmark* requer que o sistema a ser testado esteja pronto e disponível para uso. Em adição, a implantação desta técnica exige um custo relativamente elevado. Desta forma, a técnica de *benchmark* deve ser utilizada apenas em sistemas que já atingiram uma certa maturidade, os quais necessitam de testes extensivos de desempenho. Segundo [Tur87], isto já ocorre com a tecnologia de banco de dados, principalmente relacional. Analogamente, [GR87, Mar90, Gra91, Sto+93] citam que a tecnologia SIG também já atingiu a maturidade mínima para a criação de um *benchmark* voltado à sua análise.

É importante que um *benchmark* possua quatro características: ser relevante (representativo) para a aplicação alvo à qual representa, ser portável entre diferentes arquiteturas e configurações de sistemas computacionais, ser escalável, podendo ser executado desde pequenos computadores pessoais até grandes *mainframes*, e ser simples de entender, caso contrário este perderá a sua credibilidade. Apesar destas características

algumas vezes serem conflitantes, principalmente no que diz respeito à simplicidade, estas devem ser alcançadas conjuntamente sempre que possível.

O uso de *benchmarks*, porém, não está restrito apenas à análise de desempenho. Em muitas situações pode-se utilizar esta técnica para testes de verificação de correção de implementação e de conformidade com determinados modelos de SGBDs. Nestes casos, devido à grande complexidade dos sistemas a serem testados, como é o caso de SGBDs, a verificação de correção através de comprovações teóricas ainda é inviável. Entretanto, a aplicação de *benchmarks* é mais freqüente para análise de desempenho e sua utilização em testes de correção de programas ocorre mais comumente durante os estágios iniciais de implementação de um SGBD, tal como ocorreu nas primeiras implementações de SGBDs utilizando o modelo relacional, Ingres e System R.

Para evitar-se ambigüidade, para todo o restante da tese qualquer citação de *benchmark* deverá ser entendido implicitamente como *benchmark* de sistemas computacionais, mais especificamente *benchmark* de banco de dados. As próximas seções descreverão os principais conceitos relacionados a *benchmarks*, dentre os quais pode-se citar: carga de trabalho, sistema de análise, dados e medidas. A definição destes conceitos é orientada a sistemas centralizados.

3.3 Carga de trabalho

O objetivo principal de um *benchmark* é a avaliação de desempenho de sistemas computacionais. Basicamente, quer-se medir quão rápido um dado sistema computacional efetua um determinado conjunto de tarefas. Entretanto, a eficiência de um sistema computacional está diretamente relacionada aos fatores que degradam o seu desempenho. Esta degradação, provocada pela sobrecarga de execução dos processos[2] ativos no sistema, limita a velocidade na qual estas tarefas são executadas. Isto ocorre porque cada um destes processos ativos gera uma sobrecarga de execução específica, a qual constitui-se da quantidade de recursos computacionais utilizados e do conseqüente tempo gasto durante o uso destes recursos.

A avaliação de desempenho através da técnica de *benchmark* implica necessariamente na implementação e posterior execução de um conjunto de programas em um dado sistema computacional. Deste modo, ao se avaliar o desempenho, insere-se uma sobrecarga de execução específica, gerada pelo próprio *benchmark*. A esta sobrecarga de execução dá-se o nome de carga de trabalho do *benchmark*. A carga de trabalho de um *benchmark* engloba todos os processos criados ou acionados para execução, diretamente ou indiretamente,

[2] o termo processo [Gui80, Dei90, Tan92b] largamente utilizado na área de sistemas operacionais, consiste basicamente em um programa em execução. Este conceito foi criado para permitir a implantação dos conceitos de multiprogramação e multiprocessamento.

pelo *benchmark*. Adicionalmente à carga de trabalho do *benchmark*, pode-se ter processos independentes, os quais geram uma sobrecarga de execução particular, chamada carga de trabalho externa ao *benchmark*. Estes processos independentes, ao utilizarem recursos computacionais, também degradam o desempenho do sistema computacional como um todo. Assim, os processos ativos no sistema podem ser divididos em dois grupos: processos pertencentes à carga de trabalho do *benchmark* e processos independentes. A sobrecarga de execução gerada por estes dois grupos caracteriza a carga de trabalho total existente em um determinado sistema computacional, a qual é ilustrada na figura 3.1. Esta carga de trabalho envolve somente aspectos de *software*, ou seja, a sobrecarga de execução gerada por um conjunto de processos.

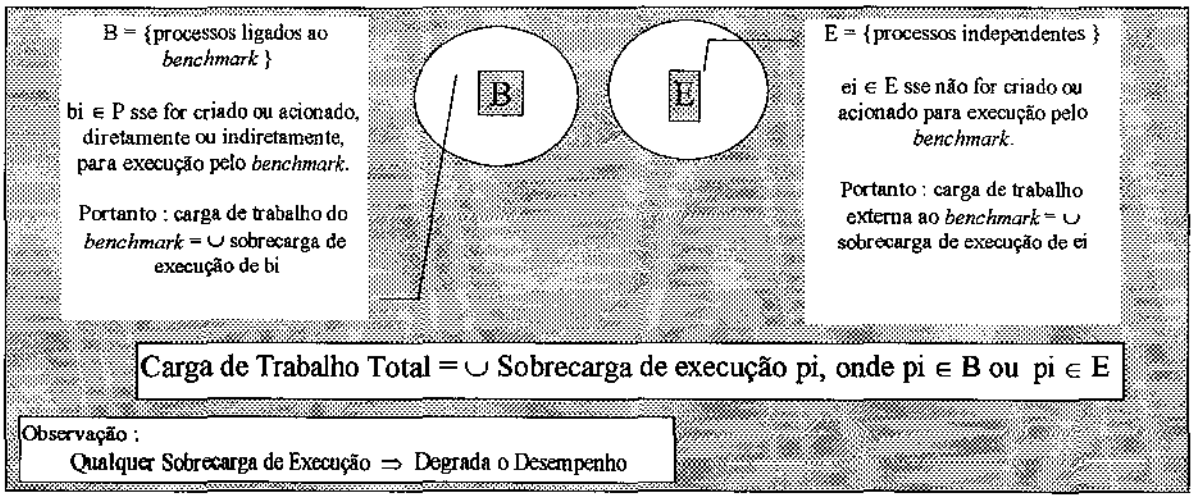


Figura 3.1 Carga de trabalho durante testes de desempenho.

Genericamente, os processos ativos em um sistema computacional podem ser oriundos de qualquer nível de *software*. Em um sistema computacional multinível, cada nível esconde os detalhes provenientes do nível imediatamente inferior, acrescenta novas funcionalidades e oferece ao nível imediatamente superior um ambiente mais simples através da disponibilização de primitivas de mais alto nível (figura 3.2). Desta forma, o sistema computacional torna-se mais simplificado, nível a nível, sendo o nível mais superior voltado ao usuário final. Segundo [Tan92a], os sistema computacionais atuais seguem esta tendência de desenvolvimento, sendo chamados de sistemas computacionais multiníveis, máquinas multiníveis ou ainda máquinas virtuais. A figura 3.3 mostra de maneira simplificada o esquema de um sistema computacional multinível contendo os componentes genéricos, de cada nível, que pertencem à carga de trabalho de um *benchmark*.

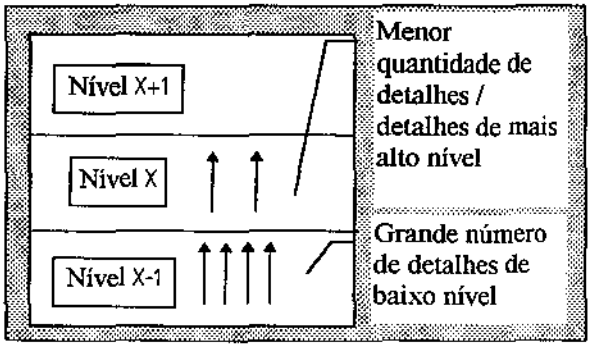


Figura 3.2 Esquema de máquina multinível.

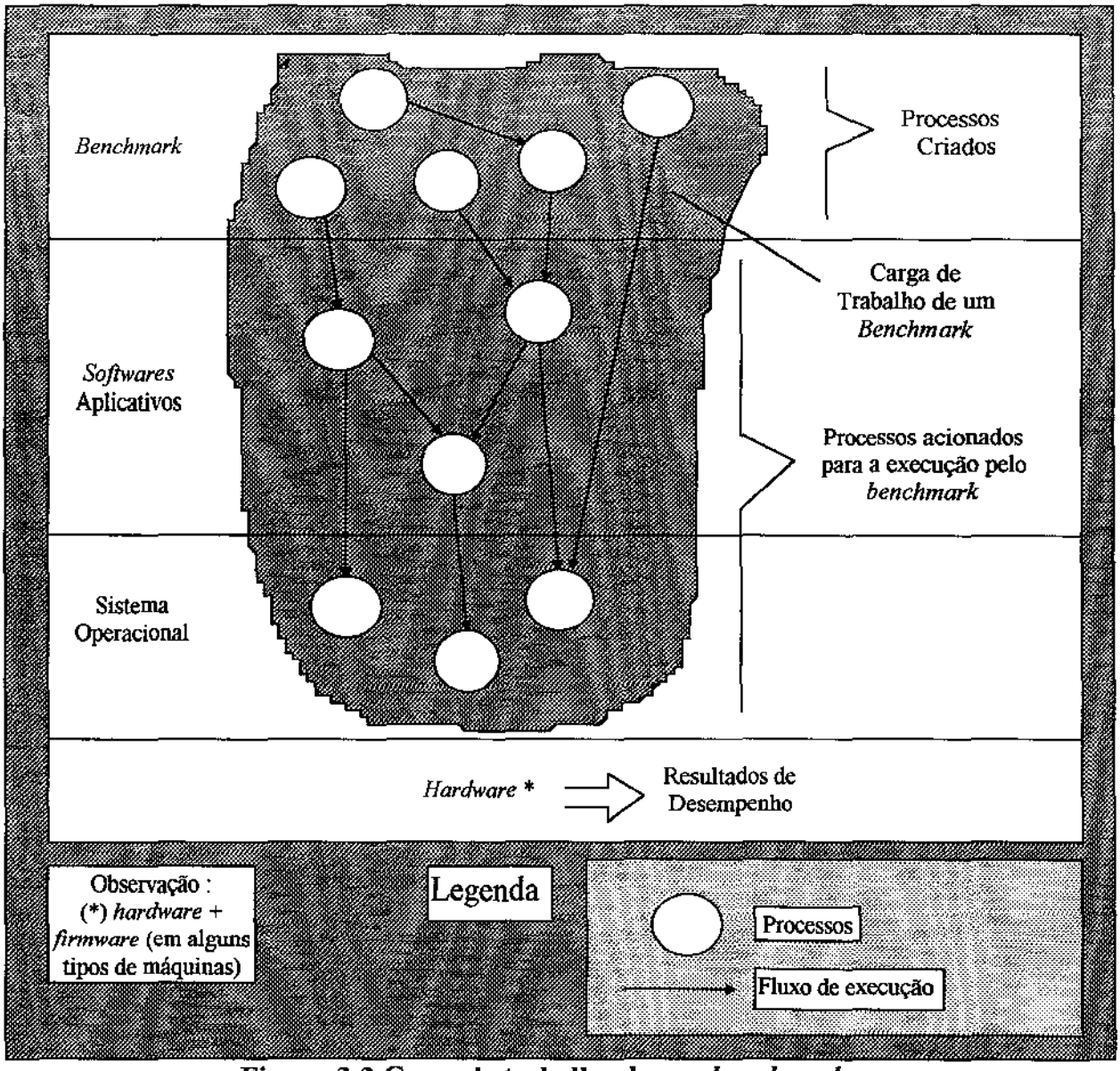


Figura 3.3 Carga de trabalho de um *benchmark*.

Vale novamente ressaltar que tanto os processos criados pelo *benchmark* quanto os processos acionados por ele, os quais são provenientes dos *softwares* aplicativos e do sistema operacional, fazem parte da carga de trabalho do *benchmark*. Em adição, vários processos independentes (que não são criados ou acionados para execução pelo *benchmark*) compartilham os recursos computacionais com os processos pertencentes à carga de trabalho do *benchmark*, influindo diretamente na degradação do desempenho do sistema computacional. A figura 3.4 mostra esta situação. Nesta figura, a carga de trabalho identificada com o rótulo A refere-se à carga de trabalho do *benchmark*, enquanto a carga de trabalho identificada com o rótulo B é denominada carga de trabalho externa ao *benchmark*, a qual é gerada pelos processos independentes.

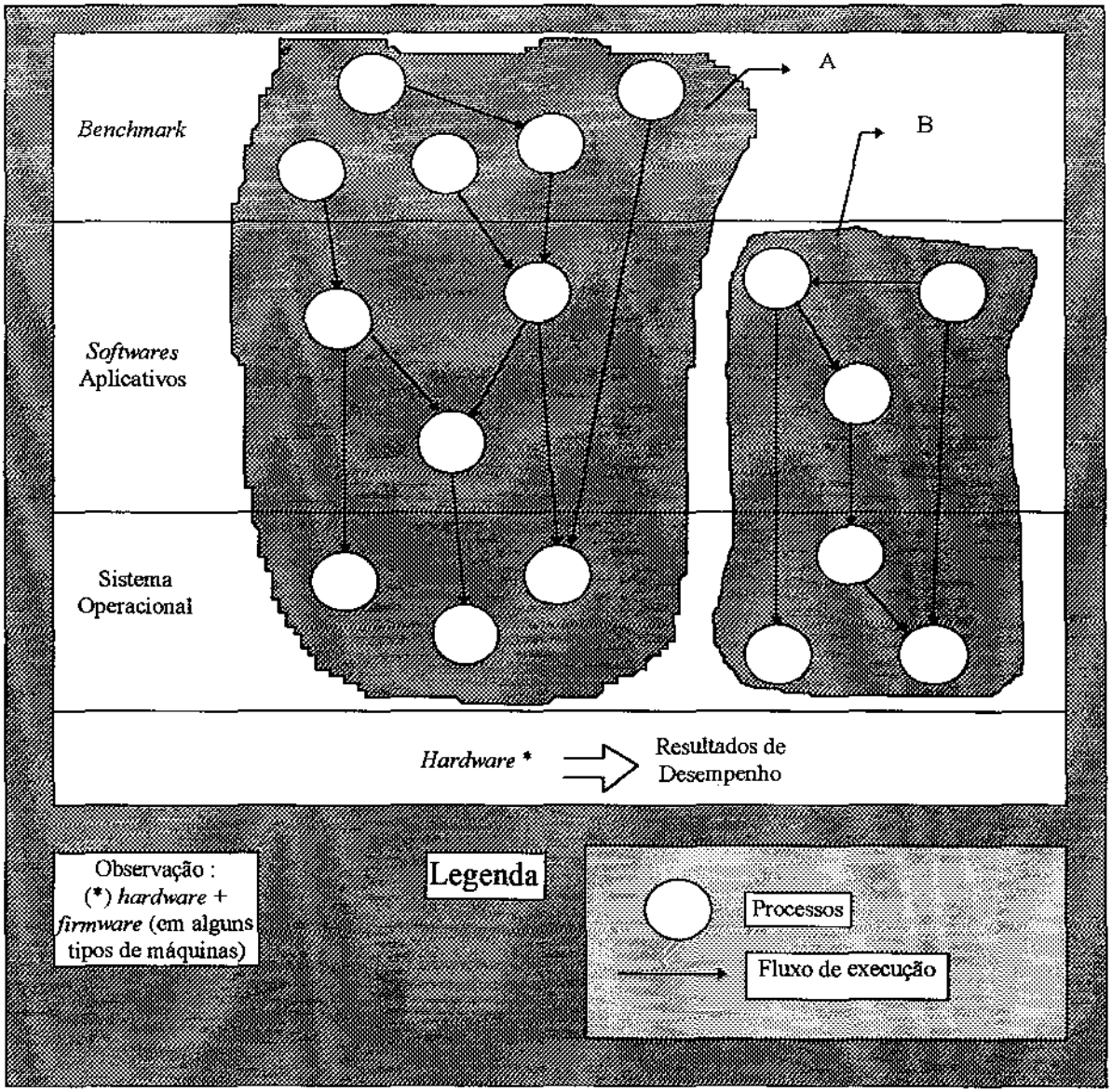


Figura 3.4 Carga de trabalho externa e carga de trabalho do *benchmark*.

Tecnicamente, o conceito de transação, comumente encontrado na terminologia de banco de dados, é de certa forma equivalente ao conceito de processo. Uma transação logicamente consiste em um conjunto de operações de escrita e leitura em um banco de dados, além de operações de processamento. A implementação de transações é efetuada através de linguagem de alto nível, geralmente com o uso de uma linguagem de 4ª geração, fornecida pelo *software* SGBD, em adição a uma linguagem convencional, tal como a linguagem C. Como transações são implementadas através de programas e estes programas ao executarem tornam-se processos, pode-se dizer que uma transação constitui-se de um tipo específico de processo, o qual tem por objetivo a leitura e a escrita de dados em vários arquivos pertencentes a um determinado banco de dados.

Este texto tratará os conceitos de processo e transação como equivalentes. Assim, a carga de trabalho total pode ser redefinida como sendo a união da sobrecarga de execução das transações ativas no sistema computacional, as quais ou são ligadas ao *benchmark* ou são tidas como independentes. [Per90] também define a carga de trabalho em função do conceito de transação. Para este autor, um *benchmark* de banco de dados possui um conjunto de transações, ou carga de trabalho como é comumente denominada, e estas são executadas sobre um banco de dados conhecido. Quando a carga de trabalho de um *benchmark* é formada por vários tipos distintos de transações, para cada uma destas deve-se caracterizar rigidamente: a freqüência relativa de execução (distribuição das transações), ordem de execução e tipo de "*bufferização*" (se o *buffer* deve ou não ser "limpo" após a execução de cada transação).

Um aspecto muito importante a ser identificado relaciona-se com o que quer-se analisar, ou seja, com a identificação do que constitui o sistema de análise. Genericamente, o sistema de análise é formado por um subconjunto da carga de trabalho presente no sistema computacional durante a realização dos testes de desempenho, no que tange o *software*, e opcionalmente pelos recursos computacionais presentes para suportar esta carga de trabalho, caso o *hardware* seja objeto de análise. Particularmente, o sistema de análise pode englobar o sistema computacional inteiro (ambiente de aplicação), ou apenas parte deste, tais como um nível do sistema computacional, alguns processos pertencentes a um nível específico ou alguns processos pertencentes a vários níveis, entre outros.

A identificação do sistema de análise é essencial pois cada nível de *software* de um sistema computacional multinível exerce influência específica no desempenho, através de sobrecargas de execução distintas, assim como o *hardware* provê recursos distintos, tais como capacidade de processamento, capacidade de armazenamento, capacidade de transferência de dados para dispositivos de entrada/saída, entre outros. Um exemplo de sistema de análise é mostrado na figura 3.5, onde este é formado apenas por um *software* básico específico, apesar da carga de trabalho total ser formada tanto por processos oriundos da carga de trabalho do *benchmark* quanto por processos oriundos da carga de trabalho externa ao *benchmark*. Os rótulos A e B têm os mesmos significados que os da figura 3.4.

Após definidas as transações do *benchmark* e os dados (seção 3.4) que comporão o *benchmark*, deve-se definir a abrangência de cada transação em relação aos dados. Assim, por exemplo, uma dada transação pode acessar exclusivamente um arquivo X, selecionando dados de acordo com o valor do campo *c1* de cada registro deste arquivo. Outra transação também pode acessar exclusivamente o arquivo X, mas selecionando dados de acordo como o valor do campo *c2* de cada registro deste arquivo. Desta maneira, pode-se determinar a seletividade de cada transação (em relação aos dados).

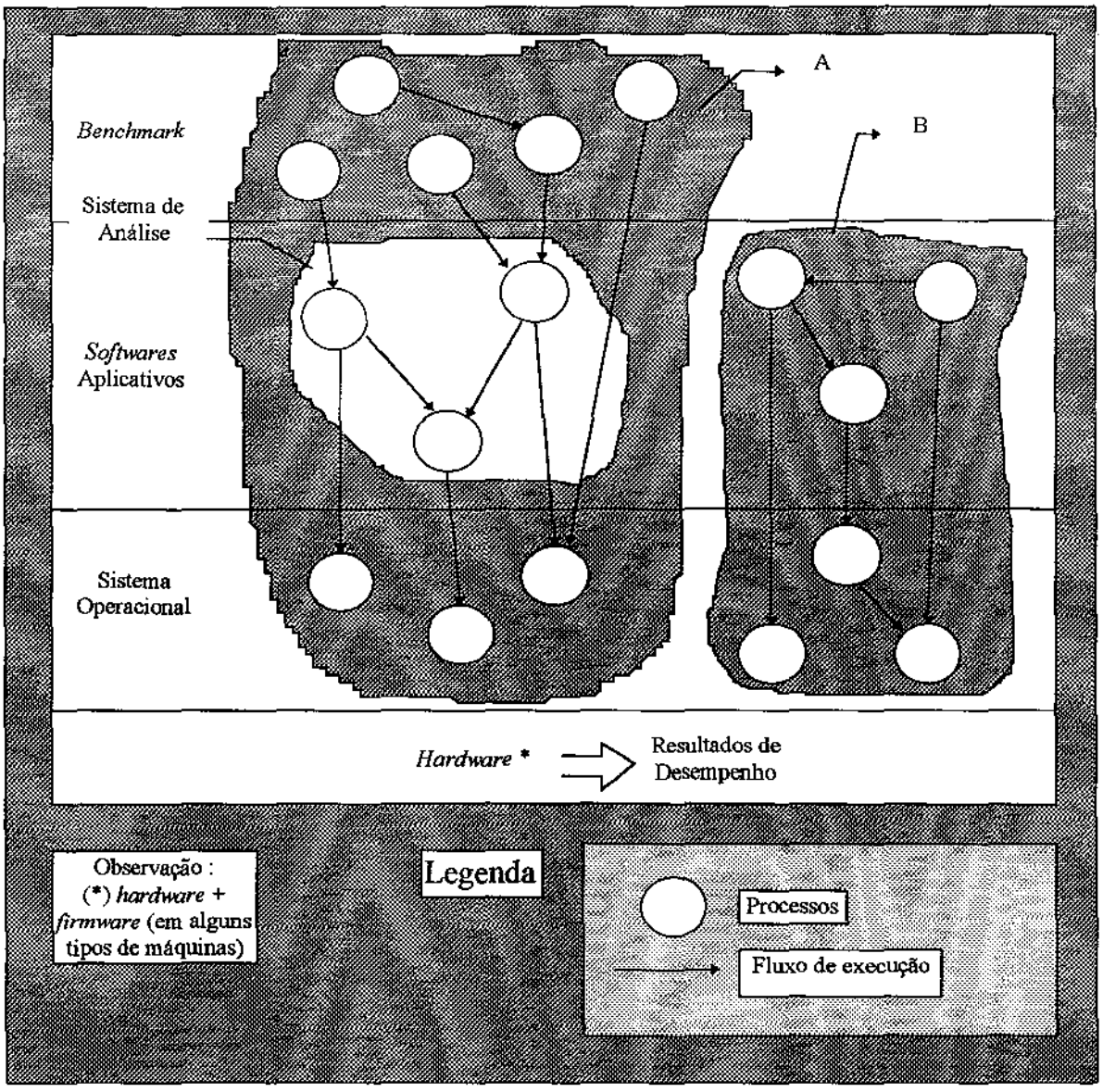


Figura 3.5 Exemplo de um sistema de análise.

3.4 Dados em um *benchmark*

Assim como cargas de trabalho distintas influenciam de modo particular o desempenho, composições de dados distintas degradam o desempenho diferentemente. A caracterização dos dados que comporão o *benchmark* deve ser feita de maneira não ambígua, de modo a inserir uma constante de degradação única no desempenho de todos os sistemas a serem analisados. A indefinição, por exemplo, do tipo de dado de um arquivo, pode provocar resultados imprecisos de comparação de desempenho, uma vez que tipos distintos de dados possuem, entre outras coisas, quantidade de *bytes* distintas. Deste modo, a execução de uma transação em um arquivo com dados do tipo *integer* (2 *bytes*) tende a dar melhores resultados de desempenho que a sua execução em um arquivo com dados do tipo *long integer* (4 *bytes*), desde que nestes últimos faz-se necessária a leitura/gravação do dobro de *bytes*[3]. Deste modo, alguns aspectos relacionados aos dados devem ser fixados para que os processos sempre atuem sobre dados com as mesmas características, possibilitando desta maneira a comparação de resultados de desempenho.

A caracterização de dados em *benchmarks* pode ser feita de dois modos: através de dados sintéticos e reais.

O primeiro, dados sintéticos, consiste em dados gerados artificialmente, possibilitando a inserção de características especiais nos dados de modo a explorar deficiências encontradas ou pressentidas nos sistemas a serem analisados. Entretanto, sua caracterização torna-se mais complexa, uma vez que para cada execução do *benchmark* os dados devem ser os mesmos (ou equivalentes), independente da configuração do sistema computacional sendo analisado. Para tanto, devem ser descritas características tais como o modelo conceitual do banco de dados sintético que representa a aplicação alvo do *benchmark* (com o propósito de facilitar o entendimento, principalmente do inter-relacionamento entre os dados), composição dos arquivos (tipos de dados, quantidade, estrutura do banco de dados), e a forma de geração dos dados para os diversos arquivos (distribuição e controle de seletividade).

Em especial, a geração de dados sintéticos pode optar por uma distribuição uniforme ou não uniforme dos dados. A distribuição uniforme geralmente é efetuada com base em uma função randômica uniforme, na qual os dados são gerados aleatoriamente, sendo a freqüência para cada valor distinto constante ao longo do tempo. Entretanto, distribuições uniforme de dados inexistem em aplicações reais, sendo este fato primeiramente observado por G.K. Zipf, o qual relatou o comportamento de distribuições assimétricas. Observando este tipo de comportamento, por exemplo, com a utilização de palavras em populações de cidades, nomes de pessoas, etc, ele formulou uma lei que descreve este comportamento.

[3] desprezando-se (mantendo constante) outros fatores relacionados aos dados, tais como quantidade de registros, organização dos arquivos, mecanismos de indexação e distribuição dos dados, entre outros.

A lei de *Zipf*, como passou a ser chamada, pode ser enunciada da seguinte forma:

> "O *ranking*[4] das palavras em um texto é inversamente proporcional à sua freqüência de ocorrência".

O elemento mais freqüente recebe a posição do topo do *ranking*, ou seja, o valor numérico 1, e o menos freqüente recebe a última posição. Pode-se, ainda, expressar a lei de *Zipf* matematicamente, onde o valor w_i é a freqüência do i-ésimo elemento (equação 3.1). Em uma distribuição do tipo *Zipf*, z é chamado de fator de decaimento. Dependendo deste valor, a distribuição pode se tornar mais ou menos assimétrica. Pode-se verificar que quando z = 0 a distribuição é uniforme e quando z = 1 a distribuição é *Zipf*. Assim, a geração de dados sintéticos pode ser efetuada com base na lei de *Zipf*, sendo a freqüência de cada elemento e_i distinto obtida da equação 3.1 por isolamento de w_i. Em [Per90] são encontrados vários algoritmos de geração de dados sintéticos, os quais geram desde distribuições uniforme até distribuições do tipo *Zipf* (z = 1), para vários tipos de dados convencionais.

$$\text{ranki}^{z} \times \text{wi} = \frac{1}{\sum_{k=1}^{n} \frac{1}{(k)^{z}}}, \text{ onde } \sum_{i=1}^{n} \text{wi} = 1$$

Equação 3.1 Lei de *Zipf*.

A utilização de distribuições assimétricas é muito importante para o projeto de otimizadores de consultas de SGBDs, pois a hipótese de comportamento uniforme pode levar a uma estimativa incorreta da seletividade de um predicado. Em geral, quando a seletividade real de uma amostra é bastante pequena, mas os valores são distribuídos não uniformemente, a seletividade estimada (supondo um comportamento uniforme) tende a ser superestimada, o que na prática pode resultar na escolha de uma busca seqüencial, quando neste caso o uso de um índice seria mais apropriado. De forma inversa, quando a seletividade real de um segmento de valores é elevada, mas os valores possuem uma distribuição muito assimétrica, a seletividade estimada, da mesma maneira que a anterior, tende a ser subestimada, o que pode resultar na escolha de um índice ao invés de uma busca seqüencial, a qual seria mais conveniente neste caso. A determinação incorreta da seletividade, por parte do otimizador de consultas, prejudica em muito a execução de transações em SGBDs.

[4] ordenação de acordo com a freqüência de ocorrência de cada elemento distinto.

Durante a geração dos dados, deve-se também procurar obter, além de uma dada distribuição (uniforme, não uniforme), uma seletividade controlada dos dados. O controle da seletividade dos dados, isto é, a possibilidade de se recuperar apenas uma certa percentagem dos dados, como exemplo 1%, 5% ou 30%, é muito importante, dado que uma transação pode ter um desempenho não proporcional à quantidade de dados que ela acessa (Logicamente, uma transação do tipo *scan* – busca seqüencial – pode ter seu desempenho alterado em função do número de registros lidos. Entretanto, esta diferença no desempenho segue um crescimento proporcional ao número total de registros. Assim sendo, para um arquivo com 100 registros, se a leitura de 10 registros gastar 1 segundo, a leitura de 50 registros deve gastar aproximadamente 5 segundos, e assim por diante. Quando a leitura dos 50 registros gastar, por exemplo, 30 segundos, verifica-se um desempenho não proporcional à quantidade de dados). O *benchmark* de *Wisconsin* [Dew85], largamente utilizado para análise de desempenho de SGBDs relacionais, distribui os dados em diversas tabelas, onde cada uma destas tabelas proporciona uma seletividade controlada. Neste *benchmark*, a seletividade é obtida diretamente da distribuição uniforme dos dados. Entretanto, pode-se também obter um certo grau de seletividade controlada a partir de uma distribuição não uniforme, tal como *Zipf* (a partir dos *ranks* gerados).

A segunda forma de caracterização de dados em um *benchmark*, dados reais, considera um conjunto de dados existente, encontrado em uma aplicação referente ao contexto sendo analisado. Esta abordagem muitas vezes não consegue gerar configurações de dados que testem gargalos, sendo também difícil obter um critério controlado de seletividade. Entretanto, pela sua natureza real, consegue-se obter resultados de desempenho mais associados, teoricamente, às aplicações deste tipo (desde que os dados sejam oriundos de uma aplicação real representativa, a qual misture um conjunto adequado de características presentes nas diversas aplicações inerentes ao contexto sendo analisado). Além disto, a descrição destes dados torna-se mais simples, uma vez que esta é efetuada sobre o conteúdo dos dados, os quais implicitamente se relacionam através dos diversos arquivos do banco de dados, e não sobre como organizar e gerar estes dados. A descrição destes dados, inclusive, muitas vezes tem por finalidade apenas destacar a representatividade do *benchmark*.

Para ambos dados sintéticos e reais, os principais tipos de dados presentes em *benchmarks* voltados à análise de desempenho de SGBDs convencionais são basicamente valores numéricos e alfanuméricos (dados convencionais). A representação destes tipos de dados nos SGBDs não requer nenhuma estratégia específica dos sistemas de banco de dados, uma vez que fazem parte do conjunto básico fornecido pelas linguagens de programação nas quais SGBDs são escritos: *integer*, *long integer*, *float*, *char*, *string*, *date* e *enum*, dentre outros. A seguir, é efetuada uma breve descrição de alguns tipos de dados convencionais.

- **tipo *integer*:** corresponde ao armazenamento de números inteiros que podem variar entre 0 e 65.535, sem sinal. A quantidade de *bytes* necessária para o armazenamento de dados deste tipo é de 2 *bytes*.

- **tipo *long integer*:** corresponde ao armazenamento de números inteiros que podem variar entre 0 e 4.294.967.295, sem sinal. A quantidade de *bytes* necessária para o armazenamento de dados deste tipo é de 4 *bytes*.

- **tipo *float*:** corresponde ao armazenamento de números reais que podem variar entre 3.4E-38 e 3.4E+38, com seis dígitos de precisão. A quantidade de *bytes* necessária para o armazenamento de dados deste tipo é de 4 *bytes*.

- **tipos *char* e *string*:** um dado do tipo *char* corresponde ao armazenamento de um caractere, o qual ocupa 1 *byte*, em geral. O tipo *string*, uma generalização do tipo *char*, consiste no armazenamento de n caracteres, representados por *string*[n], ocupando n *bytes*.

- **tipo *date*:** corresponde ao armazenamento de uma data completa, a qual deve conter dia, mês e ano, como exemplo, 12/06/1991 ou 05/11/1543. Supondo o uso de números para o armazenamento deste tipo, uma vez que isto facilita a manipulação numérica de datas, faz-se necessário 2 *bytes* para o seu armazenamento.

- **tipo *enum*:** corresponde a uma enumeração de valores inteiros. Cada valor inteiro enumerado representa uma característica distinta, tal como 1-pecuária, 2-indústria, etc. A quantidade de *bytes* requerida para o armazenamento de um dado deste tipo varia de acordo com a quantidade de elementos da enumeração. Entretanto, a quantidade de elementos distintos nas enumerações utilizadas para a caracterização dos dados geralmente é limitada, permitindo o seu armazenamento em 2 *bytes*, caso o tipo enumerado seja formado a partir do tipo *integer*.

3.5 Tipos de medidas

Outro conceito importante a ser caracterizado refere-se aos tipos de medidas fornecidas. [Fer78] cita três classes de medidas de desempenho: medidas que analisam a produtividade, medidas que detectam a utilização e medidas que captam o tempo de resposta de transações individuais.

A medida de produtividade, ou *throughput* como é comumente denominada na literatura, consiste no número de operações que são efetuadas em um determinado período de tempo. A caracterização do tipo das operações é que diferencia as distintas medidas de *throughput* existentes. O tipo das operações, por sua vez, está intimamente relacionado ao sistema de análise em questão, sendo estas de granularidade baixa, como operações de ponto flutuante, quando o sistema de análise é formado apenas por um recurso computacional específico, tal como um processador, ou de granularidade alta, como transações, quando o sistema de análise é mais abrangente, tal como um sistema *OLTP*

(*On-line Transaction Processing*). Medidas de produtividade são ditas macroscópicas, uma vez que elas mostram um panorama do desempenho em nível global, sem se preocupar com a quantificação unitária da degradação de desempenho proporcionada por cada transação ativa no sistema computacional. Portanto, estas medidas são indicadas para analisar o desempenho de sistemas de análise compostos pelo sistema computacional total.

As medidas *mips* (milhões de instruções por segundo) e *mflops* (milhões de operações de ponto flutuante por segundo) são as medidas de produtividade mais populares, devido em parte à simplicidade de suas definições. No entanto, por serem medidas ditas de baixo nível, mais especificamente para análise de desempenho de processadores, estas não podem ser utilizadas em sistemas de análise mais abrangentes. Isto se deve ao fato de que para determinados sistemas de análise a influência gerada pelo processador no desempenho global é parcial, e muitas vezes reduzida, sendo esta influência dependente de vários outros recursos computacionais, principalmente de dispositivos de entrada/saída. Assim, pode-se dizer que para estes tipos de sistema de análise as medidas *mips* e *mflops* não conseguem capturar a influência proporcionada pela diferença de qualidade de *software*, sendo esta diferença detectada apenas quando a operação é caracterizada em um nível de granularidade alto, ou seja, em nível de rotinas ou transações. [Ano+85, Gra91] descrevem a limitação do uso destas medidas para análise de desempenho de sistemas *OLTP*, dizendo que estas não medem a influência gerada pela arquitetura de entrada/saída, pela rede de comunicação e pelo *softwares* presentes nestes sistemas.

A medida *mips*, em particular, é altamente criticada, mesmo quando utilizada para análise de desempenho de processadores. Críticos desta medida argumentam que esta ignora aspectos relacionados às diferenças entre as arquiteturas *RISC* (*Reduced Instruction Set Computer*) e *CISC* (*Complex Instruction Set Computer*) onde sabe-se muito bem que uma instrução *CISC* pode corresponder a várias instruções *RISC*. Assim, um resultado de Rx *mips* medido em um sistema de análise com arquitetura *RISC* é inferior ao mesmo resultado medido em um sistema de análise com arquitetura *CISC*. Entretanto, a medida *mips* não consegue identificar este fato. Para que a medida *mips* possa ser usada em sistemas computacionais distintos, independente do tipo de arquitetura de processador utilizado, esta deve ser normalizada. A normalização consiste em um multiplicador que estabelece a equivalência entre o número de instruções *RISC* e o número de instruções *CISC*, tomando como base a arquitetura *CISC* (instrução *CISC* = 1). Como uma instrução *CISC* geralmente corresponde a várias instruções *RISC*, o multiplicador para sistemas de análise com a arquitetura *RISC* será inferior a 1. Isto é mostrado na tabela 3.1. Nesta tabela vê-se claramente a diferença entre as medidas *mips* e *mips* normalizada. Os sistemas de análise *DEC VAX 8800* e *Sun Sparc* apresentam um resultado de desempenho semelhante quando comparados através da medida *mips*. Entretanto, ao se aplicar o multiplicador no sistema de análise Sun Sparc, o qual possui arquitetura *RISC*, o desempenho deste mostra-se sensivelmente inferior.

Processador	*Speed (Mhz)*	*Mips*	*Multiplicador*	*Mips ajustado*
DEC VAX 780	5,00	0,47	1,00	0,47
DEC VAX 8800	22,00	11,00	1,00	11,00
Intel 80386	24,00	5,30	1,00	5,30
MIPS 3000	25,00	20,00	0,80	16,00
Sun Sparc	16,70	11,90	0,50	5,95

Tabela 3.1 *Mips* x *mips* ajustado [Gra91].

A medida *mips* recebe outros tipos de críticas, mesmo quando normalizada. A determinação do multiplicador, por exemplo, é altamente subjetiva e imprecisa, sendo motivo de debates constantes entre os fabricantes de computadores. Já [Gra91] cita que a medida *mips* além de ser muito simples, não é escalável, ou seja, o seu uso em sistemas de análise complexos não é bem definido, como exemplo em sistemas de análise com arquitetura multiprocessada. Nestes tipos de sistemas a medida *mips* se confunde, uma vez que não é claro se um sistema de análise com 1000 processadores e com desempenho de 1000 *mips* (1 *mips*/processador) é equivalente a um sistema de análise com um único processador com o mesmo desempenho de 1000 *mips*. Além disto, a medida *mips* é altamente influenciada pelo tipo de compilador utilizado na geração de código dos processos que compõem a carga de trabalho total. Dependendo do compilador utilizado, pode-se ter uma baixa ou uma alta sobrecarga de execução, fazendo com que o desempenho apresentado pelo processador varie por um fator multiplicativo de até três. Qualquer medida baseada em *mips*, tal como custo/*mips*, apresentará os mesmos problemas inerentes a ela.

Outro exemplo de medidas de produtividade são as medidas TPS e TPM encontradas nos *benchmarks TPC-A*/B [Tra89a, Tra89b] e *TPC-C* [Tra92], respectivamente. A medida TPS significa número de transações efetuadas por segundo. Analogamente, TPM significa número de transações efetuadas por minuto. A caracterização precisa do conjunto de transações é muito importante, uma vez que tipos diferentes de transações influenciam de modo diferente no desempenho. Assim, antes de comparar uma medida como TPS, divulgada em *benchmarks* distintos, deve-se comparar a caracterização da carga de trabalho total de cada um dos *benchmarks*. Caso as cargas de trabalho presentes nestes *benchmarks* sejam diferentes, não é possível a comparação dos resultados gerados. A simples conversão, por exemplo, da medida TPM encontrada no *benchmark TPC-C* para TPS, através da fórmula TPS = TPM/60, não permite a comparação com a medida TPS encontrada no *benchmark TPC-A*, uma vez que a carga de trabalho destes *benchmarks* difere em relação aos tipos de transações e freqüência de execução.

A utilização de recursos computacionais, quando medida, serve para detectar possíveis fontes de gargalos decorrentes de recursos computacionais específicos. A utilização pode ser reportada de duas maneiras: através de quantificação absoluta ou através de quantificação relativa (também chamada de taxa de utilização). Medidas de tempo e de capacidade são utilizadas para a caracterização da quantificação absoluta, enquanto a caracterização da quantificação relativa é dada em percentagem (%), a qual indica a

freqüência relativa de utilização de um dado recurso computacional. Exemplos típicos de recursos computacionais onde a utilização é medida são:

- **unidade central de processamento (UCP):** a quantificação absoluta da utilização é efetuada por alguma medida de tempo, tal como nanosegundo. Assim, pode-se, por exemplo, verificar o tempo despendido pela UCP para realizar as computações requeridas por uma dada transação e a partir desta estimativa de tempo decidir se é necessário a atualização para um modelo de UCP mais potente. Vale lembrar que este tempo será apenas parte do tempo total gasto para executar a transação, o qual inclui também o tempo de entrada e saída de dados. Desta forma, a atualização da UCP pode ou não influir significativamente no desempenho total do sistema computacional. Na prática, porém, existe um aumento de desempenho com a atualização de UCP devido à importância das funções desempenhadas por este componente dentro da estrutura global de um sistema computacional. Já a taxa de utilização (x%) é utilizada para indicar o percentual de ociosidade (100% - x%) deste recurso computacional. Uma ociosidade elevada, a qual reflete um sistema computacional subutilizado em termos de UCP, pode ser gerada por três fatores principais. O primeiro está diretamente relacionado às degradações de desempenho indiretas, ou seja, degradações proporcionadas pela lentidão relativa de outros recursos computacionais, principalmente dispositivos de entrada/saída. Uma solução para este tipo de problema seria a troca dos recursos computacionais considerados "lentos" por outros considerados mais "velozes". Um segundo fator está relacionado à falta de processos ativos suficientes para exercitarem o processador durante todo o tempo. Sistemas de análise com esta característica podem aumentar a carga de trabalho ativa sem que haja qualquer degradação de desempenho adicional. Um terceiro fator está relacionado ao sistema operacional, o qual pode escalonar a UCP ineficientemente, fazendo com que esta fique ociosa quando poderia estar executando um outro processo, como exemplo, em requisições de entrada/saída. A troca de sistema operacional poderia solucionar este tipo de problema. Inversamente, uma taxa de utilização elevada pode indicar a necessidade de substituição da UCP.

- **memória principal e secundária:** a quantificação absoluta da utilização, para estes dois tipos de recursos computacionais, é efetuada pela medida *byte*, múltiplos e submúltiplos. Esta medida ajuda a identificar a real necessidade de armazenamento para o sistema de análise em questão, assim como ajuda a desenvolver modelos que conseguem predizer a quantidade de memória mínima que um dado sistema de análise precisaria ter, em função de certos parâmetros inerentes a este sistema, para que o desempenho não se degrade significativamente. Uma vez que sistemas de análise similares necessitam de quantidade de armazenamento diferentes para efetuar a mesma tarefa, o sistema de análise que necessitar da menor quantidade de memória pode ser considerado o melhor em relação ao tópico armazenamento de dados. Um sistema de análise ideal deveria conciliar bons resultados com outras medidas de desempenho a bons resultados em termos de armazenamento. Assim, o custo para o armazenamento de dados seria minimizado e o tempo individual e global de execução de transações seria aceitável. Quando a medida de utilização de memória é reportada percentualmente, esta pode ser usada para determinar possíveis restrições críticas de

armazenamento (taxa de utilização > 95%). Um outro uso é a determinação da subutilização do meio de armazenamento, permitindo reconfigurações no sistema computacional que possibilitem aumentar o desempenho, como exemplo, a expansão da área de memória virtual utilizada pelo sistema operacional ou a expansão da área de memória *cache* de disco presente na memória principal. Em relação ao armazenamento de dados em memória secundária, geralmente mede-se também o número de entrada / saída lógicas e físicas efetuadas durante a realização de uma dada transação. Isto ajuda principalmente na reconfiguração da área de *buffer*. Como a transmissão de dados para dispositivos de armazenamento secundário degrada efetivamente o desempenho, desenvolvedores de *software* podem utilizar um *benchmark* durante os estágios iniciais de desenvolvimento para determinar o número de entrada/saída lógicas e físicas efetuadas e assim minimizá-las em novas versões com o intuito de melhorar o desempenho.

Uma outra medida interessante é a taxa de utilização global, que indica o percentual de utilização do sistema computacional total. Esta taxa de utilização é calculada de acordo com um conjunto de regras bem definidas, as quais relacionam para cada recurso computacional um peso específico. A implementação destas regras é efetuada através de rotinas específicas providas pelo próprio *benchmark*, que por sua vez utilizam primitivas fornecidas pelo sistema operacional alvo ou diretamente o *hardware* para serem implementadas. No entanto, independentemente do tipo de sistema operacional ou *hardware* em questão, as regras devem ser implementadas de acordo como foram definidas em alto nível. O uso desta medida para efeitos comparativos é reduzido, sendo mais utilizada para indicar o espaço de medição de uma medida de produtividade, ou seja, o intervalo na qual uma dada medida de produtividade deve ser reportada. Um exemplo deste uso é descrito a seguir. Suponha que um determinado sistema de análise A apresente como resultado de desempenho 5 TPS com uma taxa de utilização global de 3%. Agora considere um sistema de análise B, que sobre a mesma carga de trabalho obtém o desempenho de 10 TPS, porém com uma taxa de utilização global de 98%. Vê-se claramente neste exemplo que o espaço de medição para o sistema de análise A foi inadequado. Este sistema de análise poderia obter um desempenho muito superior desde que o espaço de medição fosse dependente da taxa de utilização global do sistema. Já o sistema de análise B não conseguiria aumentar significativamente o seu desempenho, uma vez que este utilizou de todos os seus esforços (98% de utilização global) para obter o resultado de 10 TPS. Uma comparação direta dos resultados apresentados para os sistemas de análise A e B, sem a observação anterior sobre a inadequação do espaço de medição para o sistema de análise A, poderia levar a conclusões equivocadas. Assim, a taxa de utilização global, quando utilizada para indicar o espaço de medição de uma dada medida de produtividade, atua como um mecanismo protetor contra possíveis distorções decorrentes da subutilização do sistema computacional. O espaço de medição de uma medida de produtividade constitui-se então de um intervalo de tempo onde a taxa de utilização global assume somente valores apropriados, ou seja, $x1 < \text{taxa de utilização} < x2$ para um certo Δt.

A quantificação unitária da degradação de desempenho proporcionada pelas transações individuais são medidas através de medidas microscópicas, as quais são medidas de tempo simples de resposta, como hora, minuto e segundo, além de seus múltiplos e submúltiplos. Estas medidas podem ser obtidas externamente através de algum mecanismo de contagem de tempo, tal como um cronômetro, ou internamente através de rotinas que acessam os dados do relógio presente no sistema computacional. Em geral, efetua-se várias vezes a medição do tempo de resposta de uma dada transação e depois calcula-se a média aritmética destes tempos, evitando-se assim a generalização de erros decorrentes de uma única medição. Uma importante caracterização é a definição do espaço de medição de medidas microscópicas, ou seja, a definição do início e do fim do momento da medição. Freqüentemente, este espaço de medição envolve a visualização dos resultados, onde o início da medição é caracterizado como o momento em que o primeiro *byte* é recebido do dispositivo de entrada e o término da medição acontece somente quando o último *byte* dos resultados gerados são mostrados no dispositivo de saída. Espaços de medição deste tipo são denominados espaço de medição externo, uma vez que o resultado de desempenho reportado inclui tanto o tempo necessário para processamento quanto o tempo necessário para a entrada e saída de dados. Em alguns casos, onde o usuário necessita fornecer dados para a transação, o tempo gasto para este fim deve ser incluído, sendo medido empiricamente ou adicionado ao tempo total de execução da transação de acordo com estimativas estatísticas. Espaços de medição externo, portanto, são usados com transações *on-line*. Já o espaço de medição interno envolve somente o tempo necessário para processamento, não sendo medidos os tempos necessários para a entrada e saída de dados, sendo usado somente com transações *batch*. Similarmente às medidas de produtividade, medidas de tempo de resposta são indicadas para analisar o desempenho de sistemas de análise compostos pelo sistema computacional total, porém o desempenho reportado por estes dois tipos de medidas são considerados diferentes.

Medidas de produtividade e medidas microscópicas de tempo de resposta reportam o desempenho através de ângulos de observação distintos. A medida de produtividade, em sistemas de análise abrangentes, por exemplo, reporta o número de transações efetuadas em um certo período de tempo. No entanto, esta não se preocupa em medir o tempo de resposta individual de cada transação ativa no sistema de análise. Algumas vezes, apesar do desempenho de produtividade se mostrar elevado, este sacrifica o tempo de resposta de algumas transações individuais. Deste modo, um determinado sistema computacional pode optar por ter um alto *throughput*, e em contrapartida por ter um baixo tempo de resposta para alguns tipos de transações. Isto é identificado apenas através de medidas microscópicas, as quais são aplicadas individualmente aos tipos de transações consideradas desprivilegiadas. A figura 3.6 mostra esta situação, para um ambiente de processamento *batch*. A situação 1 descreve um sistema computacional simples onde as transações são executadas na ordem em que chegam. Neste exemplo, a transação A é executada imediatamente e consome portanto 10 minutos para ser executada. Com isto, o tempo de resposta de 10 minutos desta transação é o mínimo possível. Entretanto, nestes 10 minutos o *throughput* é de apenas 6 transações/hora. Já a situação 2 descreve um sistema computacional onde as transações são executadas de acordo com o tempo estimado de execução, e com isto as transações que possuem o menor tempo estimado são executadas primeiramente. Aqui, as transações C, E, H e D são executadas anteriormente à transação

A, e o *throughput* aumenta significativamente para 24 transações/hora, no mesmo intervalo de tempo. Por outro lado, o tempo de resposta para transações deste tipo será dilatado e portanto maior que 10 minutos, 30 minutos neste exemplo – o triplo.

Com o intuito de juntar os ângulos de observações distintos gerados por medidas de produtividade e tempo de resposta, o espaço de medição de uma medida de produtividade pode ser definido com base em uma medida de tempo de resposta simples. [Ano+85] define a medida de produtividade TPS, no *benchmark* de débito e crédito, como sendo o número máximo de transações efetuados por segundo antes que o sistema de análise fique saturado, ou seja, que o tempo de resposta de transações individuais exceda 1 segundo. Assim, o espaço de medição para esta medida de produtividade fica condicionado ao intervalo em que o tempo de resposta individual das transações não é comprometedor, neste caso $0 < \tau \leq 1$, onde τ é o tempo medido em segundos.

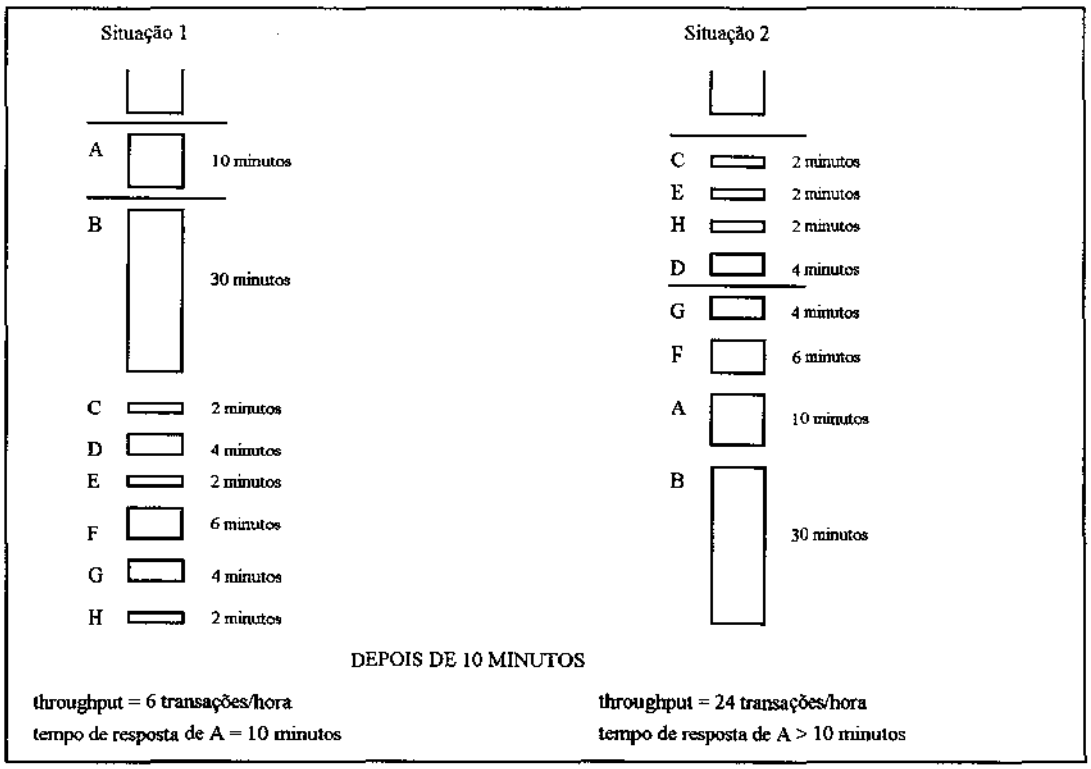


Figura 3.6 *Throughput* x tempo de resposta.

Benchmarks, porém, são criados e usados freqüentemente para responder simples questões, tais como "Qual sistema computacional eu devo comprar ?"ou "Qual SIG eu devo escolher para a minha empresa ?". A resposta óbvia a estas questões deveria ser algo como "O sistema que faz o trabalho que você quer da melhor maneira possível e ao menor custo". As medidas de desempenho já apresentadas, medidas de produtividade, utilização e tempo de resposta, não respondem completamente a estas questões, conseguindo somente determinar se um dado sistema possui a capacidade mínima, em termos de desempenho, para suportar a aplicação alvo requerida. Desta forma, sistemas que não atingirem esta

capacidade mínima certamente não estarão incluídos na resposta a estas questões. Além disto, estas medidas, apesar de complementares, pois cada uma reporta um tipo particular de desempenho e com isto obtém-se uma visão genérica do desempenho do sistema de análise em questão, não capturam um importante fator presente no processo de escolha de sistemas, o custo.

A computação do custo final envolvido na implantação de um sistema é difícil de ser efetuada, pois envolve custos provenientes de diversas áreas, tais como custos de análise e projeto, custos de programação, custos de operação, custos de manutenção, custos de treinamento e aprendizagem, além dos custos de *hardware* e *software* envolvidos. Muitos destes custos são difíceis de se quantificar, quando não impossíveis. Apesar da alta complexidade envolvida na computação do custo final, este deve ser de alguma forma reportado para ser usado comparativamente, uma vez que o preço proibitivo de alguns sistemas o torna um fator decisivo no processo de escolha. Na prática, os custos reportados são os custos associados ao *hardware*, ao *software* e à manutenção para um período de 5 a 6 anos[5], não sendo medidos qualquer outro custo envolvido [Ano+85, Tra89a, Tra89b].

Esta simplificação não invalida totalmente o processo de medição do custo, pois os fatores medidos correspondem a uma boa parcela do custo final. Além disto, devido à impossibilidade de se medir alguns tipos de custo, uma comparação simplificada torna-se importante, particularmente melhor do que nada. Assim, os custos reportados por um *benchmark* devem ser vistos apenas como uma estimativa parcial do custo total, medida exclusivamente para propiciar comparação entre sistemas de análise distintos. Uma vez escolhidos os sistemas de análise mais indicados para uma dada aplicação, deve-se novamente levantar os custos envolvidos, já que estes custos podem inclusive estar sujeitos a descontos promocionais, ou simplesmente terem aumentado. Um ponto importante a ser destacado é que o processo de escolha de sistemas não envolve somente a análise de desempenho e custos associados, e sim adicionalmente vários outros fatores, tais como funcionalidades oferecidas, disponibilização de *software* e *hardware* compatíveis, escalabilidade do sistema, adequação a padrões consolidados, facilidade de uso e aprendizado, entre outros [Tel91]. *Benchmarks* correspondem apenas a uma ferramenta de auxílio no processo de escolha de sistemas, particularmente importantes na determinação de capacidades e no levantamento superficial de custos. O uso exclusivo de *benchmarks* para responder questões do tipo "Qual sistema eu devo comprar ?" não é indicado, pois este se mostra incompleto na análise de todos os fatores pertinentes ao processo de escolha de sistemas, conforme já descrito.

[Ano+85] descreve uma maneira alternativa de se computar a influência do custo. O custo total, apesar de envolver custos provenientes de componentes de *hardware*, de *software* e de manutenção para um período de 5 anos, não deve ser computado como a simples soma dos preços individuais gerados por cada um destes componentes. Segundo este autor, deve-se relacionar o custo individual de cada componente ao tempo em que é usado no teste. Tomando como base o intervalo de 5 anos para que o preço do sistema

[5] período estimado para que o custo do equipamento se deprecie totalmente.

total se deprecie, ou seja, perca todo o valor, um segundo custaria 1/157.680.000, o que é aproximadamente 6.10^{-9} unidades monetárias, enquanto um minuto e uma hora custariam aproximadamente respectivamente 4.10^{-7} e 2.10^{-5} unidades monetárias. Um exemplo de levantamento de custos segundo esta abordagem é descrita na tabela 3.2.

O custo, apesar de ser reportado separadamente, raramente é utilizado como mecanismo direto de comparação entre sistemas de análise. A necessidade de relacionar o custo a questões de desempenho, as quais muitas vezes compõem uma importante restrição existente no processo de escolha de sistemas, faz com que o custo seja disponibilizado somente na forma de uma razão entre grandezas escalares, do tipo custo/desempenho ou desempenho/custo, onde o desempenho geralmente é medido por uma medida de produtividade e o custo geralmente é medido em milhares de unidades monetárias. Medidas que reportam o custo através de razões são chamadas de medidas de *custo x benefício*, algumas vezes também chamadas de medidas de *custo x desempenho*.

Custo de 45 minutos de Sort ("Teste de Ordenação") [Ano85+]			
Componente	*Custo Total*	*Custo/Minuto*	*Custo Computado*
Processador	80.000,00	0,024	1,08
Memória	15.000,00	0,0045	0,20
Disco	50.000,00	0,015	0,68
Software	50.000,00	0,015	0,68
Total =	195.000,00	0,06	2,63
Obs : custos em dólar (US$), incluindo estimativa de manutenção			

Tabela 3.2 Levantamento de custos em relação ao tempo de uso.

Exemplos típicos de medidas deste tipo são as medidas K$/TPS e K$/TPM dos *benchmarks TPC-A/B* e *TPC-C*, respectivamente (figura 3.7). O custo, para estes três *benchmarks*, é reportado em milhares de dólares. Estas medidas permitem a comparação de diferentes sistemas de análise, desde microcomputadores pessoais até *mainframes*, enfocando o aspecto *custo x benefício*, ao invés somente da eficiência com a qual o sistema de análise atende à realização das transações. Assim, por exemplo, no *benchmark TPC-C* o sistema de análise que apresentar o menor valor de K$/TPM será o que terá a melhor relação *custo x benefício*. Analogamente, para os *benchmark TPC-A* e *TPC-B*, o sistema de análise que apresentar o menor valor de K$/TPS será o que terá a melhor relação de *custo x benefício*.

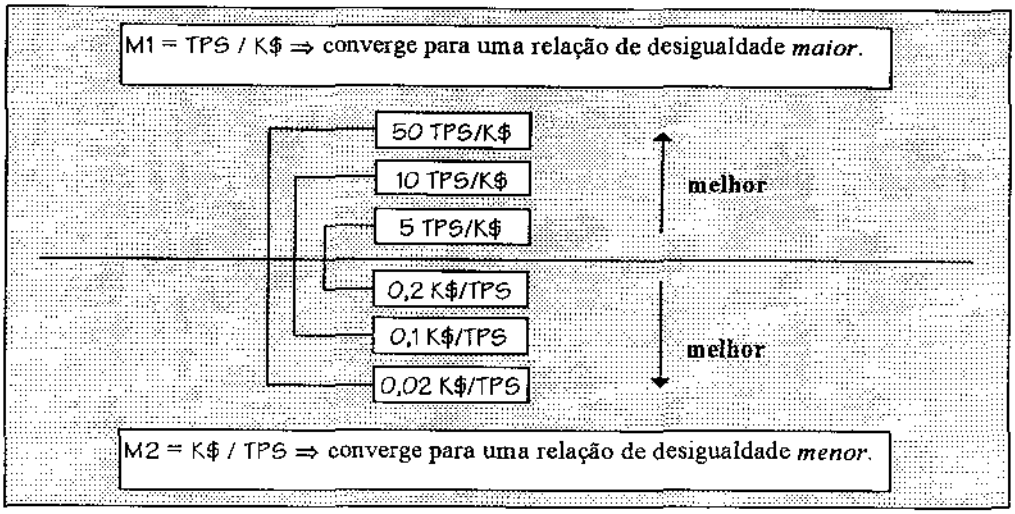


Figura 3.7 Relacionamento inverso entre as medidas TPS/K$ **e** K$/TPS.

Por fim, outra medida interessante, particularmente voltada para sistemas que utilizam armazenamento secundário intensivo, é chamada de *equivalent database size* [TOB91]. Esta medida consiste em reportar o tamanho máximo do banco de dados que um conjunto de transações pode processar dentro de um certo limite de tempo. Isto é muito útil, visto que muitas vezes implementadores tendem a minimizar o tamanho do banco de dados visando tempo de resposta de transações individuais reduzidos. No entanto, caso isto seja feito, a medida *equivalent database size* será decrescida significativamente. A partir desta medida pode-se obter o custo/*Mbyte*, através da razão (custo total do sistema de análise / *equivalent database size)*. O cálculo do custo total é efetuado da mesma forma como descrito anteriormente nesta seção. Esta medida segue o tipo de uso já descrito para as medidas de produtividade, tempo de resposta e custo x benefício.

3.6 Comparação de resultados de desempenho

Suponha que um determinado *benchmark* considere o sistema de análise como sendo o sistema computacional completo. Neste caso, quer-se medir o desempenho do sistema computacional como um todo, não importando quais partes deste geram mais ou menos sobrecarga de execução. Isto é típico de *benchmarks* que modelam ambientes de aplicação, onde o *benchmark* reproduz as atividades tipicamente encontradas em um tipo de aplicação. O objetivo destes *benchmarks*, portanto, é medir o desempenho de uma determinada aplicação, rodando em uma determinada plataforma de *hardware* com um determinado sistema operacional, sendo que esta aplicação utiliza um *software* específico, também chamado de *software* básico, para auxiliar a execução de suas atividades (figura 3.8). Neste caso não existe carga de trabalho externa ao *benchmark*. *Benchmarks* voltados para este propósito são os principais tipos existentes. Para estes sistemas de análise, as

medidas de produtividade, tempo de resposta e *custo x benefício* são indicadas, pois estas realmente refletem o desempenho em nível macro.

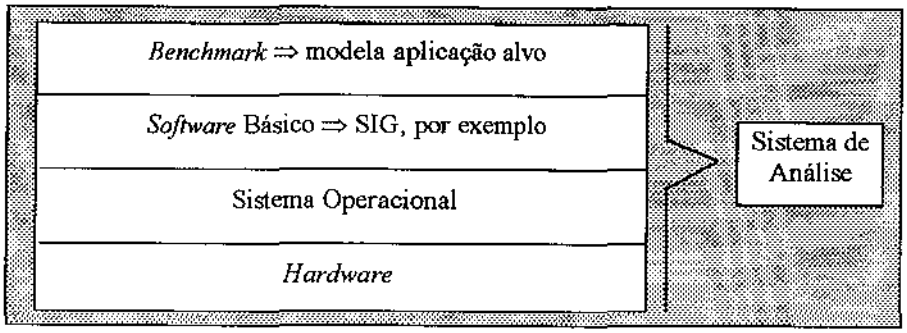


Figura 3.8 Sistema de análise – total sem carga de trabalho externa.

Uma pequena variação do esquema anterior, apresentado na figura 3.9, constitui-se de um sistema de análise com carga de trabalho externa ao *benchmark*. *Benchmarks* voltados ao estudo destes tipos de sistema de análise almejam medir a degradação de desempenho provocada pela carga de trabalho externa ao *benchmark*, a qual influencia diretamente o ambiente de aplicação que o *benchmark* representa.

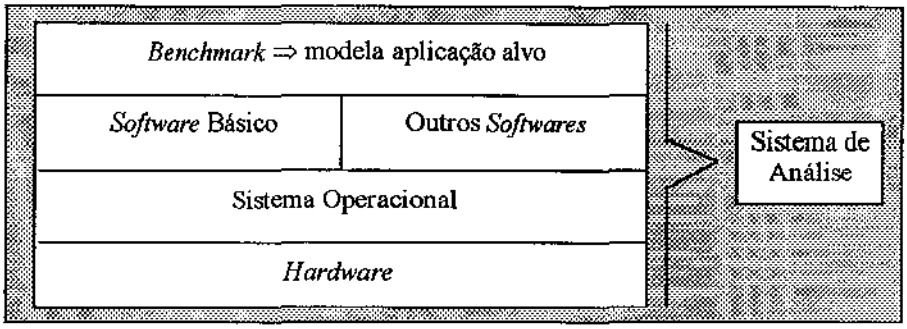


Figura 3.9 Sistema de análise – total com carga de trabalho externa.

Algumas vezes, entretanto, o sistema de análise constitui-se de parte do sistema computacional total, chamado de sistema de análise parcial. Nestes casos, apenas deseja-se medir a influência proporcionada por partes específicas no desempenho do sistema computacional total. Entretanto, o desempenho medido sempre é influenciado pela sobrecarga de execução de todos os processos presentes no sistema, assim como pelo *hardware*, o qual oferece uma determinada capacidade de execução[6]. Assim, a medição da influência gerada por partes específicas, em um único sistema computacional, torna-se impossível, exceção feita a componentes de *hardware*, os quais podem ser medidos independentemente através de medidas de utilização, conforme descrito na seção 3.5. Como sistemas de análise deste tipo geralmente não são formados somente por componentes de *hardware*, e sim principalmente por componentes de *software*, para propósitos práticos o uso de diferentes configurações de sistemas computacionais ajuda a

[6] o termo capacidade de execução está relacionado a todas as capacidades geradas pelo *hardware*, tais como processamento, transferência de dados, armazenamento, entre outros.

isolar a influência gerada por partes específicas. A figura 3.10 visualiza esta situação na ausência de carga de trabalho externa ao *benchmark*. A presença de carga de trabalho externa ao *benchmark*, neste contexto, é semelhante à discutida na figura 3.9.

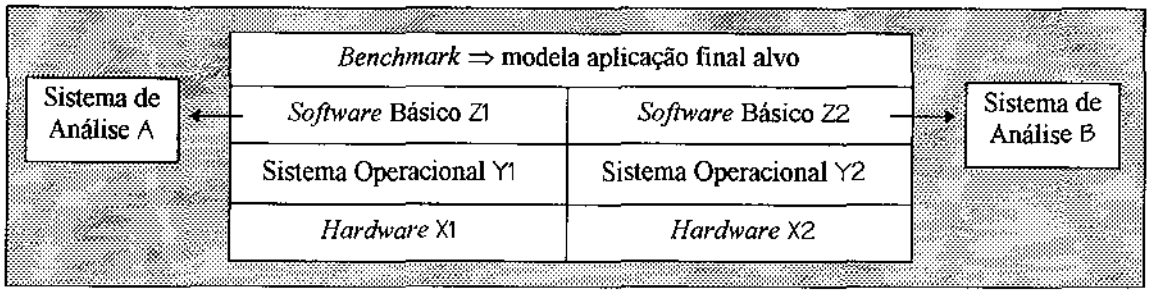


Figura 3.10 Comparação de sistema de análise parciais – sem carga de trabalho externa.

Um *benchmark* acrescenta diretamente uma sobrecarga adicional de execução ao sistema computacional, através da criação de vários processos, durante a sua execução. No entanto, esta sobrecarga pode ser ignorada quando usada na comparação entre configurações distintas de sistemas computacionais, uma vez que ela se anula automaticamente. Isto é ilustrado a seguir.

Na relação abaixo, considere i(x) como sendo a influência proporcionada por um determinado componente (x) no desempenho de um dado sistema computacional, ou seja, a sua sobrecarga de execução. Considere também que (B) = *benchmark*, (SB) = *software* básico, (SO) = sistema operacional e (H) = *hardware*. Além disto, esta relação (⇔) pode ser uma relação de igualdade (=) ou de desigualdade (>) ou (<). Vale notar que os itens em *itálico*, nas equações 3.2, 3.3 e 3.4 se anulam automaticamente na relação.

Configuração 1	Configuração 2
$i(B) + i(SB\ z1) + i(SO\ y1) + i(H\ x1) \Leftrightarrow i(B) + i(SB\ z2) + i(SO\ y2) + i(H\ x2)$	

Equação 3.2 Relacionamento entre configurações de sist. computacionais distintas.

Em adição, quando o (sistema operacional y1 = sistema operacional y2) e o (*hardware* x1 = *hardware* x2), pode-se isolar comparativamente o desempenho do *software* básico (tabela 3.4).

Configuração 1	Configuração 2
$i(B) + i(SB\ z1) + i(SO\ y) + i(H\ x) \Leftrightarrow i(B) + i(SB\ z2) + i(SO\ y) + i(H\ x)$	

Equação 3.3 Relacionamento entre configurações de sistemas computacionais distintas com ênfase no *software* básico.

Em geral, quando quer-se medir a influência proporcionada por partes específicas de um sistema computacional, utiliza-se a abordagem de análise comparativa entre diferentes configurações de sistemas computacionais (tabela 3.5). Note que as demais partes permanecem constantes, ou seja, sempre estão presentes nestas configurações.

Configuração 1	Configuração 2
$I(B) + I(\text{Parte p1}) + I(\text{Outras partes}) \Leftrightarrow I(B) + I(\text{Parte p2}) + I(\text{Outras partes})$	

Equação 3.4 Uso de distintas configurações de sistemas computacionais para se isolar o desempenho proporcionado por partes específicas.

Os resultados de desempenho gerados por medidas de produtividade, tempo de resposta e *custo x benefício* refletem, entretanto, somente o desempenho do sistema computacional total e portanto não podem ser usados para indicar o desempenho de sistemas de análise parciais. Estas medidas podem ser usadas somente para providenciar comparação relativa de sistemas de análise parciais distintos. Isto, apesar de simples, muitas vezes é ignorado, por questões de *marketing*. Assim, um resultado produzido pelo *benchmark TPC-A* pode ser identificado por um fabricante de *hardware* como sendo o desempenho do *hardware*, enquanto que para um fabricante de SGBD este constitui o desempenho do SGBD, quando na verdade este é o desempenho do sistema computacional total, em relação à aplicação alvo representada pelo *benchmark*.

As classes descritas a seguir, resumidas na tabela 3.6, são úteis quando o sistema de análise é formado por um subconjunto do sistema computacional total, permitindo-se comparação de desempenho relativa. Nestes casos, cada tipo de classe é voltada à análise de desempenho de um subconjunto específico do sistema computacional total, sempre em pares, comparativamente. Para as classes 1 e 2, onde o subconjunto analisado é o próprio sistema computacional total, a obrigatoriedade da análise comparativa é relaxada. Assim, pode-se dizer que a análise individual de um resultado gerado por um sistema de análise composto pelo sistema computacional total é correta, caso contrário não, podendo ser efetuada apenas comparativamente com outros resultados gerados para outros sistemas de análise do mesmo tipo. Como o *benchmark* representa neste caso a aplicação alvo, os componentes que podem variar são o *hardware*, o sistema operacional e o *software* básico (SIG no contexto deste trabalho), conforme ilustrado na figura 3.8, a qual representa um sistema computacional multinível simplificado que contém os principais níveis que influenciam significativamente o desempenho. Isto conduz a 8 classes distintas de comparação, sendo que cada uma possui um propósito especial. Vale ressaltar que a carga de trabalho total considerada é gerada rigorosamente pelos componentes variáveis e pelo *benchmark*. A adição de carga de trabalho externa ao *benchmark* apenas especializa cada tipo de classe, sendo que os conceitos não se alteram, indicando somente que o sistema de análise em questão está sob efeito desta carga de trabalho adicional.

Classes	*Hardware*	*S.O.*	*S.M.*	*Uso*	*Sistema de análise*
1	=	=	=	controle de qualidade	sistema computacional total
2	<>	<>	<>	comparação entre soluções completas de distintos fornecedores	sistema computacional total
3	<>	=	=	comparação de diferentes modelos de computadores.	hardware
4	=	<>	=	comparação de diferentes sistemas operacionais.	SO
5	=	=	<>	comparação de diferentes softwares básico.	SB
6	=	<>	<>	medir influência do SO + SB no hardware	SO + SB
7	<>	=	<>	medir influência do hardware + SB no sistema operacional	hardware + SB
8	<>	<>	=	medir influência do hardware + SO no software básico	hardware + SO

Tabela 3.3 Classes de comparação para sistemas de análise parciais.

- **classe 1:** comparação de sistemas de análise com as mesmas configurações de *hardware*, sistema operacional e *software* básico. A priori, uma comparação deste tipo não teria sentido algum, uma vez que ambos os sistemas possuem a mesma configuração. No entanto, isto pode ser utilizado no processo de controle de qualidade. Dois sistemas de análise com a mesma configuração podem possuir componentes físicos distintos, como exemplo, UCPs físicas. Assim, caso a UCP de um destes sistemas de análise apresente problemas de fabricação, o desempenho para este sistema de análise será degradado e portanto inferior ao do outro sistema de efetuarem testes intensivos de controle de qualidade, minimizando situações como a descrita anteriormente análise, embora ambos possuam a mesma configuração. Apesar dos fabricantes, casos específicos são passíveis de ocorrer. Situações em que o baixo desempenho parece ser gerado por algum componente de *hardware* são as indicadas para este tipo de comparação.

- **classe 2:** comparação de sistemas de análise com diferentes configurações de *hardware*, sistema operacional e *software* básico. Isto indica um panorama típico de competitividade entre diferentes vendedores, onde cada um fornece sua solução completa. Assim, pode-se comparar a influência de distintas soluções completas no desempenho da aplicação.

- **classe 3:** comparação de sistemas de análise com diferentes configurações de *hardware* somente. O enfoque deste tipo de comparação é a análise do *hardware*, sendo usado para determinar a influência específica que o *hardware* gera no desempenho. Através deste tipo de comparação consegue-se determinar o desempenho relativo de vários modelos de computadores. Este desempenho sempre é em relação à aplicação alvo representada pelo *benchmark*. A mudança de aplicação, e conseqüentemente do *benchmark*, pode alterar este desempenho. Assim, um modelo de computador é dito ser melhor que outro modelo, tomando como base a execução da aplicação alvo. Não se deve pensar jamais que esta comparação é universal. Desta forma, um modelo de computador por possuir melhor resultado de desempenho que outro modelo através da aplicação de um determinado *benchmark* não deve ser visto sempre como superior, a não ser especificamente para a aplicação alvo que o *benchmark* representa. Resultados de desempenho não podem ser extrapolados.

A tabela 3.7 mostra um caso específico deste tipo de comparação, aplicado a vários modelos de computadores da mesma família.

Modelo	*TPS*	*TPS Relativo*	*K$/TPS*	*K$/TPS Relativo*
HP 960	38,20	7,64	29,20	2,13
HP 957LX	49,40	9,88	13,70	1,00
HP 948	38,40	7,68	19,90	1,45
HP 932	13,60	2,72	28,80	2,10
HP 920	5,00	1,00	28,80	2,10
Observações :				
1 - para a medida TPS o modelo base é o HP 920				
2 - para a medida K$/TPS o modelo base é o HP957LX				

Tabela 3.4 Desempenho de modelos de computadores de uma mesma família.

- **classe 4:** comparação de sistemas de análise com diferentes configurações de sistema operacional. Neste caso quer-se analisar a influência gerada pelo sistema operacional no desempenho. Muitas vezes, por exemplo, versões de um mesmo *software* básico disponíveis para sistemas operacionais distintos geram desempenho desiguais. Este tipo de comparação ajuda a identificar este fato. Além disto, pode-se determinar o desempenho relativo de sistemas operacionais, inclusive de *releases*.

- **classe 5:** comparação de sistemas de análise com diferentes configurações de *software* básico somente. Aqui, o intuito é quantificar comparativamente o desempenho gerado por um *software* básico, tal como um SIG. Uma vez que o único componente alterado é o próprio *software* básico, a diferença nos resultados de desempenho reportados se deve exclusivamente a este componente. Este tipo de comparação é usado tanto entre *softwares* básicos de fabricantes distintos, caracterizando uma típica situação de competitividade entre vendedores de *software*, quanto entre *releases* de um mesmo fabricante. A comparação de *releases* é particularmente interessante. Devido ao acréscimo de novas funcionalidades, as quais tornam a sobrecarga de execução do *software* básico maior, este geralmente tende a perder desempenho. A tabela 3.8 mostra este uso, para o *software* RdB.

Software	*TPS*	*K$/TPS*		
Mestre				Observação : os resultados foram reportados
				a partir do benchmark TPC-A, com configuração
RdB release 3	21,7	31,9		de rede local e rodando em um computador
RdB release 4	21	17,4		VAX 4000-300 com um único processador

Tabela 3.5 Comparação entre *releases* de um mesmo *software* básico.

- **classes 6, 7 e 8:** estas classes mantêm fixo somente um dos possíveis componentes variáveis. A classe 6 mantém fixo o *hardware*, a classe 7 o sistema operacional e a

classe 8 o *software* básico. O intuito destes tipos de comparações é avaliar a influência gerada pelos componentes variáveis no componente fixo, para uma dada aplicação particular também fixa. Por exemplo, comparações do tipo da classe 8 demonstram quanto um *software* básico é influenciado com a alteração conjunta do *hardware* e do sistema operacional. Isto é típico em situações onde um *software* básico irá rodar a mesma aplicação sobre plataformas de *hardware*/sistema operacional distintas. Analogamente, a classe 6 ilustra situações onde um *hardware* será adotado como padrão por uma organização, porém esta utilizará de sistemas operacionais e *softwares* básicos distintos, para executar a mesma aplicação. Este tipo de comparação ajuda a identificar possíveis combinações de sistema operacional/*software* básico que degradam muito o desempenho para o *hardware* em questão, indicando a necessidade de *upgrade* do *hardware* escolhido. Por fim, a classe 7 ilustra uma situação menos freqüente, onde um único sistema operacional comporá todas as configurações de *software* básico/*hardware* utilizadas para rodar uma dada aplicação.

Capítulo 4

Levantamento de características de aplicações georeferenciadas

4.1 Análise de desempenho de SIGs

Na literatura, a caracterização de transações típicas de aplicações georeferenciadas é feita basicamente em função de duas transações: superposição (convencional – seção 5.2.2) e análise de proximidade (simples – seção 5.2.4). Adicionalmente, a descrição da necessidade de computação de valores escalares derivados diretamente de dados espaciais, tais como área e perímetro, é largamente referenciada. Em relação à busca espacial, somente são citadas transações topológicas *booleanas* (seção 5.2.6), principalmente para os relacionamentos topológicos de adjacência, inclusão e interseção. Este mesmo fato também tem sido observado na caracterização de cargas de trabalho de *benchmarks* para SIGs. Para estes *benchmarks*, os dados utilizados são exclusivamente reais, não havendo nenhuma menção a respeito da geração de dados sintéticos. A utilização de dados sintéticos permite a adição de características especiais, de modo a explorar deficiências encontradas ou pressentidas em aplicações georeferenciadas, as quais podem degradar o desempenho significativamente.

[GR87] descreve um modelo formal de análise de desempenho que associa produtos às subtarefas (transações primitivas – seção 5.1) necessárias às suas execuções, em função de uma estimativa de utilização de recursos computacionais (tempo de UCP, tempo do operador, tempo de "*plottagem*" e espaço de armazenamento em disco) e de custos. A ênfase deste trabalho é a caracterização de um modelo genérico que permita a modelagem de aplicações georeferenciadas distintas. Desta forma, não é definido um conjunto de transações consideradas típicas de aplicações georeferenciadas, sendo apenas mencionado algumas transações encontradas em aplicações reais modeladas segundo este modelo. Estas transações incluem as tradicionais transações de superposição e análise de proximidade, além de algumas operações ligadas às fases de coleta de dados e pré-processamento, tais

como *edgematching* (junção de tiras distintas de um mesmo mapa, as quais foram digitalizadas separadamente), "*scannerização*", vetorização semi-automática e correções de erros neste último processo. A busca de informações topológicas não é abordada neste trabalho. Para os dados do *benchmark*, sugere-se a adoção de dados reais, de acordo com a aplicação sendo analisada. Como citado pelo próprio autor, o uso deste modelo formal tem por objetivo apenas a geração de resultados de desempenho dependentes de uma aplicação georeferenciada específica, não podendo ser utilizado para propósitos de comparação de SIGs distintos.

[Bou88] descreve um conjunto de consultas tipicamente encontradas em aplicações urbanas, as quais são voltadas principalmente ao gerenciamento de redes, tais como rede de eletricidade, de esgoto, de água e de telefonia. Dentre as principais consultas pode-se citar: análise de proximidade, cruzamento de redes e localização de pontos em redes. Estas consultas (transações), por serem específicas, não servem para propósitos comparativos, sendo indicadas somente para SIGs que utilizam redes de modo intensivo.

Já [Mcl90] descreve um conjunto de testes funcionais e de desempenho necessários a um *benchmark* voltado à análise de SIGs. Entretanto, esta descrição é feita apenas superficialmente, mais para estimular a criação de um *benchmark* do que para caracterizar um. Nenhuma menção em relação à geração de dados é citada neste trabalho.

Por fim, [Sto+93] descreve um *benchmark* voltado à análise de desempenho do projeto Sequoia 2000. Este *benchmark* possui uma carga de trabalho composta de 11 transações, as quais abrangem: a construção inicial de índices durante a carga do banco de dados, consultas específicas para o formato *raster*, consultas envolvendo objetos geográficos 0 e 1-dimensionais no formato vetorial, uma consulta contendo conjuntamente dados espaciais no formato *raster* e vetorial, e recursão em redes. Os dados utilizados pelo *benchmark* são reais e pertencentes ao projeto Sequoia 2000, sendo que dependendo da quantidade de dados utilizada o *benchmark* é classificado em regional (1 *Gbyte*), nacional (17 *Gbytes*) ou global (2 *Tbytes*). No entanto, este *benchmark* possui um conjunto restrito de transações, insuficientes para expressar a gama de aplicações georeferenciadas. Em outras palavras, este conjunto de transações é específico, tendo sido projetado para atender somente às necessidades do projeto Sequoia 2000. Isto pode ser notado facilmente pela ausência de algumas típicas transações SIGs, tais como superposição e análise de proximidade.

Inversamente aos trabalhos acima descritos, a definição da carga de trabalho de um *benchmark* voltado à análise de desempenho de SIGs requer um conjunto abrangente de transações, como será verificado no capítulo 5. Algumas destas transações são: reclassificação (seção 5.2.1), superposição (seção 5.2.2), análise de ponderação (seção 5.2.3), análise de proximidade simples, múltipla e ao redor de múltiplos objetos geográficos (seção 5.2.4), decomposição de objetos geográficos bidimensionais (seção 5.2.5), transações topológicas *booleanas* (seção 5.2.6), busca topológica (seção 5.2.7), transações baseadas em conjunto (seção 5.2.10) e transações de conversão de formato de dados (seção 5.2.11).

Em relação aos dados, é interessante que um *benchmark* utilize ambos dados reais e sintéticos (seção 3.4), de forma contrária aos trabalhos já citados. A caracterização dos dados que comporão o *benchmark* será feita de forma independente de sua organização (estrutura dos arquivos, quantidade de dados armazenados, entre outros), de modo a permitir a adaptação dos dados segundo aplicações georeferenciadas específicas. Esta caracterização será efetuada em termos dos tipos de dados necessários para a representação de aplicações georeferenciadas, e adicionalmente procedimentos para se realizar a geração de dados sintéticos. A seção 6.2 descreve uma aplicação alvo composta de dados sintéticos que permite a comparação de desempenho de SIGs distintos, através da inserção de uma constante de degradação única no desempenho, relativa aos dados e às transações.

A forma na qual o conjunto de transações e os dados do *benchmark* proposto foram determinados é descrita nas próximas seções.

4.2 Métodos de levantamento de características

Esta seção descreve as formas de levantamento das características típicas de aplicações georeferenciadas. Os principais meios utilizados para este propósito foram a distribuição de um questionário, visitas a algumas das principais instituições e eventos relacionados ao geoprocessamento no Brasil, além da leitura de revistas e textos especializados. O questionário foi de propósito geral, e englobou aspectos relacionados ao *hardware*, *software* e *liveware*, servindo não somente para o levantamento das características espaciais destes sistemas, mas também para caracterizar o segmento de geoprocessamento nas principais instituições brasileiras. A criação do questionário foi efetuada em parceria com o Instituto de Geociências da Unicamp e o CPqD Telebrás, com o intuito de aproximar a pesquisa aos usuários de SIGs.

O levantamento de características típicas de aplicações georeferenciadas visa principalmente à elaboração da carga de trabalho e à caracterização dos dados do *benchmark*. O contato com usuários e desenvolvedores, através de entrevistas, questionário ou simples leitura de textos especializados, faz com que o *benchmark* se torne mais representativo. O conceito de representatividade é importante, uma vez que um *benchmark* deve definir com rigor o que é representativo para a aplicação alvo que este representa (aplicações georeferenciadas, no caso deste trabalho). Isto deve ser feito para que os resultados de desempenho produzidos pelo *benchmark* sejam os mais próximos da realidade. Por exemplo, *benchmarks* que analisam sistemas bancários, onde existe uma alta taxa de entrada e saída tanto em nível de arquivos quanto em nível de terminais devem possuir transações do tipo simples e *on-line*. Por outro lado, estes *benchmarks* não devem possuir transações que façam cálculos matemáticos complexos, uma vez que em sistemas bancários a maioria absoluta das operações matemáticas são básicas, tais como soma e subtração.

Segundo [Ros95], a utilização de SIGs no Brasil se divide basicamente em três segmentos distintos: empresas governamentais e prefeituras, empresas privadas, e universidades e centros de pesquisa. O questionário foi enviado a instituições representantes destes três segmentos, sendo no total enviado a 120 instituições e respondido por 50 instituições. Como exemplo de instituições que responderam o questionário preenchido pode-se citar: Light, Eletropaulo, Celpe, Telebrás, Telesp, Telemig, Petrobrás, IBGE, INPE, Embrapa/NMA, IplanRio, USP, UFRJ, UFSCar, NEPO/Unicamp, UnB, UFPE, UFMG, UFV, UNISINOS, IPT, UFSC, UNIVAP, P.M. de São Paulo, P.M. de Santo André, IPPUC – Curitiba, Prodabel – Belo Horizonte, P.M. de Guarulhos, P.M. de Goiânia, P.M. de Porto Alegre, P.M. de Lajeado, P.M. de São Bernardo do Campo, Carta Consultoria, Inpacel e Aerosul S/A.

O questionário utilizado para o levantamento de características de aplicações georeferenciadas é descrito resumidamente no apêndice B. A seção 4.3 descreve os resultados coletados através do questionário e outras fontes.

4.3 Resultados coletados

O principal objetivo da elaboração do questionário, no contexto da dissertação, foi realizar um estudo de aplicações georeferenciadas reais implantadas em instituições localizadas no território brasileiro e, deste modo, determinar as transações básicas e o conjunto de dados utilizados por estas aplicações reais. Para as aplicações georeferenciadas pesquisadas, as seguintes características foram detectadas:

- Verificou-se o uso de distintos SIGs, sendo os mais utilizados os SIGs Arc/Info, Spring, SGI/Inpe, Idrisi, GeoVision, Grass, Intergraph, Apic, MapInfo e Erdas.

- Verificou-se a necessidade de armazenamento dos três tipos de dados geográficos (espaciais, convencionais e gráficos) para a maioria das aplicações. Entretanto, a forma como estes dados se organizam diferiu. Aplicações de pequeno porte utilizam bancos de dados baseados em PC (tais como Clipper e D-Base), e também a estrutura de arquivos *flat*. Um segundo grupo de aplicações de médio e grande porte utilizam somente a estrutura de armazenamento interna do SIG na qual estão baseadas. Como exemplo, pode-se citar a Prodabel – Processamento de Dados do Município de Belo Horizonte, que utiliza o SGBD proprietário cchr do SIG APIC. Por fim, um terceiro grupo de aplicações utilizam dois SGBDs, um interno ao SIG para o armazenamento de dados espaciais e gráficos, e outro externo ao SIG (SGBD convencional) para o armazenamento de dados convencionais. Isto é verificado na maioria das aplicações de grande porte, tais como na Telebrás (GeoVision – Oracle), IBGE (Arc/Info,

Intergraph, Microstation – Informix), Eletropaulo (Arc/Info – Sybase), Light (Arc/Info – Adabas) e INPE (Spring – Ingres).

Para aplicações de grande porte, a escolha da utilização de dois SGBDs deve-se principalmente à necessidade de se aproveitar uma base de dados existente, ou à necessidade de se garantir portabilidade e independência em relação ao SIG subjacente, em contraste com qualquer decisão baseada em desempenho. Ao contrário, aplicações do segundo grupo visam aproveitar ao máximo toda potencialidade do SIG subjacente de forma a obter um alto desempenho. A execução do *benchmark* proposto nesta tese em aplicações organizadas segundo os grupos 2 e 3 poderá revelar desempenhos completamente diferentes para cada grupo, assim como poderá comprovar a independência do desempenho em relação à forma de organização dos dados geográficos (seção 6.2). Aplicações do grupo 1, devido à sua limitada abrangência, devem ser examinadas apenas através de configurações específicas de dados geográficos reais.

- Verificou-se uma quantidade de poucos *Mbytes* de dados geográficos armazenados em aplicações de pequeno porte, até 5 *Gbytes* em aplicações de grande porte. Entretanto, nestas últimas aplicações, o banco de dados geográfico de cada uma destas ainda está em fase de formação, sendo que projeções de até 30 *Gbytes* de dados geográficos (Eletropaulo) são previstas na conclusão da aplicação georeferenciada. Estes parâmetros serviram de base para a elaboração da quantidade de dados presente na aplicação alvo, descrita na seção 6.2. Quanto ao formato de armazenamento de dados espaciais, verificou-se um uso predominante de dados no formato *raster* (do tipo quadrático), para o modelo baseado em campos, e um uso predominante de dados no formato vetorial segundo os modelos Arc/Info, DIME e de objetos relacional, para o modelo baseado em objetos. Os formatos de dados *raster* e vetorial foram detectados conjuntamente na maioria das aplicações.

- Verificou-se que os dados geográficos, principalmente os dados convencionais, estão armazenados predominantemente segundo o modelo relacional. A utilização do paradigma de orientação a objetos, apesar de ser pouco utilizado em sistemas reais, foi constatada inclusive em uma aplicação de médio a grande porte. Além disto, verificou-se que a modelagem conceitual, quando efetuada, segue em maioria o modelo entidade-relacionamento. Em poucas aplicações constatou-se o uso do modelo entidade-categoria-relacionamento, e em somente uma aplicação constatou-se o uso de um modelo de dados geográfico desenvolvido especificamente pelo grupo de desenvolvimento da aplicação.

- Verificou-se o uso predominante de plataformas baseadas em estações de trabalho (IBM Risc/6000, Sun Sparc, HP-Apolo, entre outras), com sistema operacional Unix (Aix, SunOS, Solaris e HP-UX, entre outros). Entretanto, constatou-se também uma grande parcela de plataformas baseadas em PC (PC 386/486) com sistema operacional DOS/Windows 3.1. O uso dos sistemas operacionais Windows for Workgroups, OS/2 e VMS, apesar de verificados, formam uma parcela irrelevante

das aplicações consultadas. A quantidade de memória principal presente nas configurações de *hardware* variou de 4 *Mbytes* em PCs até 128 *Mbytes* em estações de trabalho.

Resultados Coletados (resumo)

- 21 SIGs distintos (diversidade)

Arc/Info	20	22,47%
Spring	11	12,36%
Idrisi	10	11,24%
Intergraph e SGI	8	8,99%

- Tipos de Dados

Convencionais e Espaciais no formato vetorial	100,0%
Espaciais no formato *raster*	93,2%

- Organização dos Dados

SIG + SGBD Relacional (Ex: Telebrás ⇒ Vision + Oracle)
SIG + SGBD O.O. Proprietário (Ex: Prodabel ⇒ Apic)

Figura 4.1 Resumo dos resultados coletados

- Para a maioria das aplicações, verificou-se uma necessidade real do SIG oferecer algum modo de se formar consultas envolvendo operadores espaciais, geométricos, topológicos e convencionais conjuntamente. Isto basicamente é efetuado através de uma linguagem de consulta proprietária do SIG subjacente (freqüentemente baseada em SQL), ou por uma estrutura de menus. Dentre os operadores mais utilizados, basicamente se destacam os operadores já presentes no questionário, tais como os operadores geométricos de área, perímetro, comprimento e distância, os operadores topológicos de interseção, inclusão e adjacência, e os operadores espaciais de união, interseção e decomposição de objetos geográficos bidimensionais em interior e limites, além claro de operadores convencionais típicos. O resultado de consultas, entretanto, ficou dividido entre a dependência e a independência em relação à sua visualização.

- Dentre as funções típicas de um SGBD, somente as funções de consulta e inserção foram tidas como freqüentes por todas as aplicações consultadas. Isto caracteriza um ambiente com poucas alterações nos dados, em parte devido ao alto custo e tempo para se realizar a fase de coleta de dados.

- Para a maioria das aplicações, verificou-se uma aderência às funções analíticas presentes no questionário. Em adição, algumas funções analíticas foram verificadas, dentre as quais pode-se citar: distância mínima entre dois objetos geográficos, cálculo da diferença entre dois objetos geográficos, conversão dos formatos *raster* e vetorial para os formatos DXF e PostScript, além do agrupamento de células e manipulação de matrizes no formato *raster*. Dentre as consultas típicas verificadas pode-se citar: busca topológica, descrita na seção 5.2.7.

Capítulo 5

Proposta de um *Benchmark* para SIGs: carga de trabalho e dados

5.1 Introdução

Sistemas de informações geográficas são utilizados em muitos tipos distintos de aplicações, as quais incluem aplicações de mapeamento urbano básico, aplicações de monitoramento ambiental, entre outras (seção 2.4). A ênfase deste capítulo é caracterizar a carga de trabalho de um *benchmark* voltado à análise de desempenho de SIGs, além de descrever aspectos relacionados aos dados. A caracterização da carga de trabalho será efetuada de acordo com o modelo de subtarefas. Neste modelo, um conjunto de transações primitivas é definido, possibilitando assim a formação de transações mais complexas. Já a caracterização dos dados será efetuada em termos dos tipos de dados necessários para a representação de aplicações georeferenciadas, e adicionalmente procedimentos para se realizar a geração de dados sintéticos.

A seguir são descritas várias características que devem ser levadas em consideração tanto para este capítulo quanto para os capítulos subseqüentes.

- a caracterização de dados espaciais no formato varredura será feita exclusivamente em função do formato *raster* quadrático. Aplicações georeferenciadas utilizam em sua maioria este formato regular, desqualificando a utilização de outros formatos de células (seção 2.3). O termo *raster* quadrático, entretanto, será mencionado apenas como *raster*, por simplicidade.

- a caraterização de dados espaciais no formato vetorial será efetuada em termos de pontos, linhas e polígonos, independentemente do modelo utilizado (modelo Arc-Node, DIME, de objetos relacional, entre outros).

- a utilização das palavras ponto, linha e polígono será realizada apenas para se referir às coordenadas de dados espaciais no formato vetorial. Devido à utilização destes conceitos em ambos formatos (*raster* e vetorial), o uso das palavras ponto, linha e polígono, indistintamente, poderia causar dúvidas de interpretação. Por exemplo, a descrição "polígonos são superpostos" poderia levar à equivocada conclusão de que os polígonos estão no formato vetorial, enquanto estes poderiam estar exclusivamente no formato *raster* (conjunto de células contíguas entre si agrupadas logicamente para formar um polígono), ou ainda em ambos formatos.

- a utilização do termo objeto geográfico será feita para referenciar geometria, independente do formato de dados utilizado (*raster* e vetorial). A especificação da dimensão espacial, quando necessária, será feita logo após o termo objeto geográfico (OG), sendo abreviado para OG-0D para objetos geográficos sem dimensão espacial (geometria na forma de um ponto), OG-1D para objetos geográficos unidimensionais (geometria na forma de uma linha) e OG-2D para objetos geográficos bidimensionais (geometria na forma de um polígono). Quando a dimensão espacial for irrelevante, será utilizado apenas o termo objeto geográfico (OG). Como exemplo, a descrição de "polígonos são superpostos" será efetuada como "OGs-2D são superpostos", sendo implícita a interpretação que estão sendo superpostas as geometrias dos objetos geográficos bidimensionais envolvidos.

- a utilização do conceito de categoria será realizada independente do formato de dados e, quando este formato for o vetorial, será considerada implícita a necessidade de se associar um dado convencional ao objeto geográfico sendo representado. SIGs utilizam o conceito de categoria para descrever o domínio de temas, principalmente para dados no formato *raster*. Entretanto, aplicações que utilizam dados no formato vetorial também têm implementado este conceito, através da associação de um dado convencional, o qual representa a categoria, a cada objeto geográfico de um dado tema, como exemplo polígonos. Assim, objetos geográficos são estruturados de acordo com o tema a que pertencem. A figura 5.1 ilustra um exemplo do uso do conceito de categoria. Vale notar que não existe nenhuma restrição que impeça que dois objetos geográficos distintos de um mesmo tema possuam a mesma categoria. Nesta figura, o tema atividades agropecuárias é formado por seis objetos geográficos, existindo somente quatro tipos de categorias. Em geral, categorias são associadas a objetos geográficos bidimensionais.

As transações primitivas que comporão a carga de trabalho do *benchmark* proposto serão predominantemente orientadas aos dados espaciais, sendo a priori independentes do formato de dados utilizado (*raster* ou vetorial). A caracterização de transações independentemente do formato é preferível, desde que atualmente existem algoritmos nos dois formatos padrões, para a maioria das transações, sendo que adicionalmente existem

algoritmos de conversão entre estes formatos. Entretanto, algumas destas transações serão especialmente voltadas para o formato *raster* (seção 5.3.9), e ainda algumas necessitarão de ambos formatos (seção 5.3.10). Em geral, a cultura de geoprocessamento afirma que para uma mesma transação, um algoritmo vetorial torna-se mais complexo que um algoritmo *raster*. Esta afirmação baseia-se na premissa que um algoritmo *raster* basicamente é um algoritmo seqüencial, o qual lê os valores célula por célula em cada uma das matrizes envolvidas e efetua para cada célula lida alguma operação simples que resulta também na escrita seqüencial da matriz de saída. Por outro lado, em algoritmos vetoriais existe a formação de novos polígonos, além da fusão de polígonos antigos, o que supostamente exige uma sofisticação adicional.

A independência da caracterização das transações primitivas em relação ao formato de dados propicia um método para a verificação da asserção que algoritmos *raster* são mais rápidos que algoritmos vetoriais. O grande número de células armazenadas no formato *raster* puro, assim como o uso da técnica de codificação para diminuir o número destas, acrescentam uma complexidade extra na execução de algoritmos *raster*, o que pode tornar-se um fator determinante no desempenho destes algoritmos. Além disto, apesar da complexidade inerente para determinar a formação de novos polígonos e a fusão de polígonos antigos, o número de polígonos gerados no formato vetorial é relativamente reduzido quando comparado à quantidade de células geradas no formato *raster*.

Este capítulo está organizado da seguinte maneira. A seção 5.2 apresenta a carga de trabalho do *benchmark* proposto, descrevendo conceitualmente cada transação primitiva considerada representativa para SIGs. Com o intuito de facilitar a implementação destas transações em linguagens de consultas de SIGs comerciais, uma notação formal é apresentada na seção 5.3. Adicionalmente, exemplos de representação de transações complexas (consultas presentes em aplicações reais) por um conjunto de transações primitivas é descrito na seção 5.4. Este capítulo também descreve na seção 5.5 aspectos relacionados aos dados, desde a geração de dados sintéticos até a caracterização dos principais tipos de dados presentes em aplicações georeferenciadas.

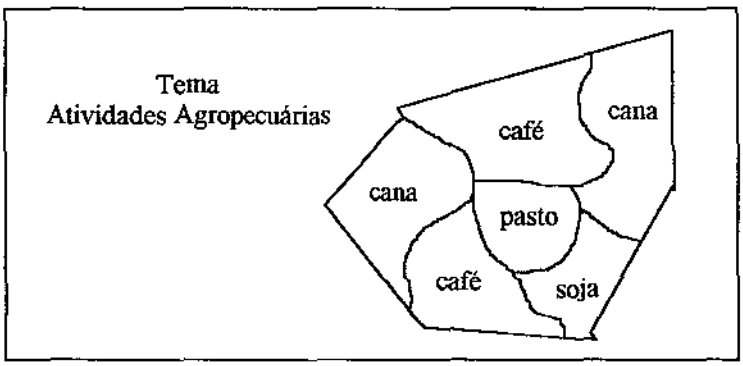


Figura 5.1 Objetos geográficos x categorias.

5.2 Carga de trabalho

Esta seção descreve as categorias básicas de transações efetuadas em SIGs.

5.2.1 Reclassificação

A reclassificação consiste em agrupar categorias de um tema, primitivo ou derivado, de modo a gerar um novo conjunto de categorias menos detalhado para o tema em questão. Ao agrupar categorias, vários OGs do tema original podem ser fundidos em um único OG, diminuindo o número de OGs no tema reclassificado. Em geral, reclassifica-se categorias de OGs-2D.

Um exemplo desta transação é descrito a seguir. Durante a fase inicial de construção de estradas é necessário saber o tipo de solo, desde que certos tipos são impróprios para tal propósito. Um engenheiro ao consultar um mapa geológico encontraria a descrição dos tipos de solos (categorias) e das regiões que cada tipo abrange. Entretanto, as informações contidas em um mapa geológico são de certa forma inapropriadas, desde que são muito detalhadas. Para o engenheiro, o ideal seria um mapa que mostrasse de maneira simples quais áreas são impróprias e quais áreas são próprias para a construção de estradas, segundo o aspecto solos. Para isto, os tipos de solos podem ser agrupados em dois tipos: próprios e impróprios. Suponha que uma determinada região geográfica possua os tipos de solos S1, S2, S3, S4 e S5, sendo que os tipos S1, S3 e S4 são considerados impróprios para a construção de estradas. Assim, a reclassificação agruparia estes tipos em uma categoria C1 (imprópria), enquanto as categorias S2 e S5 seriam agrupadas na categoria C2 (própria). A figura 5.2 ilustra esta situação.

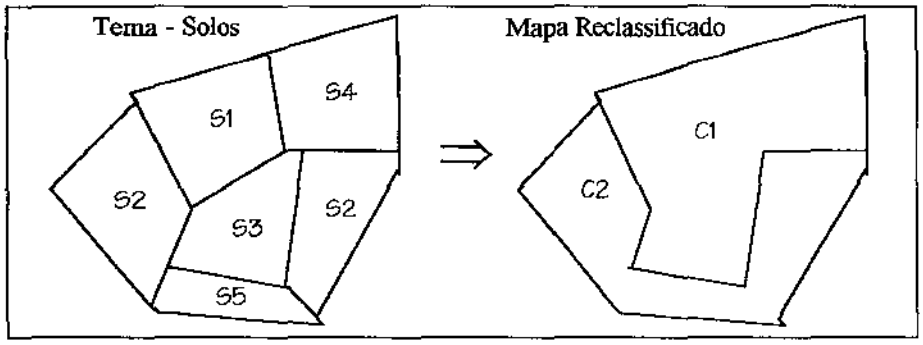


Figura 5.2 Exemplo de Reclassificação.

A reclassificação, segundo a cultura de geoprocessamento, é mais complexa em sistemas que utilizam o formato vetorial, uma vez que o agrupamento de categorias pode levar à fusão de polígonos, enquanto que em sistemas que utilizam o formato *raster* a transação consiste simplesmente em modificar o valor de cada célula da matriz. Entretanto, o número de células no formato *raster* puro, assim como a codificação de células para diminuir o número destas acrescentam uma complexidade extra para a execução desta transação, conforme discutido na seção 5.1.

5.2.2 Superposição

Existem vários tipos de variantes desta transação. Entretanto, todos os tipos possuem em comum o fato de superpor OGs-2D de temas distintos, diferindo somente no modo como as categorias resultantes são geradas. A superposição pode ser aplicada para vários temas, sendo efetuada geralmente aos pares, sucessivamente. A descrição desta transação será, por simplificação, efetuada apenas para dois temas.

O primeiro tipo de superposição, o mais utilizado, é chamado de superposição convencional. Este consiste em superpor OGs-2D de dois temas, sendo que a categoria de cada OG-2D resultante é formada pela junção das categorias de cada um dos temas envolvidos. Assim, novas categorias são geradas indicando a interseção das categorias dos temas originais, ou seja, cada categoria gerada indica a ocorrência simultânea de uma categoria do tema 1 e de uma categoria do tema 2.

Na figura 5.3 estão representados dois temas distintos: solos e vegetação. A superposição consiste, neste exemplo, em superpor estes dois temas de forma a gerar uma nova subdivisão geográfica e conseqüentemente uma nova distribuição de categorias. Para o tema solos, a região geográfica em questão possui quatro OGs-2D, sendo que as categorias são de apenas três tipos: arenito (S1), basalto (S2) e misto (S3). Já o tema vegetação divide a região geográfica em dois OGs-2D, sendo um de vegetação arbórea (V1) e outro de vegetação rasteira (V2). Ao superpor estes dois temas, obtém-se seis OGs-2D e cinco categorias, as quais são resultantes da interseção das categorias dos temas originais. As categorias formadas são: solo arenito e vegetação arbórea (S1/V1), solo basalto e vegetação arbórea (S2/V1), solo basalto e vegetação rasteira (S2/V2), solo misto e vegetação arbórea (S3/V1), e solo misto e vegetação rasteira (S3/V2).

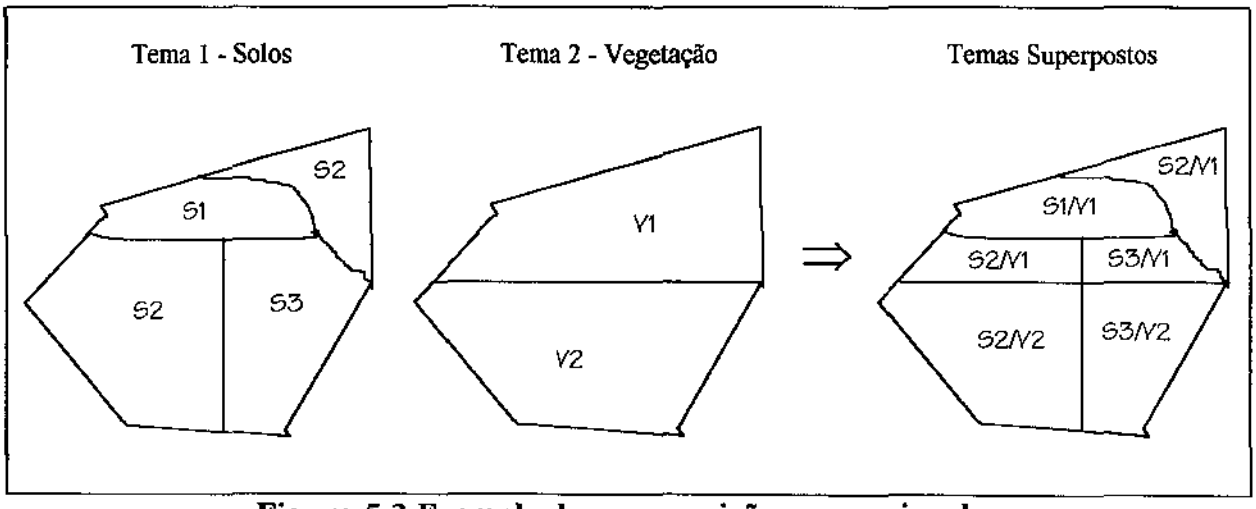


Figura 5.3 Exemplo de superposição convencional.

Muitas vezes, as categorias de um tema são formadas por números inteiros que indicam algum aspecto quantitativo, tais como número de pessoas, quantidade de espécies vegetais, entre outros. A superposição quando aplicada a temas cujas categorias são formadas por números permite a manipulação destas categorias através de operações aritméticas e relacionais, sendo chamada de superposição numérica. As principais operações efetuadas neste tipo de superposição são: adição, subtração, multiplicação, divisão, exponenciação, minimização e maximização [Sil94]. Na superposição numérica os temas preferencialmente devem ter os mesmos OGs-2D, uma vez que a superposição é feita somente para os OGs-2D comuns a ambos os temas. Assim, a subdivisão do tema superposto é igual em número e localização às subdivisões dos temas originais e portanto não existe nova distribuição de categorias, sendo as categorias resultantes iguais em número, e apenas desiguais em valor. O valor gerado para cada categoria no tema superposto indica uma quantificação referente à entidade lógica genérica (pessoa, espécie, etc) que os temas originais representam, sendo estes valores gerados pela operação aplicada na superposição.

A figura 5.4 ilustra a superposição de dois temas, dossel e subosque, utilizando-se as operações de adição, exponenciação e minimização. O tema dossel ilustra o número de espécies do dossel presentes em cada OG-2D, onde o termo dossel corresponde ao conjunto de árvores cujas copas ocupam o extrato mais alto da floresta, impedindo a passagem da luz solar. Já o tema subosque ilustra o número de espécies do subosque em cada OG-2D, sendo que por subosque entende-se as árvores que vivem na sombra (nascem, crescem e reproduzem), protegidas pelas espécies do dossel. A superposição por adição destes temas, por exemplo, produz o total de espécies arbóreas presentes em cada OG-2D. Neste exemplo, a superposição é efetuada da seguinte maneira: para cada OG-2D do tema superposto, o valor da categoria associada é obtida pela soma dos valores das categorias do dossel (tema 1) e do subosque (tema 2). Analogamente à adição, são as transações de superposição por subtração, multiplicação, divisão e exponenciação, sendo que nesta última as categorias do segundo tema consistem nas potências das categorias do primeiro tema. Na superposição por minimização, para cada OG-2D do tema superposto escolhe-se o menor valor entre as categorias dos temas originais para ser o valor da categoria associada. Analogamente, na superposição por maximização, escolhe-se o maior valor entre as categorias dos temas originais.

Outro tipo de superposição é chamada de superposição *booleana*. Este tipo de superposição consiste em superpor OGs-2D de dois temas, sendo que para cada OG-2D resultante a sua categoria é formada pela aplicação de uma operação *booleana* nas categorias dos temas originais. Típicas operações *booleanas* são: *And, Or, Xor e Not.* O valor das categorias dos OGs-2D dos temas originais são *0* ou 1, indicando respectivamente ausência e presença de alguma característica temática, tal como área de preservação. Isto é ilustrado na figura 5.5. Neste exemplo, quer-se encontrar a partir de dois temas, um descrevendo áreas de conservação e outro descrevendo áreas com jazidas de ouro, áreas indicadas à mineração de ouro. Para isto, a área deve obviamente ter potencial mineral – jazidas de ouro, e não ser de conservação. Isto é conseguido aplicando-se as operações *booleanas And* e *Not* conjuntamente – (*Not* T1) *And* T2.

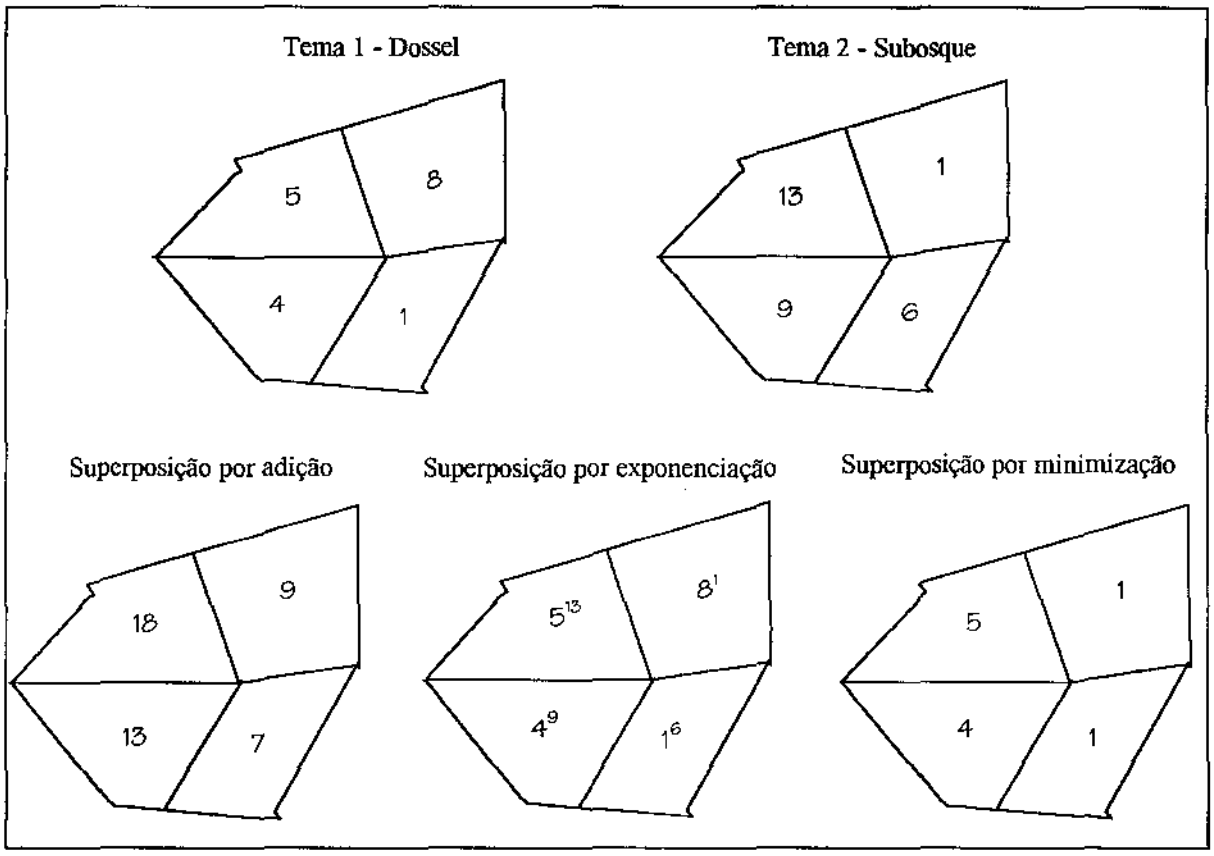


Figura 5.4 Superposição numérica.

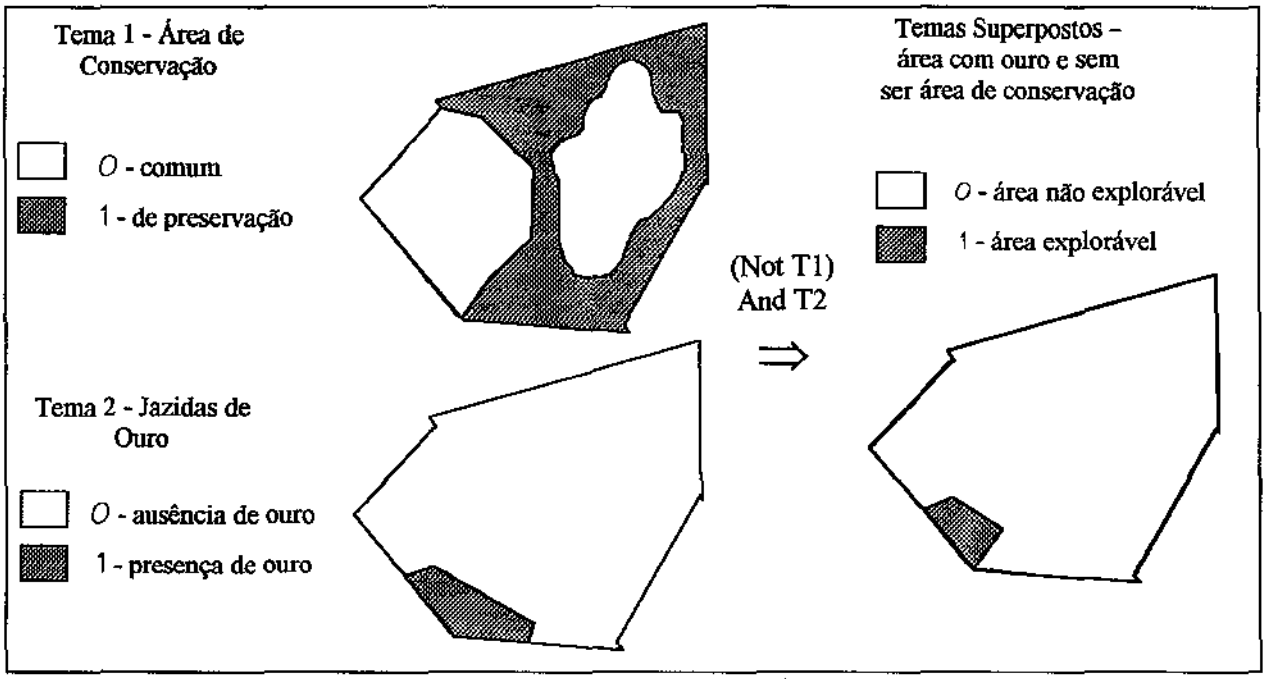


Figura 5.5 Superposição *booleana*.

Um quarto tipo de superposição é chamada de superposição *cookie cutter*. Neste tipo de superposição, um tema, composto de um conjunto de OGs-2D, é superposto por um segundo tema, formado por um conjunto de OGs-2D contidos na região representada pelo primeiro. Os OGs-2D do segundo tema, neste tipo de superposição, "encobrem" os OGs-2D do primeiro tema, gerando uma nova distribuição de OGs-2D que mantém, sem nenhuma alteração, os OGs-2D do segundo tema, rearranjando apenas os OGs-2D do primeiro tema. As categorias dos OGs-2D do tema superposto são as mesmas categorias encontradas nos temas originais, de acordo com a origem de cada uma destas.

Um exemplo deste tipo de superposição é mostrado na figura 5.6. Neste exemplo, um tema representando o zoneamento fiscal de uma cidade é superposto por um tema que representa novas subdivisões de zoneamento fiscal. O resultado desta transação é um conjunto de OGs-2D que mantém os OGs-2D do segundo tema, alterando somente os OGs-2D do tema original (zoneamento fiscal). O intuito deste tipo de superposição é alterar a distribuição original dos OGs-2D de modo que estes ocupem uma menor área de abrangência, sendo que novos OGs-2D irão abranger as áreas que deixaram de ser abrangidas pelos OGs-2D originais. No exemplo da figura 5.6, a instalação de novos escritórios de fiscalização faz com que haja um rearranjo na distribuição das zonas de fiscalização, uma vez que escritórios que antes tinham que atender áreas afastadas, agora não precisam mais, em vista que um novo escritório de fiscalização mais próximo foi implantado.

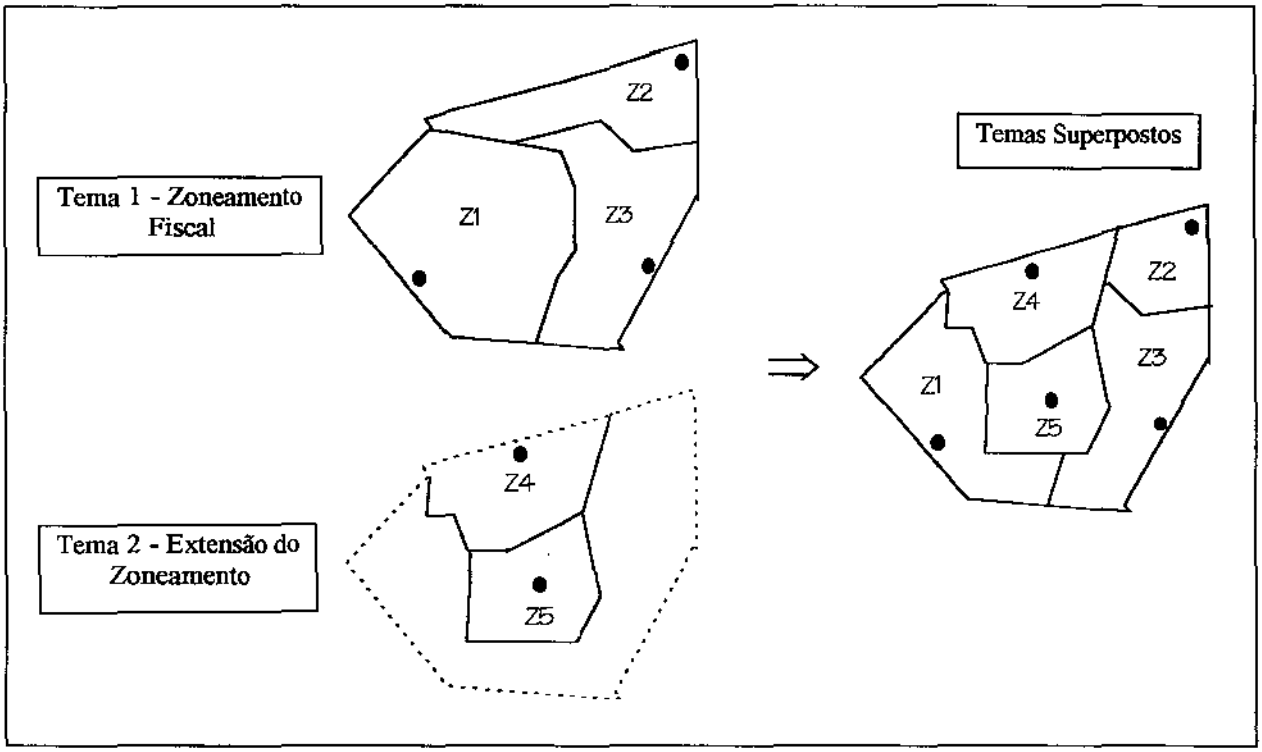


Figura 5.6 Exemplo de superposição *cookie cutter*.

5.2.3 Análise de ponderação

A análise de ponderação, ou simplesmente ponderação, é uma variante da superposição. Esta transação mistura aspectos relacionados à superposição convencional e à superposição numérica, diferindo principalmente destas em relação à geração das categorias dos OGs-2D do tema superposto. Esta geração é efetuada com base em pesos associados às categorias dos temas originais, ao invés de ser efetuada usando-se diretamente as próprias categorias deste temas. Existem dois tipos de ponderação: simples e tabelada. A ponderação simples considera as categorias dos temas originais como sendo parte de um conjunto único. Assim, para cada elemento deste conjunto é associado um peso. Em geral, para eliminar-se qualquer ambigüidade que impeça a determinação das categorias originais a partir dos valores gerados para as novas categorias, usa-se pesos derivados de potência de 2 não repetidos. Deste modo, para cada OG-2D do tema superposto, a categoria é obtida através de uma operação aritmética, freqüentemente uma adição, com os pesos das categorias dos OGs-2D correspondentes nos temas originais.

A figura 5.7 ilustra um exemplo de ponderação simples por adição, a qual utiliza pesos com potência de 2 não repetidos. Neste exemplo, dois temas são ponderados, declividade e vegetação, derivando um tema superposto que indica as áreas mais propícias à pecuária de bovinos. O tema declividade possui quatro OGs-2D e quatro categorias associadas: D1 – declividade ≥ 20%, D2 – 15% ≤ declividade < 20%, D3 – 10% ≤ declividade < 15% e D4 – declividade < 10%. Já o tema vegetação possui três OGs-2D e três categorias associadas: V1 – mata mista, V2 – plantação e V3 – pasto. Os dois principais fatores determinantes de áreas propícias à pecuária de bovinos são: vegetação de pasto e baixa declividade. Desta forma, uma seqüência de distribuição de pesos poderia ser: V3 (2^6), D4 (2^5), V2 (2^4), D3 (2^3), D2 (2^2), V1 (2^1), D1 (2^0). Os valores das categorias no tema superposto possuem uma relação de ordem crescente, na qual OGs-2D com um maior valor de categoria são considerados mais propícios à atividade em questão, neste caso à pecuária de bovinos.

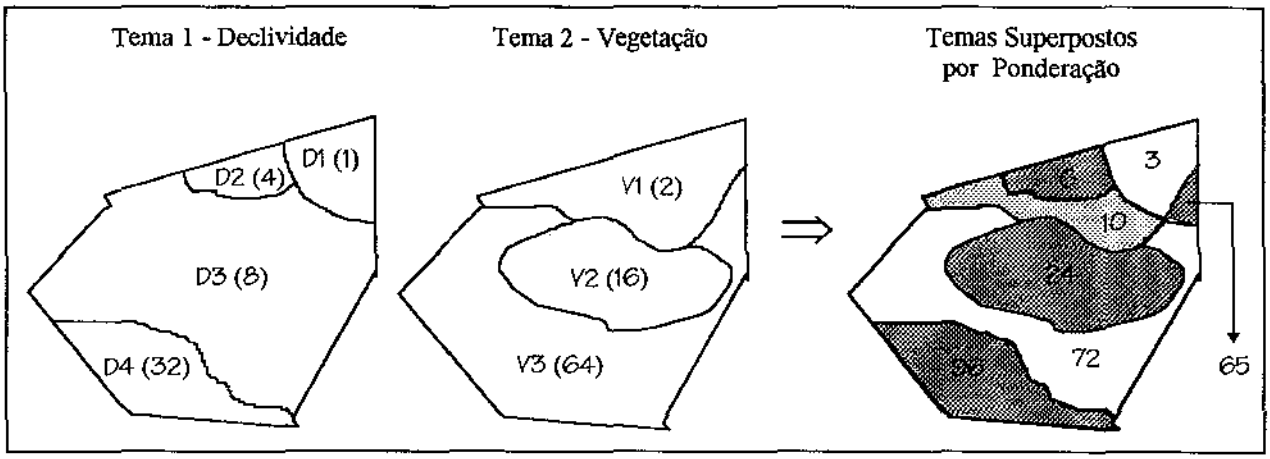


Figura 5.7 Ponderação simples por adição.

A ponderação tabelada atribui pesos aos relacionamentos entre as categorias dos temas originais, os quais são armazenados logicamente na forma de uma tabela. A categoria de cada OG-2D do tema superposto é obtida através de uma consulta à tabela de pesos em relação às categorias dos OGs-2D correspondentes nos temas originais. A figura 5.8 ilustra o mesmo exemplo descrito na figura 5.7, porém modificando-se a distribuição dos pesos. A tabela 5.1 mostra os pesos associados aos relacionamentos entre as categorias dos temas declividade e vegetação.

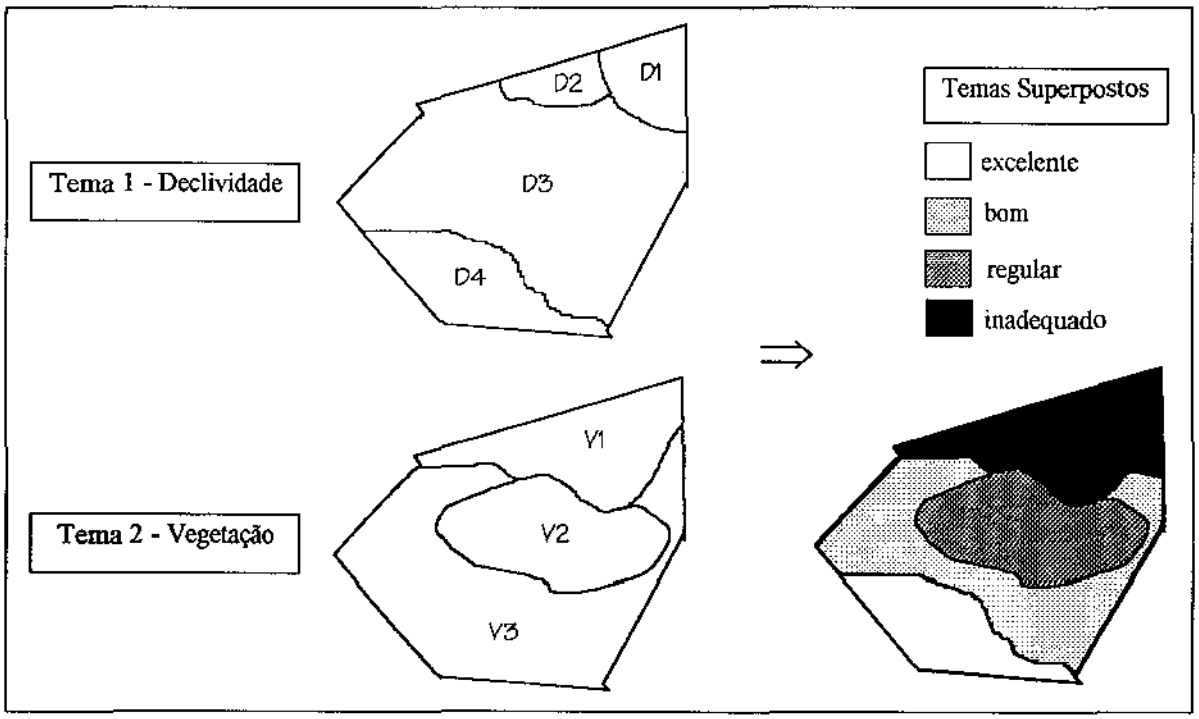


Figura 5.8 Ponderação tabelada.

veg. / decliv.	D1 (d ≥ 20)	D2 (15 ≤ d < 20)	D3 (10 ≤ d < 15)	D4 (d < 10)
V1 (mata mista)	inadequado	inadequado	inadequado	inadequado
V2 (plantação)	inadequado	inadequado	regular	regular
V3 (pasto)	inadequado	regular	bom	excelente

Tabela 5-1 Tabela de pesos dos relacionamentos entre as categorias dos temas vegetação e declividade.

5.2.4 Análise de proximidade

Juntamente com a superposição, a análise de proximidade é uma das transações mais efetuadas por aplicações georeferenciadas. O tipo mais básico, chamado de análise de proximidade simples, consiste em gerar um OG-2D (*buffer* simples, zona de *buffer* simples ou simplesmente *buffer*), na forma de um "corredor", cujos limites externos possuem uma distância fixa k em relação a um OG fonte, e cujos limites internos são formados pelos limites do próprio OG fonte a partir de onde a distância k começou a ser medida.

A figura 5.9 mostra uma zona de *buffer*, termo genérico usado para designar o OG-2D gerado. Neste clássico exemplo, uma zona de *buffer* simples de 1500 metros é gerada ao redor de um rio, representado logicamente através de um OG-1D. Assim, um órgão de defesa ambiental pode preservar a mata ciliar ao redor deste rio, proibindo o desmatamento dentro da zona de *buffer*. A escolha de locais para a construção de hospitais também pode ser facilitada com a análise de proximidade. Hospitais devem ser construídos de preferência ao redor de vias de acesso rápido. Deste modo, zonas de *buffer* ao redor das principais ruas e avenidas devem ser geradas para indicar possíveis áreas para a construção. Este último exemplo ilustra a geração de múltiplas zonas de *buffer* simples. Neste tipo de análise de proximidade, pode ocorrer interseção entre as diversas zonas de *buffer* simples geradas, sendo o resultado final a união de todas estas. A figura 5.10 ilustra a geração de múltiplas zonas de *buffer* simples ao redor de OGs-0D, com interseção ($k_1 = k_2 = \ldots = k_6$).

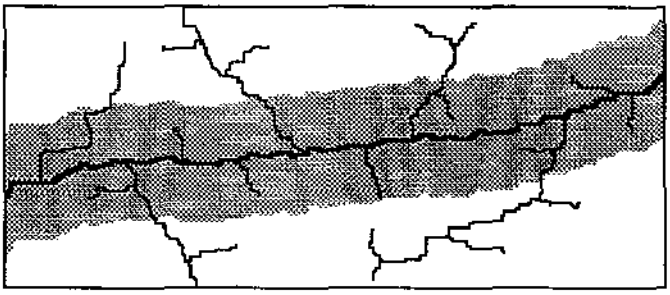


Figura 5.9 Exemplo de *buffer* simples.

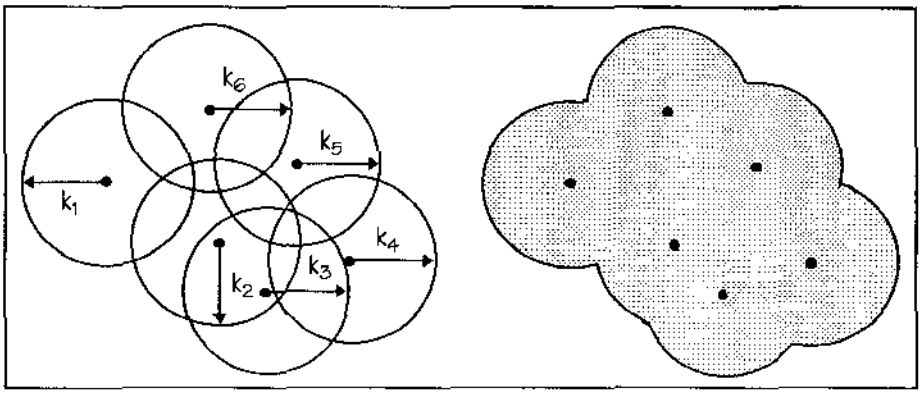


Figura 5.10 Múltiplas zonas de *buffer* simples.

A análise de proximidade ao redor de OGs-0D pode ser efetuada segundo duas abordagens: circular e quadrática. A primeira, tida como tradicional, gera os limites externos do *buffer* com uma distância fixa em relação a um OG-0D fonte. Já a segunda, uma variante, consiste em gerar os limites externos do *buffer* na forma de um quadrado, cuja mediatriz possui uma distância fixa em relação a um OG-0D (figura 5.11).

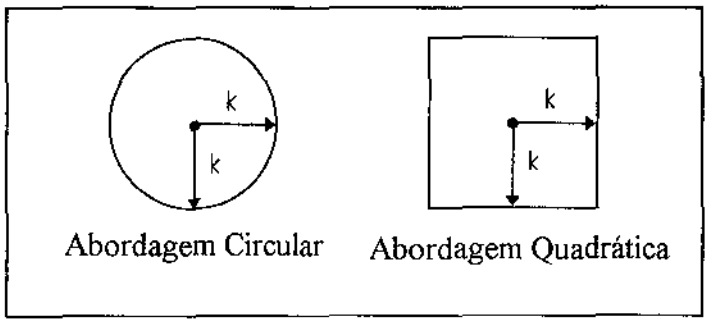


Figura 5.11 Análise de proximidade ao redor de OGs-0D.

A análise de proximidade ao redor de OGs-2D pode ser efetuada segundo duas abordagens: externa (tradicional), quando a zona de *buffer* gerada situa-se externamente ao OG-2D fonte, e interna, quando a zona de *buffer* gerada situa-se internamente ao OG-2D fonte (figura 5.12). Já a análise de proximidade ao redor de OGs-1D (figura 5.9) não possui variantes.

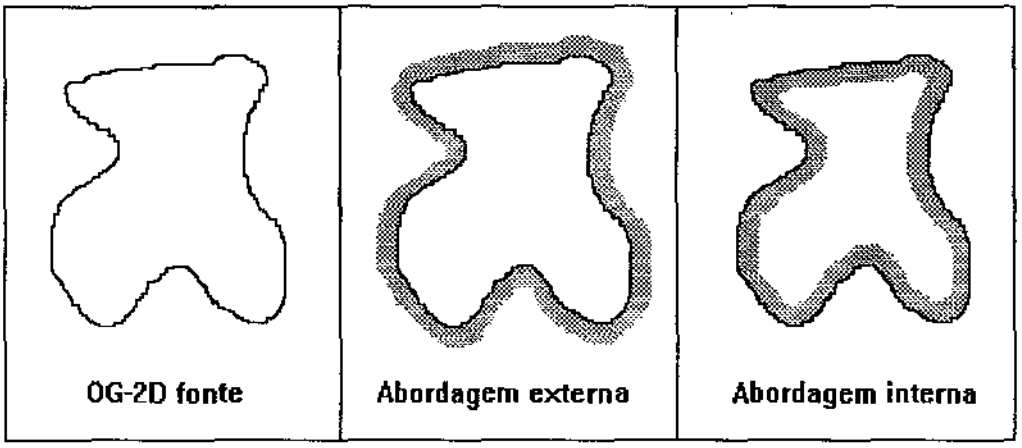


Figura 5.12 Análise de proximidade interna e externa ao redor de OGs-2D.

A análise de proximidade múltipla, uma generalização da análise de proximidade simples, consiste em gerar múltiplas zonas de *buffer* ao redor de um mesmo OG fonte, sendo que estas zonas de *buffer* possuem $k_1, k_2, \ldots, k_n$ unidades de distância, onde $k_n = k_{n-1} + d_n$, $k_{n-1} = k_{n-2} + d_{n-1}$, ..., $k_2 = k_1 + d_2$, $k_1 = d_1$. Em geral, $d_1 = d_2 = \ldots = d_n$, tornando a relação da seguinte forma: $k_i = i * d$, para $i \geq 1$. Cada zona de *buffer* gerada compreende o "corredor" formado entre os limites externos da zona de *buffer* anterior e o seu limite

externo, sendo que no caso da primeira zona de *buffer* o limite interno é formado pelo próprio limite do OG fonte (figura 5.13).

A análise de proximidade múltipla pode ser efetuada recursivamente utilizando-se a análise de proximidade simples. A primeira zona de *buffer* gerada pela análise de proximidade múltipla consiste na mesma zona de *buffer* gerada pela análise de proximidade simples, e portanto pode ser gerada por esta última. Para gerar a segunda zona de *buffer*, a análise de proximidade simples deve considerar o OG fonte como sendo a união (seção 5.2.10) do OG fonte original com a primeira zona de *buffer* produzida. Para gerar a terceira zona de *buffer* repete-se a idéia, onde a análise de proximidade simples considera o OG fonte como sendo a união do OG fonte original com a primeira e a segunda zona de *buffer* produzidas. Assim, para a geração da n-ésima zona de *buffer*, a partir da análise de proximidade simples, basta considerar o OG fonte como sendo a união do OG fonte original com as zonas de *buffer* n-1 até 1 até então produzidas.

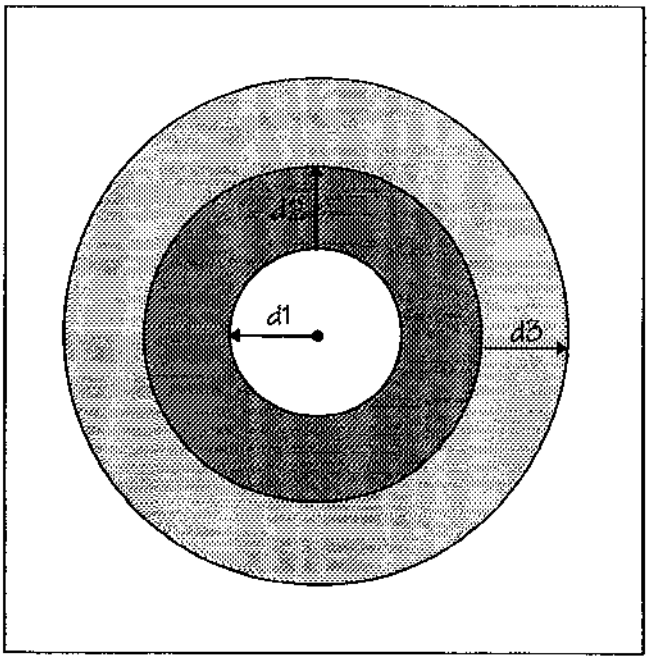


Figura 5.13 Análise de proximidade múltipla ao redor de um OG-0D.

A análise de proximidade possui uma maior precisão quando efetuada em dados vetoriais, uma vez que o cálculo da zona de *buffer* é feito usando-se dados que não possuem arredondamentos. Dados *raster*, ao contrário, geram arredondamentos, uma vez que trabalham com o conceito de célula, a menor unidade espacial representável, sendo qualquer cálculo efetuado com base nas coordenadas destas células. A figura 5.14 ilustra a mesma zona de *buffer* nos formatos de dados *raster* e vetorial[1].

[1] o tamanho da célula propositadamente foi escolhido de forma exagerada – para acentuar os erros.

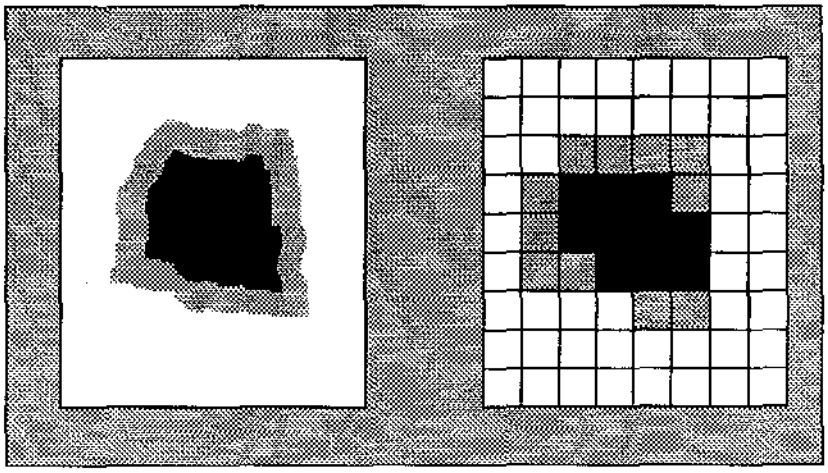


Figura 5.14 Zonas de *buffer* nos formatos *raster* e vetorial.

5.2.5 Decomposição de OGs-2D

Um OG-2D é composto por dois componentes básicos: seu interior e sua fronteira (figura 5.15). Muitas vezes quer-se saber se um dado OG-0D, por exemplo, está contido em um OG-2D, entretanto este não pode estar localizado na fronteira. Para isto, primeiramente deve-se obter o interior do OG-2D, para depois efetuar-se a transação topológica de inclusão (seção 5.2.6).

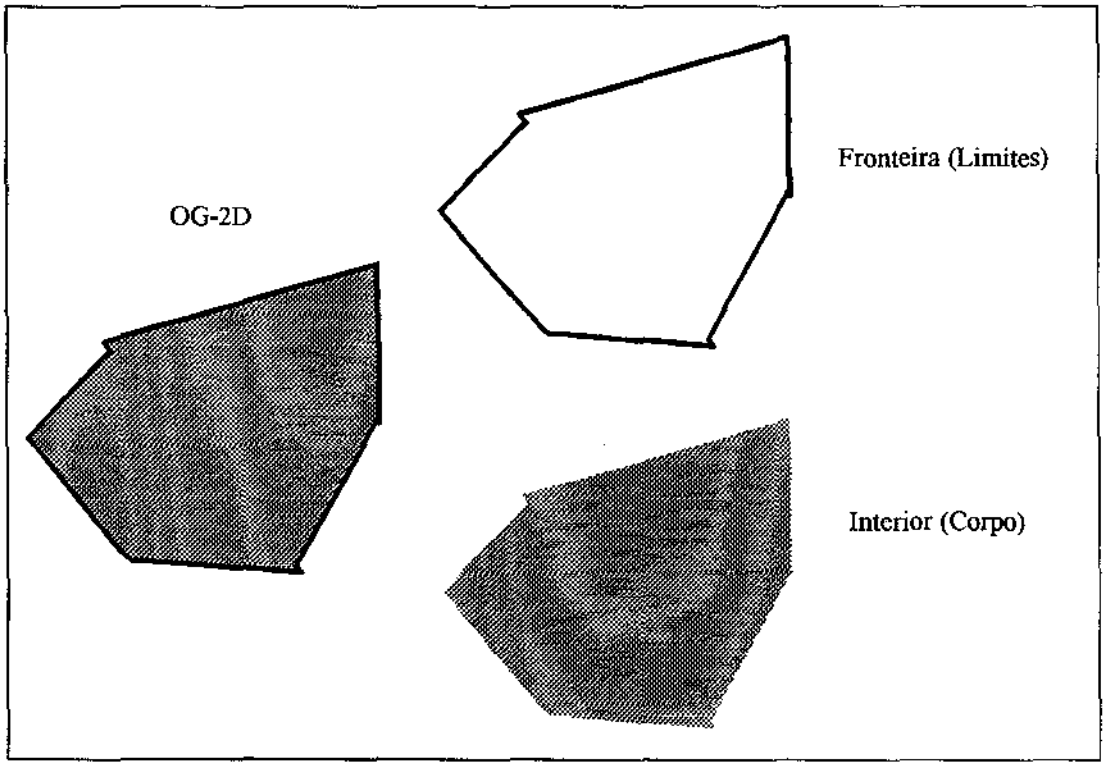


Figura 5.15 OG-2D: interior e fronteira.

5.2.6 Transações topológicas *booleanas*

Uma característica exclusiva de dados espaciais é a relação de topologia. Um relacionamento topológico é um relacionamento entre dois ou mais OGs, indicando relações de inclusão, adjacência, interseção, entre outras. Em geral, a determinação de relacionamentos topológicos requer métodos de acesso aos dados específicos, chamados de métodos de acessos espaciais, uma vez que métodos de acesso convencionais são voltados exclusivamente para determinar relacionamentos de ordenação alfanumérica entre os dados convencionais armazenados.

Transações topológicas *booleanas* são caracterizadas por retornar um valor lógico para indicar a ausência ou presença de um dado relacionamento topológico existente entre OGs. As principais transações topológicas *booleanas* são descritas a seguir (figura 5.16).

- **cruzamento:** verificar se OGs [não] atravessam (cruzam) um OG fonte.
- **interseção:** verificar se OGs [não] intersectam um OG fonte.
- **disjunção:** verificar se OGs [não] são disjuntos em relação a um OG fonte.
- **adjacência:** verificar se OGs [não] são adjacentes em relação a um OG fonte.
- **inclusão:** verificar se OGs [não] estão contidos em um OG fonte.
- **igualdade:** verificar se OGs são iguais em forma (geometria).

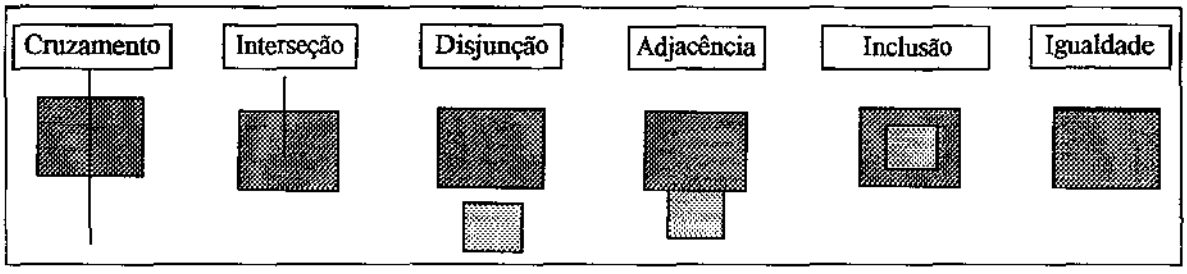


Figura 5.16 Principais relações topológicas.

5.2.7 Transações de busca topológica

A determinação de relacionamentos topológicos entre OGs é útil e praticável somente quando se conhece os OGs em questão. Muitas vezes, o que se deseja obter é o conjunto de OGs que satisfaçam um certo relacionamento topológico em relação a um OG fonte. Transações que efetuam isto são chamadas de transações de busca topológica. Estas transações utilizam métodos de acesso espaciais para determinar os OGs que satisfazem um dado relacionamento topológico, constituindo-se de transações especiais, devido à

enorme quantidade de dados espaciais armazenados em aplicações georeferenciadas. Um método de acesso espacial ineficiente pode comprometer significativamente o desempenho de transações deste tipo. Além disto, um método de acesso espacial geralmente não é próprio para todos os tipos de relacionamentos topológicos.

Em geral, para cada transação topológica *booleana* pode-se derivar uma transação de busca topológica equivalente, e vice-versa. Entretanto, algumas transações de busca topológica derivadas de transações topológicas *booleanas* perdem a importância semântica, tornando-se não usuais (e vice-versa). A seguir são descritas somente as principais transações de busca topológica.

- determinação do OG mais próximo de um OG fonte.
- determinação dos OGs que [não] são adjacentes a um OG fonte.
- determinação dos OGs que [não] estão contidos em um OG fonte.
- determinação dos OGs que [não] contêm um OG fonte.
- determinação dos OGs que [não] intersectam um OG fonte.
- determinação dos OGs que [não] atravessam um OG fonte.

A figura 5.17 ilustra a determinação dos OGs-0D que estão contidos em um OG-2D fonte. Esta transação exemplifica a necessidade de ter-se métodos de acesso espaciais eficientes. Neste exemplo, quer-se encontrar os criadouros do mosquito da dengue que estão contidos no bairro Ipiranga. Vê-se claramente a rara existência destes criadouros neste bairro, ao contrário do que ocorre no bairro adjacente. Caso o método de acesso não possua um meio de guardar o relacionamento topológico de inclusão, este perderá muito tempo analisando criadouros que não pertencem ao bairro Ipiranga.

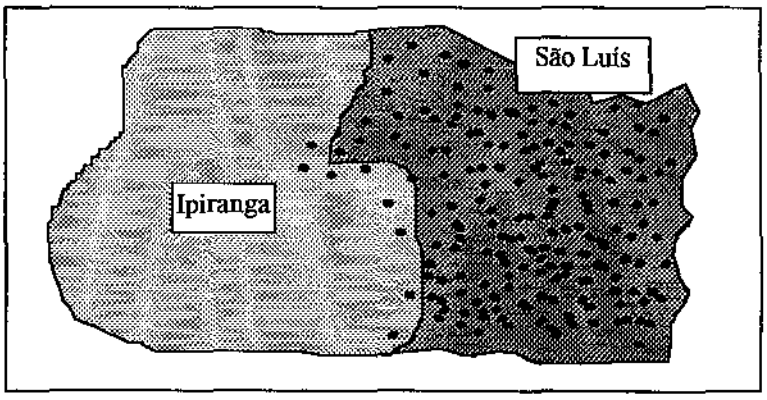


Figura 5.17 Exemplo de transação de busca topológica.

5.2.8 Transações topológicas escalares

Estas transações determinam as coordenadas geográficas onde ocorre um determinado relacionamento topológico. Transações topológicas escalares típicas são a interseção e o cruzamento. Assim, pode-se determinar, por exemplo, os pontos de interseção de um OG-2D com vários OGs-1D. Isto é ilustrado na figura 5.18. Neste exemplo, enquanto o rio das Cabras possui dois pontos de interseção com a região em questão, o rio São Lourenzo possui apenas um e o rio Biritiba não intersecta a fronteira da região. Através deste exemplo, distingue-se intuitivamente os conceitos de interseção e cruzamento. O rio das Cabras intersecta e cruza a região em questão. Já o rio São Lourenzo apenas intersecta a mesma.

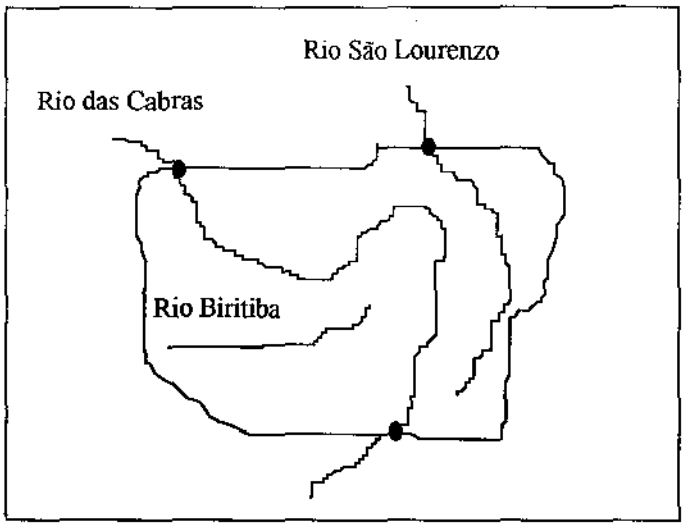


Figura 5.18 Exemplo de transação topológica escalar.

5.2.9 Transações escalares

Transações escalares possuem em comum o fato de, a partir de um OG, gerar um dado escalar que representa uma propriedade intrínseca ao OG analisado. Estas transações são baseadas principalmente no cálculo de área, distância e comprimento. Assim, por exemplo, pode-se saber a área ou o perímetro de um OG-2D. Em alguns sistemas, o resultado de transações deste tipo são pré-computados e armazenados de modo a diminuir o tempo de resposta das transações, pois para se obter estes resultados é necessário primeiramente o acesso aos dados espaciais do OG, e em seguida realizar um certo conjunto de cálculos matemáticos com estes dados [Ele94]. Entretanto, devido à grande variedade de propriedades intrínsecas, além da grande quantidade de OGs armazenados, torna-se inviável o armazenamento de muitas destas propriedades, sendo que em geral estas são quantificadas dinamicamente.

As principais transações escalares são:

- cálculo da distância mínima entre dois OGs.
- cálculo da área de um OG-2D.
- cálculo do perímetro de um OG-2D.
- cálculo do comprimento de um OG-1D.

5.2.10 Transações baseadas em conjunto

Estas transações possuem em comum o fato de retornarem um conjunto de OGs baseadas em operações matemáticas de conjunto. As principais transações deste tipo são: união, interseção e diferença. A figura 5.19 mostra a aplicação destas três transações em um par de OGs-2D que se intersectam. A transação de união retorna um conjunto de OGs resultante da união dos limites de vários OGs. Esta transação também é ilustrada na figura 5.20. Neste exemplo, quatro OGs distintos, nomeados de A a D, são unidos. O resultado desta transação é um conjunto de dois OGs nomeados X e Y. Nem sempre, como visto neste exemplo, o resultado da união de vários OGs resulta em um único OG. As transações de interseção e diferença, de modo semelhante, retornam um conjunto de OGs resultante, respectivamente, da interseção e da diferença dos limites de vários OGs.

Transações baseadas em conjunto são generalizações da superposição *booleana*. Nestas últimas, os OGs envolvidos devem ser necessariamente bidimensionais, envolvendo a junção de temas através de suas categorias. Já transações baseadas em conjunto podem ser aplicadas a qualquer dimensão, além de não se ter nenhum relacionamento obrigatório com dados temáticos.

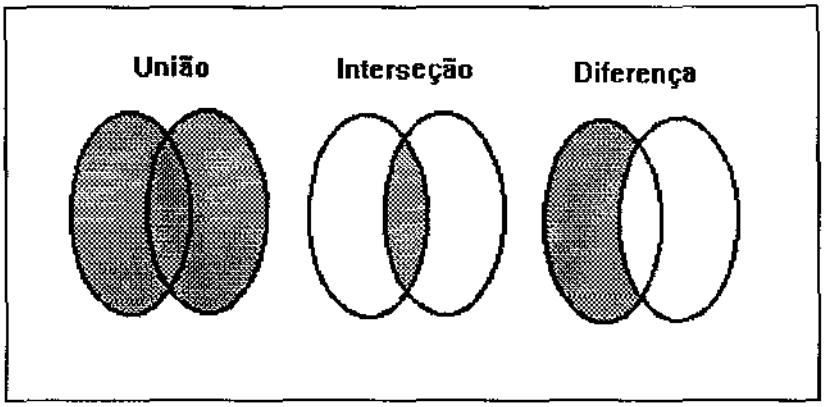


Figura 5.19 Transações de união, interseção e diferença de conjuntos.

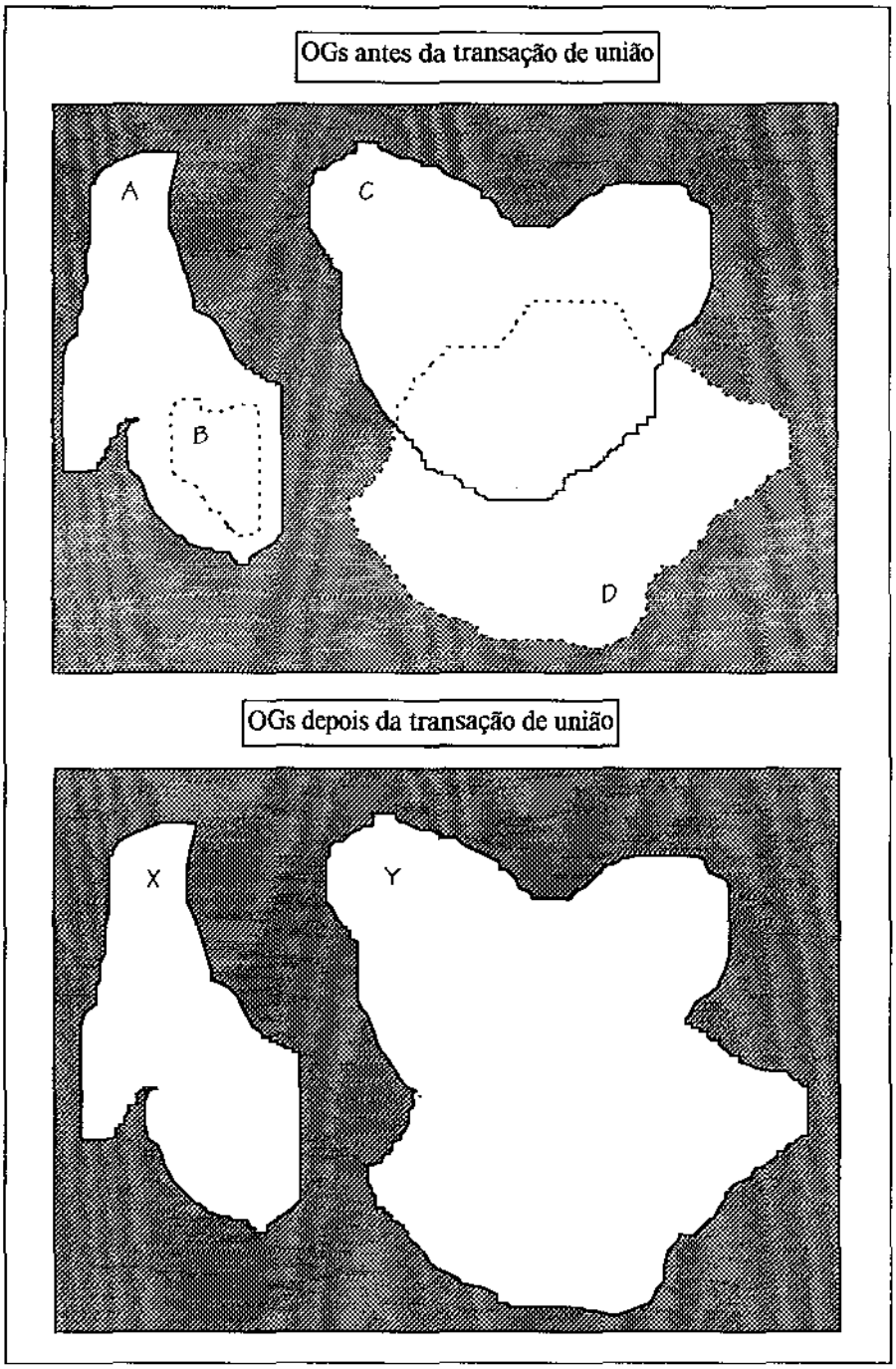


Figura 5.20 Exemplo de transação de união.

5.2.11 Transações de conversão de formato de dados

Uma importante funcionalidade a ser provida por SIGs é a conversão de formato de dados. Sistemas de informações geográficas trabalham geralmente com os formatos *raster* e vetorial. Muitas vezes, entretanto, uma transação é implementada em apenas um destes formatos, sendo necessário converter os dados entre estes dois formatos.

A conversão do formato vetorial para o formato *raster* é conceitualmente simples [Sil94], desde que não ocorra o problema de competição por localização (seção 2.1). No caso de pontos, a célula correspondente no formato *raster* englobará as coordenadas geográficas deste. No caso de linhas, a estratégia mais simples de conversão consiste em identificar as células que são cortadas por uma linha, e posteriormente codificar estas células com os atributos associados à linha. Em relação às linhas que não são paralelas aos eixos de orientação, a representação *raster* mostra uma distorção chamada escada (figura 5.21). Esta distorção pode ser minimizada usando-se dois procedimentos: a diminuição da resolução espacial, e a modificação do contraste das células adjacentes. Os polígonos são convertidos no formato *raster* em dois estágios. Primeiro os segmentos de linha que formam os limites do polígono são convertidos no formato *raster*, tal como foi descrito acima, produzindo o que se chama de esqueleto do polígono. A seguir, os elementos *raster* contidos pelos limites do polígono são registrados com a categoria do polígono. A competição por localização é um problema grave encontrado durante a conversão de dados do formato vetorial para o formato *raster*. Pontos, linhas e polígonos podem, durante esta conversão, ocuparem as mesmas células. Neste caso, o SIG deve fornecer regras que determinem para cada célula disputada qual o elemento que ela representará. Em relação a temas, a conversão do formato vetorial para *raster* é efetuada somente com polígonos que não possuem nenhum relacionamento topológico de inclusão entre si, evitando-se o problema da disputa de localização.

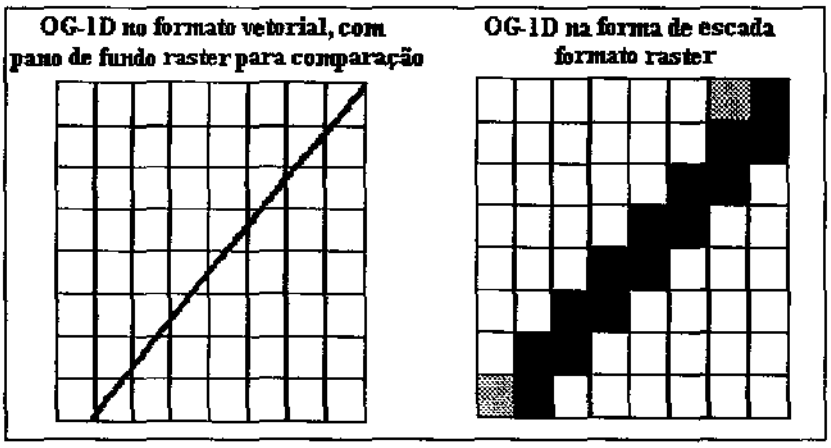


Figura 5.21 Escada.

A conversão do formato *raster* para o formato vetorial é bem mais complexa. Os arredondamentos proporcionados pelas células impedem uma precisão na definição das coordenadas de pontos, linhas e polígonos. Além disto, uma vez efetuada a conversão inversa, vetorial para *raster*, todos os elementos descartados durante o problema da disputa de localização não podem mais ser recuperados.

SIGs também fazem conversão entre os formatos *raster* e vetorial e formatos específicos. Exemplos destes últimos formatos são: *DXF* (*Drawing eXchange Format*) e *PostScript*. Entretanto, conversões deste tipo não serão tratadas neste trabalho, uma vez

que existem inúmeros formatos existentes, sendo a adoção de uma padrão único ainda inexistente.

5.2.12 Transações que envolvem visualização gráfica

A visualização e manipulação gráfica de OGs é imprescindível para o usuário final de um SIG. Dentro deste contexto, as principais transações efetuadas são:

- visualização gráfica de OGs de qualquer dimensão.
- mudança do potencial de visualização, mais conhecida como *zoom*.
- traslado da área de visualização, mais conhecida como *pan*.
- superposição apenas visual de dados no formato *raster* e dados no formato vetorial, de um mesmo conjunto de OGs.

5.2.13 Transações específicas de dados no formato *raster*

Algumas transações são específicas de dados no formato *raster*. Isto ocorre devido à natureza particular de armazenamento em células providas por este formato. As principais transações são descritas a seguir.

- **agrupamento:** junção de um conjunto de células de mesma categoria, adjacentes entre si, para formar OGs. A figura 5.22 ilustra esta transação. Neste exemplo, são formados sete OGs, sendo três OGs-2D, um OG-1D e três OGs-0D. Vale destacar que uma categoria, como a número 1 da matriz de células original, pode originar vários OGs.

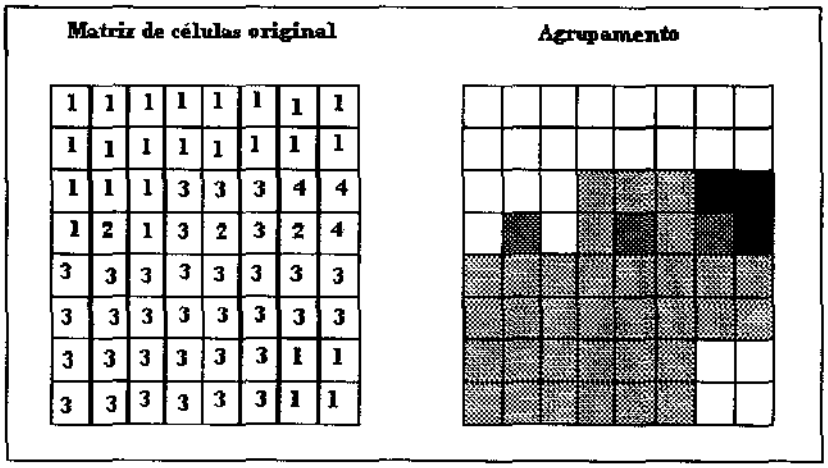


Figura 5.22 Exemplo de agrupamento.

- **mudança da resolução espacial:** deve-se diferenciar a mudança da resolução espacial de *zoom*, descrito anteriormente. O *zoom* apenas altera o tamanho das células na tela, enquanto a mudança da resolução espacial modifica o tamanho de armazenamento das células. Esta transação só pode ser efetuada de forma crescente, ou seja, pode-se aumentar a resolução espacial, mas não é possível diminuí-la, uma vez que não é possível reconstituir os valores das categorias das células resultantes da divisão de uma célula original. Desta forma, um conjunto de células é agrupado em uma única célula, após o aumento da resolução espacial, sendo a categoria desta célula a categoria dominante entre as células agrupadas (figura 5.23).

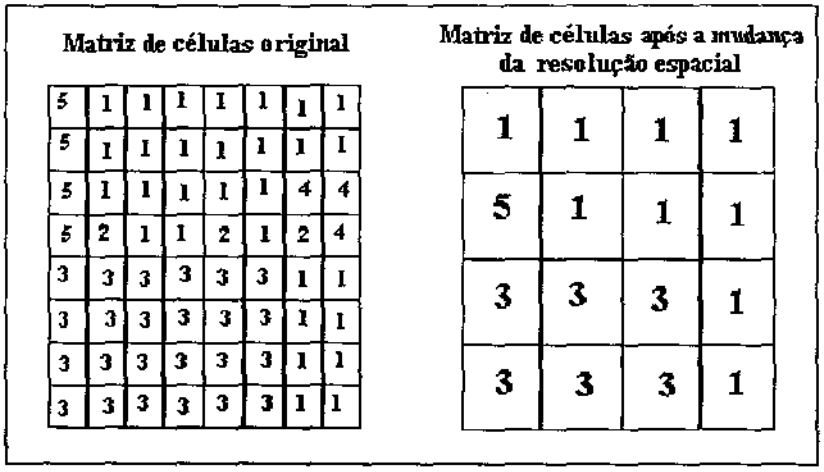


Figura 5.23 Exemplo de mudança de resolução espacial.

- **manipulação de matrizes de células:** célula com menor/maior valor de categoria, número de células, categoria média das células e comparação de matrizes de células. Esta última transação visa verificar à diferença de conteúdo de uma matriz produzida a partir de uma observação em solo e de uma matriz produzida por sensoriamento remoto, por exemplo.

5.2.14 Transações diversas

A seguir são descritas algumas outras transações comumente realizadas.

- determinação das coordenadas geográficas de um conjunto de OGs.

- geração de um OG-1D a partir de dois OGs-0D.

- carga do sistema, ou seja, inicialização do SIG, a qual inclui a construção de índices espaciais e convencionais.

- seleção de OGs a partir dos valores de seus atributos convencionais, denominada de seleção convencional.

- armazenamento de OGs em memória secundária.

5.3 Transações primitivas

5.3.1 Transações primitivas genéricas

Reclassificação

Reclass (O: Set (OG), NewCategories: TypeNC) : Set (OG)

Parâmetros
O = conjunto de OGs de um dado tema a ser reclassificado.
NewCategories = relacionamento entre as novas e as antigas categorias.

Tipos
TypeNC = $\{(Newcat_1 , Set_1 (Oldcat)), \ldots , (Newcat_p , Set_n (Oldcat))\}$
onde $Set_1 (Oldcat) \cap \ldots \cap Set_n (Oldcat) = \varnothing$
$Set_1 (Oldcat) \cup \ldots \cup Set_n (Oldcat) = \{Oldcat_1 , \ldots , Oldcat_k\}$, k = última categoria
$\{Oldcat_1 , \ldots , Oldcat_k\}$ = conjunto de categorias do tema original
$\{Newcat_1 , \ldots , Newcat_p\}$ = conjunto de categorias do tema reclassificado

Exemplo - figura 5.2
Mapa temático original
OGs: O = $\{O_1 , O_2 , O_3 , O_4 , O_5 , O_6\}$
Categorias: C = $\{S_1 , S_2 , S_3 , S_4 , S_5\}$

Reclass (O, $\{(C_1 , \{S_1 , S_3 , S_4\}), (C_2 , \{S_2 , S_5\})\}$)

Mapa temático reclassificado
OGs: O' = $\{O_1', O_2'\}$
Categorias: C' = $\{C_1 , C_2\}$

Superposição

Overlay ([Not] O1, [Not] O2: Set (OG-2D), Operation: TypeOp) : Set (OG-2D)

Parâmetros
O1 = conjunto de OGs-2D do tema 1 a ser superposto.
O2 = conjunto de OGs-2D do tema 2 a ser superposto.
Operation = operação a ser aplicada na superposição.

Tipos
Operation = {Conventional | Add | Multiply | Exponentiate | Minimize | And | Or | Cutter}
onde Conventional = superposição convencional
Add = superposição numérica por adição
Multiply = superposição numérica por multiplicação
Exponentiate = superposição numérica por exponenciação
Minimize = superposição numérica por minimização
And = superposição *booleana And*
Or = superposição *booleana Or*
Cutter = superposição *cookie cutter*

Comentários Adicionais
A operação *booleana Not* é opcional a ambos os parâmetros O1 e O2, sendo representada por *[Not]*. Esta operação complementa os valores de cada categoria, ou seja, um valor de categoria 0 é mudado para 1 e vice-versa. Entretanto, a operação *booleana Not* deve ser usada somente se a operação aplicada na superposição também for *booleana* (*And* ou *Or*).

Análise de Ponderação

Weigh (O1, O2: Set (OG-2D), Operation: TypeOp, Weight: TypeW) : Set (OG-2D)

Parâmetros
O1 = conjunto de OGs-2D do tema 1 a ser ponderado.
O2 = conjunto de OGs-2D do tema 2 a ser ponderado.
Operation = operação a ser aplicada na ponderação.
Weight = pesos associados às categorias ou aos relacionamentos existentes entre estas.

Tipos
Operation = {Add | Table}
onde Add = ponderação simples por adição, Table = ponderação tabelada

Weight = $\{(cat_1, weight_1), \ldots, (cat_n, weight_n)\}$ para Operation = Add
Weight = $\{(cat_{11}, cat_{21}, weight_1), (cat_{12}, cat_{21}, weight_2), \ldots, (cat_{1n}, cat_{2m}, weight_{n*m})\}$
para Operation = Table

Análise de proximidade simples

SimpleBuffer (O: OG, Distance: Float, Additional: TypeAd) : OG-2D

Parâmetros
O = OG fonte a partir de onde será gerada a zona de *buffer* simples.
Distance = distância (em metros) que terá o limite externo do *buffer* em relação ao OG fonte.
Additional = caracteriza os tipos específicos de *buffer*.

Tipos
TypeAd = {Interior, Exterior, Circular, Square, Null}
onde Interior = abordagem interna (OG-2D).
Exterior = abordagem externa (OG-2D).
Both = abordagem interna / externa (OG-2D).
Circular = abordagem circular (OGs-0D).
Square = abordagem quadrática (OGs-0D).
Null = para OG-1D, semelhante a Exterior

Comentários adicionais
Esta transação primitiva gera uma zona de *buffer* simples ao redor de um OG.

Análise de proximidade simples ao redor de múltiplos OGs

SimpleBufferMultiple (O: Set (OG), Distance: Float) : Set (OG-2D)

Parâmetros
O = conjunto de OGs fonte a partir de onde serão geradas as várias zonas de *buffer* simples.
Distance = distância (em metros) que terá o limite externo das zonas de *buffer* em relação ao respectivo OG fonte.

Composição lógica

$O = \{O_1, \ldots O_n\}$
Simple*Buffer* (O_1, *Distance*, x) $\Rightarrow$ *Buffer*$_1$
Simple*Buffer* (O_2, *Distance*, x) $\Rightarrow$ *Buffer*$_2$
....
Simple*Buffer* (O_n, *Distance*, x) $\Rightarrow$ *Buffer*$_n$
Union (*Buffer*$_1$, *Buffer*$_2$, ... , *Buffer*$_n$) $\Rightarrow$ *Buffer* Final

Análise de proximidade simples ao redor de múltiplos OGs (continuação)

Comentários adicionais
Esta transação primitiva gera zonas de *buffer* simples ao redor de múltiplos OGs. Zonas de *buffer* que se intersectam são unidas em apenas uma zona de *buffer*. Em relação a OGs-0D utiliza-se a abordagem circular. Em relação a OGs-2D utiliza-se a abordagem externa.

Análise de proximidade múltipla

MultipleBuffer (O: OG, Distance: Float, Levels: Integer, Additional: TypeAd) : Set (OG-2D)

Parâmetros
O = OG fonte a partir de onde será gerada a zona de *buffer* múltipla.
Distance = distância (em metros) que terá o limite externo da zona de *buffer* múltipla em relação ao OG fonte.
Levels = número de níveis que terá a zona de *buffer* múltipla
Additional = especifica tipos específicos de *buffer*.

Tipos
TypeAd = {Interior, Exterior, Circular, Square, Null}
onde Interior = abordagem interna (OG-2D).
Exterior = abordagem externa (OG-2D).
Circular = abordagem circular (OGs-0D).
Both = abordagem interna / externa (OG-2D).
Square = abordagem quadrática (OGs-0D).
Null = para OG-1D, semelhante a Exterior

Composição lógica

$sourceobj_1 = O$
SimpleBuffer ($sourceobj_1$, *Distance*, *Additional*) $\Rightarrow$ $Buffer_1$
Union ($sourceobj_1$, $Buffer_1$) $\Rightarrow$ $sourceobj_2$
SimpleBuffer ($sourceobj_2$, *Distance*, *Additional*) $\Rightarrow$ $Buffer_2$
Union ($sourceobj_2$, $Buffer_2$) $\Rightarrow$ $sourceobj_3$
....
SimpleBuffer ($sourceobj_{n-1}$, *Distance*, *Additional*) $\Rightarrow$ $Buffer_{n-1}$
Union ($sourceobj_{n-1}$, $Buffer_{n-1}$) $\Rightarrow$ $sourceobj_n$
SimpleBuffer ($sourceobj_n$, *Distance*, *Additional*) $\Rightarrow$ $Buffer_n$

Comentários adicionais
Esta transação primitiva gera uma zona de *buffer* múltipla ao redor de um OG.

Análise de proximidade múltipla ao redor de múltiplos OGs

MultipleBufferMultiple (O: Set(OG), Distance: Float, Levels: Integer) : Set (OG-2D)

Parâmetros

O = conjunto de OGs fonte a partir de onde serão gerada as várias zonas de *buffer* múltipla.

Distance = distância (em metros) que terá o limite externo das zonas de *buffer* em relação aos respectivos OGs fonte.

Levels = número de níveis que terá a zona de *buffer* múltipla.

Composição lógica

Nomenclatura: $sourceobjs_{[m,n]}$, onde m = número do objeto, n = nível

Inicialmente k_1 OGs = $sourceobjs_1 = \{sourceobj_{[1,1]}, \ldots, sourceobj_{[k1,1]}\} = O$

Nível 1

para $i = 1$ até k_1

faça *SimpleBuffer* $(sourceobj_{[i,1]}, Distance, x) \Rightarrow Buffer_{[i,1]}$

Union $(Buffer_{[1,1]}, Buffer_{[2,1]}, \ldots, Buffer_{[k1,1]}) \Rightarrow BufferNivel_1$

Union $(BufferNivel_1, sourceobj_{[1,1]}, \ldots, sourceobj_{[k1,1]}) \Rightarrow sourceobjs_2$

onde $sourceobjs_2 = \{sourceobj_{[1,2]}, \ldots, sourceobj_{[k2,2]}\}$ e $k_2 \leq k_1$

Nível 2

para $i = 1$ até k_2

faça *SimpleBuffer* $(sourceobj_{[i,2]}, Distance, x) \Rightarrow Buffer_{[i,2]}$

Union $(Buffer_{[1,2]}, Buffer_{[2,2]}, \ldots, Buffer_{[k2,2]}) \Rightarrow BufferNivel_2$

Union $(BufferNivel_2, sourceobj_{[1,2]}, \ldots, sourceobj_{[k2,2]}) \Rightarrow sourceobjs_3$

onde $sourceobjs_3 = \{sourceobj_{[1,3]}, \ldots, sourceobj_{[k3,3]}\}$ e $k_3 \leq k_2$

.....

Nível n

para $i = 1$ até k_n

faça *SimpleBuffer* $(sourceobjs_{[i,n]}, Distance, x) \Rightarrow Buffer_{[i,n]}$

Union $(Buffer_{[1,n]}, Buffer_{[2,n]}, \ldots, Buffer_{[kn,n]}) \Rightarrow BufferNivel_n$

Comentários adicionais

Esta transação primitiva gera um *buffer* múltiplo ao redor de múltiplos OGs. Zonas de *buffer* que se intersectam são unidas em apenas uma zona de *buffer*. Em relação a OGs-0D utiliza-se a abordagem circular. Em relação a OGs-2D utiliza-se a abordagem externa.

Interior de um OG-2D

GetInterior (O: OG-2D) : OG-2D

Parâmetros
O = OG-2D fonte que quer-se extrair o interior.

Fronteira de um OG-2D

GetBoundary (O: OG-2D) : OG-1D

Parâmetros
O = OG-2D fonte que quer-se extrair a fronteira.

5.3.2 Transações primitivas topológicas *booleanas*

Adjacência

Meet (O1: OG, O2: Set (OG)) : Boolean

Parâmetros
O1 = OG fonte a partir do qual será baseada a relação de adjacência.
O2 = conjunto de OGs a partir dos quais quer-se verificar a relação de adjacência em relação ao OG fonte.

Comentários adicionais
Esta transação primitiva verifica se todos os OGs contidos no conjunto O2 são adjacentes em relação ao OG fonte. Geralmente, ambos O1 e O2 são constituídos de OGs-2D.

Não Adjacência

NotMeet (O1: OG, O2: Set (OG)) : Boolean

Parâmetros
O1 = OG fonte a partir do qual será baseada a relação de adjacência.
O2 = conjunto de OGs a partir dos quais quer-se verificar a relação de adjacência em relação ao OG fonte.

Não Adjacência (continuação)

Comentários adicionais
Esta transação primitiva verifica se todos os OGs contidos no conjunto O2 não são adjacentes em relação ao OG fonte. Geralmente, ambos O1 e O2 são compostos de OGs-2D.

Inclusão

Cover (O1: OG, O2: Set (OG), Boundary: Boolean) : Boolean

Parâmetros
O1 = OG fonte a partir do qual será baseada a relação de inclusão.
O2 = conjunto de OGs a partir dos quais quer-se verificar a relação de inclusão em relação ao OG fonte.
Boundary = indica, se verdadeiro, a inclusão da fronteira do OG fonte. Relevante apenas para OGs-2D.

Comentários adicionais
Esta transação primitiva verifica se todos os OGs do conjunto O2 estão contidos no OG fonte, considerando-se ou não a fronteira deste.

Não Inclusão

NotCover (O1: OG, O2: Set (OG), Boundary: Boolean) : Boolean

Parâmetros
O1 = OG fonte a partir do qual será baseada a relação de inclusão.
O2 = conjunto de OGs a partir dos quais quer-se verificar a relação de inclusão em relação ao OG fonte.
Boundary = indica, se verdadeiro, a inclusão da fronteira do OG fonte. Relevante apenas para OGs-2D.

Comentários adicionais
Esta transação primitiva verifica se todos os OGs do conjunto O2 não estão contidos no OG fonte, considerando-se ou não a fronteira deste.

Igualdade

Equal (O1: Set (OG)) : Boolean

Parâmetros
O1 = conjunto de OGs a partir dos quais quer-se verificar a relação de igualdade.

Igualdade (continuação)

Comentários adicionais
Esta transação primitiva verifica se todos os OGs do conjunto O1 são iguais em forma.

Não Igualdade

NotEqual (O1: Set (OG)) : Boolean

Parâmetros
O1 = conjunto de OGs a partir dos quais quer-se verificar a relação de igualdade.

Comentários adicionais
Esta transação primitiva verifica se todos os OGs do conjunto O1 não são iguais em forma.

Interseção

Overlap (O1: OG, O2: Set (OG)) : Boolean

Parâmetros
O1 = OG fonte a partir do qual será baseada a relação de interseção.
O2 = conjunto de OGs a partir dos quais quer-se verificar a relação de interseção em relação ao OG fonte.

Comentários adicionais
Esta transação primitiva verifica se todos os OGs do conjunto O2 intersectam o OG fonte.

Disjunção – Não Interseção

Disjoint (O1: OG, O2: Set (OG)) : Boolean
NotOverlap (O1: OG, O2: Set (OG)) : Boolean

Parâmetros
O1 = OG fonte a partir do qual será baseada a relação de disjunção.
O2 = conjunto de OGs a partir dos quais quer-se verificar a relação de disjunção em relação ao OG fonte.

Comentários adicionais
Esta transação primitiva verifica se todos os OGs contidos no conjunto O2 são disjuntos (não intersectam) em relação ao OG fonte.

Cruzamento

Span (O1: OG, O2: Set (OG)) : Boolean

Parâmetros
O1 = OG fonte a partir do qual será baseada a relação de cruzamento.
O2 = conjunto de OGs a partir dos quais quer-se verificar a relação de cruzamento em relação ao OG fonte.

Comentários adicionais
Esta transação primitiva verifica se todos os OGs do conjunto O2 cruzam o OG fonte.

Não Cruzamento

NotSpan (O1: OG, O2: Set (OG)) : Boolean

Parâmetros
O1 = OG fonte a partir do qual será baseada a relação de cruzamento.
O2 = conjunto de OGs a partir dos quais quer-se verificar a relação de cruzamento em relação ao OG fonte.

Comentários adicionais
Esta transação primitiva verifica se todos os OGs do conjunto O2 não cruzam o OG fonte.

5.3.3 Transações primitivas de busca topológica

Busca por proximidade mínima

Nearest (O: OG, TypeOG: TO) : Set (OG)

Parâmetros
O = OG fonte a partir do qual será baseada a relação de proximidade mínima.
TypeOG = indica o tipo dos OGs que quer-se buscar.

Tipos
TO = {OG-0D, OG-1D, OG-2D} Obs.: pode-se usar, por exemplo: {OG-0D, OG-2D}

Comentários adicionais
Esta transação primitiva busca o OG do tipo TypeOG que está mais próximo do OG fonte.

Busca por adjacência

Adjacent (O: OG, TypeOG: TO) : Set (OG)

Parâmetros
O = OG fonte a partir do qual será baseada a relação de adjacência.
TypeOG = indica o tipo dos OGs que quer-se buscar.

Tipos
TO = {OG-0D, OG-1D, OG-2D}

Comentários adicionais
Esta transação primitiva busca todos os OGs do tipo TypeOG que são adjacentes ao OG fonte.

Busca por exclusão de adjacência

NotAdjacent (O: OG, TypeOG: TO) : Set (OG)

Parâmetros
O = OG fonte a partir do qual será baseada a relação de adjacência.
TypeOG = indica o tipo dos OGs que quer-se buscar.

Tipos
TO = {OG-0D, OG-1D, OG-2D}

Comentários adicionais
Esta transação primitiva busca todos os OGs do tipo TypeOG que não são adjacentes ao OG fonte.

Busca por inclusão – relacionamento topológico está contido

Inside (O: OG, TypeOG: TO, Boundary: Boolean) : Set (OG)

Parâmetros
O = OG fonte a partir do qual será baseada a relação de inclusão.
TypeOG = indica o tipo dos OGs que quer-se buscar.
Boundary = se verdadeiro, indica a inclusão da fronteira do OG fonte na pesquisa.
Relevante apenas para OGs-2D.

Tipos
TO = {OG-0D, OG-1D, OG-2D}

Busca por inclusão – relacionamento topológico está contido (continuação)

Comentários adicionais
Esta transação primitiva busca todos os OGs do tipo TypeOG que estão contidos no OG fonte, considerando-se ou não a fronteira deste.

Busca por exclusão de inclusão – relacionamento topológico está contido

NotInside (O: OG, TypeOG: TO, Boundary: Boolean) : Set (OG)

Parâmetros
O = OG fonte a partir do qual será baseada a relação de inclusão.
TypeOG = indica o tipo dos OGs que quer-se buscar.
Boundary = se verdadeiro, indica a inclusão da fronteira do OG fonte na pesquisa.
Relevante apenas para OGs-2D.

Tipos
TO = {OG-0D, OG-1D, OG-2D}

Comentários adicionais
Esta transação primitiva busca todos os OGs do tipo TypeOG que não estão contidos no OG fonte, considerando-se ou não a fronteira deste.

Busca por inclusão – relacionamento topológico contém

Embody (O: OG, TypeOG: TO, Boundary: Boolean) : Set (OG)

Parâmetros
O = OG fonte a partir do qual será baseada a relação de inclusão
TypeOG = indica o tipo dos OGs que quer-se buscar.
Boundary = se verdadeiro, indica a inclusão da fronteira do OG fonte na pesquisa.
Relevante apenas para OGs-2D.

Tipos
TO = {OG-0D, OG-1D, OG-2D}

Comentários adicionais
Esta transação primitiva busca todos os OGs do tipo TypeOG que contêm o OG fonte, considerando-se ou não a fronteira deste.

Busca por exclusão de inclusão – relacionamento topológico contém

NotEmbody (O: OG, TypeOG: TO, Boundary: Boolean) : Set (OG)

Busca por exclusão de inclusão – relacionamento topológico contém (continuação)

Parâmetros
O = OG fonte a partir do qual será baseada a relação de inclusão.
TypeOG = indica o tipo dos OGs que quer-se buscar
Boundary = se verdadeiro, indica a inclusão da fronteira do OG fonte na pesquisa.
Relevante apenas para OGs-2D.

Tipos
TO = {OG-0D, OG-1D, OG-2D}

Comentários adicionais
Esta transação primitiva busca todos os OGs do tipo TypeOG que não contêm o OG fonte, considerando-se ou não a fronteira deste.

Busca por interseção

Intersect (O: OG, TypeOG: TO) : Set (OG)

Parâmetros
O = OG fonte a partir do qual será baseada a relação de interseção.
TypeOG = indica o tipo dos OGs que quer-se buscar.

Tipos
TO = {OG-0D, OG-1D, OG-2D}

Comentários adicionais
Esta transação primitiva busca todos os OGs do tipo TypeOG que intersectam o OG fonte.

Busca por exclusão de interseção

NotIntersect (O: OG, TypeOG: TO) : Set (OG)

Parâmetros
O = OG fonte a partir do qual será baseada a relação de interseção.
TypeOG = indica o tipo dos OGs que quer-se buscar.

Tipos
TO = {OG-0D, OG-1D, OG-2D}

Comentários adicionais
Esta transação primitiva busca todos os OGs do tipo TypeOG que não intersectam o OG fonte

---

Busca por cruzamento

Cross (O: OG, TypeOG: TO) : Set (OG)

Parâmetros
O = OG fonte a partir do qual será baseada a relação de cruzamento.
TypeOG = indica o tipo dos OGs que quer-se buscar.

Tipos
TO = {OG-0D, OG-1D, OG-2D}

Comentários adicionais
Esta transação primitiva busca todos os OGs do tipo TypeOG que cruzam o OG fonte.

Busca por exclusão de cruzamento

NotCross (O: OG, TypeOG: TO) : Set (OG)

Parâmetros
O = OG fonte a partir do qual será baseada a relação de cruzamento.
TypeOG = indica o tipo dos OGs que quer-se buscar.

Tipos
TO = {OG-0D, OG-1D, OG-2D}

Comentários adicionais
Esta transação primitiva busca todos os OGs do tipo TypeOG que não cruzam o OG fonte.

5.3.4 Transações primitivas topológicas escalares

Interseção escalar

WhereIntersect (O1: OG, O2: Set(OG)) : Set ((x,y))

Parâmetros
O1 = OG fonte a partir do qual será baseada a relação de interseção.
O2 = conjunto de OGs a partir dos quais quer-se verificar a relação de interseção em relação ao OG fonte.

Interseção escalar (continuação)

Comentários adicionais
Esta transação primitiva determina as coordenadas geográficas dos pontos de interseção do O1 com os OGs do conjunto O2.

5.3.5 Transações primitivas escalares

Distância mínima entre dois OGs

MinDistance (O1, O2: OG) : Float

Parâmetros
O1 = OG fonte 1.
O2 = OG fonte 2.

Área de um OG-2D

Area (O: OG-2D) : Float

Parâmetros
O = OG-2D fonte que quer-se calcular a área.

Perímetro de um OG-2D

Perimeter (O: OG-2D) : Float

Parâmetros
O = OG-2D fonte que quer-se calcular o perímetro.

Comprimento de um OG-1D

Length (O: OG-1D) : Float

Parâmetros
O = OG-1D fonte que quer-se calcular o comprimento.

5.3.6 Transações primitivas baseadas em conjunto

União

Union (O: Set(OG)) : Set(OG)

Parâmetros
O = conjunto de OGs fontes que quer-se unir.

Comentários adicionais
Esta transação primitiva efetua a operação de união entre as fronteiras de vários OGs.

Interseção

Intersection (O: Set(OG)) : Set(OG)

Parâmetros
O = conjunto de OGs fontes que quer-se calcular a interseção.

Comentários adicionais
Esta transação primitiva efetua a operação de interseção entre as fronteiras de vários OGs.

Diferença

Difference (O1, O2: Set(OG)) : Set(OG)

Parâmetros
O1 = OGs fontes para determinação da operação de diferença.
O2 = conjunto de OGs a partir dos quais será efetuada a operação de diferença.

Comentários adicionais
Esta transação primitiva efetua a operação de diferença entre as fronteiras dos OGs fontes e os OGs do conjunto O2 (O1– O2).

5.3.7 Transações primitivas que envolvem visualização gráfica

Visualização gráfica

Display (O: Set (OG), Bottom: Set ((x,y))

Parâmetros
O = conjunto de OGs que serão visualizados graficamente na tela do computador.
Bottom = conjunto de dois pontos que determina a porção horizontal (eixo x) do *extent* que será visualizada.

Comentários adicionais
Esta transação primitiva coloca na tela de um computador, mais especificamente em uma janela, um conjunto de objetos geográficos. Para isto, é assumido o uso do conceito de *extent* (seção 5.5.3) e de um *aspect ratio* para a janela na qual os objetos geográficos serão visualizados. O valor de *aspect ratio*, assim como o tamanho real da janela, entretanto, não é fixado. O parâmetro *bottom* determina a porção horizontal do *extent* que será visualizada. Para isto, são fixados dois pontos correspondendo respectivamente aos pontos inferior esquerdo e inferior direito de um retângulo contido no *extent*. Os pontos superior esquerdo e superior direito deste retângulo são derivados a partir do *aspect ratio* da janela em questão. O parâmetro *bottom* pode ser omitido (valor = *default*), indicando que o retângulo deve conter todos os objetos geográficos contidos no conjunto *O*. Isto caracteriza uma simples transação de *display*. Entretanto, a variação do parâmetro *bottom* também pode ser utilizada para representar as transações de *zoom* e *pan*. Para a transação *zoom in*, deve-se escolher um retângulo que esteja contido no retângulo que engloba todos os objetos geográficos contidos no conjunto *O*. A transação de *zoom out*, por sua vez, consiste simplesmente em um *redisplay*, que pode ser obtido fixando o valor do parâmetro *bottom* com o valor que este possuía em um momento imediatamente anterior. Por fim, a transação de *pan* pode ser representada fixando o valor do parâmetro *bottom* de acordo com o traslado desejado.

5.3.8 Transações primitivas diversas

Determinação de coordenadas geográficas

Bearing (O: OG) : Set ((x,y))

Parâmetros
O = OG fonte.

Determinação de coordenadas geográficas (continuação)

Comentários adicionais
Esta transação primitiva determina as coordenadas geográficas de um OG fonte.

Geração de um OG-1D a partir de OGs-0D

Segment (O1, O2: OG-0D) : OG-1D

Parâmetros
O1 = OG-0D fonte 1.
O2 = OG-0D fonte 2.

Comentários adicionais
Esta transação primitiva gera um OG-1D a partir de dois OGs-0D.

Carga do sistema

LoadSystem

Comentários adicionais
Esta transação primitiva inicializa o SIG, construindo índices espaciais e convencionais

---

Seleção convencional

Select (E: Exp, TypeOG: TO) : Set (OG)

Parâmetros
E = pares de atributos e valores que devem ser satisfeitos para propiciar a seleção de um OG
TypeOG = indica o tipo de OGs que quer-se selecionar.

Tipos
Exp = {[*Not*]($attrib_1$ [rel_1 val_1]) log_1 [Not]($attrib_2$ [rel_2 val_2]) ...log_n [Not]($attrib_m$ [rel_m val_m])}
onde rel_i = operador relacional (= , > , < , <> , ≥ , ≤), p/ i = 1..m
log_i = operador lógico (*And, Or, Not*, Xor, ...), p/ i = 1..m-1
$attrib_i$ = atributo convencional, p/ i = 1..m , na forma *OG*.attrib
val_i = valor numérico ou atributo convencional ou expressão, p/ i = 1..m

TO = {OG-0D, OG-1D, OG-2D}

Seleção convencional (continuação)

Comentários adicionais
Esta transação primitiva seleciona os OGs de tipo TypeOG que possuam expressão verdadeira (*True*). Vale destacar que a semântica da expressão deve ser garantida.

Número de OGs

Number (O: Set(OG), TypeOG: TO) : Integer

Parâmetros
O = conjunto de OGs fonte.
TypeOG = tipo dos OGs na qual esta transação atuará.

Tipos
TO = {OG-0D, OG-1D, OG-2D}

Comentários adicionais
Esta transação primitiva calcula o número de OGs do tipo TypeOG que estão contidos em O

Armazenamento

Save (O: Set (OG))

Parâmetros
O = conjunto de OGs que serão armazenados em memória secundária.

5.3.9 Transações primitivas específicas do formato *raster*

Agrupamento

GroupCell (M: Set(Cell)) : Set (OG)

Parâmetros
M = matriz de células a ser agrupada.

Comentários adicionais
Dada uma matriz de células, para cada conjunto de células adjacentes entre si e com a mesma categoria é associado um OG distinto, cuja dimensão dependerá do número de células de sua fronteira.

Mudança da resolução espacial

ResolutionCell (M: Set(Cell), NewLengthCell: Integer) : Set (Cell)

Parâmetros
M = matriz de células cuja resolução espacial será alterada.
NewLengthCell = novo tamanho lateral das células.

Comentários adicionais
Esta transação associa um novo tamanho lateral para as células, formando assim um novo conjunto de células. A categoria de cada célula gerada é obtida através de regras de decisão, com base nas categorias das células agrupadas.

Comparação de matrizes

CompareCell (M1, M2: Set(Cell)) : TypeCompare

Parâmetros
M1 = matriz de células fonte 1.
M2 = matriz de células fonte 2.

Tipos
TypeCompare = {equal: ***Boolean***, number: ***Integer***, M: ***Set(Cell)***}

Comentários adicionais
Esta transação compara as categorias de duas matrizes de células. Como resultado é gerado um valor lógico indicando se estas são completamente iguais (equal), o número de células iguais (number) e o conjunto destas células iguais (M). Esta transação é utilizada para verificar a diferença entre diferentes tipos de métodos de coleta de dados.

Número de células

NumberCell (M: Set(Cell)) : LongInteger

Parâmetros
M = matriz de células fonte.

Comentários adicionais
Esta transação retorna o número de células da matriz fonte.

Célula de valor mínimo

MinValueCell (M: Set(Cell)) : Set (Cell)

Parâmetros
M = matriz de células fonte.

Comentários adicionais
Esta transação obtém as células cujas categorias possuem o menor valor. Esta transação somente pode ser aplicada em categorias numéricas.

Valor médio de categoria

AverageCell (M: Set(Cell)) : Float

Parâmetros
M = matriz de células fonte.

Comentários adicionais
Esta transação calcula o valor médio das categorias de uma matriz de células fonte. Válido somente para categorias numéricas.

5.3.10 Transações mistas

Conversão do formato *raster* para o formato vetorial

RasterVector (M: Set(Cell)) : Set (OG)

Parâmetros
M = matriz de células que será convertida para o formato vetorial.

Conversão do formato vetorial para o formato *raster*

VectorRaster (O: Set (OG), LengthCell: Integer, F: Function) : Set(Cell)

Parâmetros
O = conjunto de pontos, linhas e polígonos que será convertido para uma matriz de células.
LengthCell = tamanho lateral das células.
F = função que determina a regra de formação de categorias.

Superposição visual

VisualOverlay (O: Set (OG), M: Set (Cell))

Parâmetros
O = dados no formato vetorial a serem superpostos.
M = matriz de células a ser superposta.

Comentários adicionais
Esta transação primitiva superpõe visualmente dados no formato vetorial e dados no formato *raster*. Geralmente dados no formato *raster* constituem o pano de fundo desta superposição.

5.4 Exemplos de Consultas

O intuito desta seção é representar algumas consultas comumente encontradas em aplicações georeferenciadas a partir das transações primitivas previamente apresentadas. Com isto, pode-se verificar a abrangência e o potencial de portabilidade da carga de trabalho do *benchmark* proposto.

A formalização das transações que compõem a carga de trabalho do *benchmark* não visou, em nenhum momento, a caracterização de uma linguagem de consulta geográfica. Em outras palavras, durante a formalização destas transações optou-se em caracterizar os principais conceitos geográficos de maneira precisa, em detrimento de uma estruturação refinada. Desta forma, a representação de transações complexas a partir da carga de trabalho do *benchmark* é flexível, no sentido de não se apegar a todos os detalhes de sintaxe presentes em linguagens de consultas geográficas. Esta flexibilidade é necessária, uma vez que a carga de trabalho não é específica a um modelo de dados, tal como o relacional (SQL).

A seguir são apresentadas três consultas SIGs. Inicialmente, cada consulta é descrita tanto em linguagem corrente quanto na linguagem de consulta geográfica *Spatial SQL* [Ege94], para facilitar o seu entendimento. Finalmente, a consulta em questão é formulada com base nas transações primitivas do *benchmark* proposto.

Consulta 1

Linguagem corrente:

"Quais as cidades com população em idade escolar superior a 40.000 habitantes e que estão localizadas no máximo a 120 km do rio Pardo ?".

Representação em *Spatial SQL*:

CREATE TABLE cidade
(nome *char* (20)
população_idade_escolar *longinteger*
geometria *spatial_0*) ;

CREATE TABLE rio
(nome *char* (20)
geometria *spatial_1*) ;

SELECT cidade.nome
FROM cidade, rio
WHERE (cidade.população_idade_escolar > 40.000) *and* (rio.nome = "Pardo") *and*
(*Distance* (cidade.geometria, rio.geometria) <= 120.000)

Representação através de transações primitivas:

Esta consulta pode ser respondida de duas maneiras: optando-se primeiramente pela pesquisa do predicado convencional, ou optando-se primeiramente pela pesquisa do predicado espacial. Esta escolha deve ser efetuada cuidadosamente pelo otimizador de consultas. Uma escolha errada pode afetar significativamente o desempenho, uma vez que estes predicados atuam sobre diferentes tamanhos e tipos de conjuntos de dados.

Caso 1: pesquisa efetuada primeiramente com o predicado convencional.

Neste caso, o intuito é restringir a pesquisa buscando cidades que satisfaçam a asserção: população em idade escolar superior a 40.000 habitantes. Para cada cidade que satisfizer esta asserção, verifica-se se ela está situada a uma distância de no máximo 120 km do Rio Pardo.

A := *Select* (cidade.população_idade_escolar > 40.000, cidade) ⇒ Seleção Convencional
B := *SimpleBuffer* (rio_pardo, 120.000, *Null*) ⇒ Zona de *Buffer* Simples
C := ∅
para cada $a \in A$
faça se *Cover* (B, a, *True*) ⇒ Inclusão *Booleana*
então C := C + a

Caso 2: pesquisa efetuada primeiramente com o predicado espacial.

Neste caso, o intuito é restringir a pesquisa buscando cidades que estejam no máximo a 120 km do Rio Pardo. Em seguida, para cada cidade que satisfaça esta condição, verifica-se se ela possui uma população em idade escolar superior a 40.000 habitantes.

A := *SimpleBuffer* (rio_pardo, 120.000, *Null*) ⇒ Zona de *Buffer* Simples
B := *Inside* (A, cidade, *True*) ⇒ Busca por Inclusão
C := *Select* (cidade.população_idade_escolar > 40.000, B) ⇒ Seleção Convencional

Caso 3: composição lógica. Esta composição lógica visa representar a consulta de uma forma que o resultado final alcançado seja uma resposta à consulta, entretanto, a seqüência de execução das transações não é relevante, sendo inclusive pouco utilizado na prática.

P1 | P2 (D:= A ∩ C) ⇒ Resultado final)
P1: A := *Select* (cidade.população_idade_escolar > 40.000, cidade) ⇒ Seleção Conv.
P2: B := *SimpleBuffer* (rio_pardo, 120.000, *Null*) ⇒ Zona de *Buffer* Simples
C := *Inside* (B, cidade, *True*) ⇒ Busca por Inclusão

Consulta 2

Linguagem corrente:

"Quais bairros são adjacentes ao Taquaral e possuem uma área menor que 15 km^2 ?".

Representação em *Spatial SQL*:

CREATE TABLE bairro
(nome *char* (20)
.....
geometria *spatial_2*) ;

SELECT bairro.nome
FROM bairro
WHERE (bairro.geometria *Meet* taquaral) *and*
(*Area* (bairro.geometria) < 15.000)

Representação através de transações primitivas:

Caso 1: A estratégia para responder esta consulta consiste em primeiramente determinar todos os bairros adjacentes ao bairro Taquaral e em seguida verificar quais destes bairros possuem uma área inferior a 15 km^2.

A := *Adjacent* (taquaral, bairro) ⇒ Busca por adjacência
B := ∅
para cada $a \in A$
  faça se *Area* (a) < 15.000 ⇒ Área de um OG-2D
        então B := B + a

Caso 2: composição lógica.

P1 | P2 (C:= A ∩ B ⇒ Resultado final)
P1 : A := *Adjacent* (taquaral, bairro) ⇒ Busca por adjacência
P2: B := ∅
  para cada $b \in$ bairro
    faça se *Area* (b) < 15.000 ⇒ Área de um OG-2D
          então B := B + b

Consulta 3

Linguagem corrente:

"Quais os ramos da rede de eletricidade que não intersectam a rede de água e que possuem um comprimento superior a 20 km ? ".

Representação em *Spatial SQL*:

CREATE TABLE rede_elétrica
 (número integer
  .....
  geometria *spatial_1*) ;

CREATE TABLE rede_água
 (número integer
  .....
  geometria *spatial_1*) ;

SELECT rede_elétrica.número
FROM rede_elétrica, rede_água
WHERE (rede_elétrica.geometria *Not Overlap* rede_água.geometria) *and*
      (*Length* (rede_elétrica.geometria) > 20.000)

Representação através de transações primitivas:

Caso 1: A estratégia para responder esta consulta consiste em primeiramente determinar todos os ramos da rede elétrica que não intersectam a rede de água e em seguida verificar quais destes ramos possuem um comprimento superior a 20 km.

A := ∅
para cada r ∈ rede_elética
faça se (*Disjoint* (r, rede_água)) ⇒ Disjunção *Booleana*
and (*Length* (r) > 20.000) ⇒ Comprimento de um OG-1D
então A := A + r

Caso 2: composição lógica.

P1 | P2 (C:= A ∩ B ⇒ Resultado final)
P1 : A := ∅
para cada r ∈ rede_elética
faça se (*Disjoint* (r, rede_água)) ⇒ Disjunção *Booleana*
então A := A + r
P2: B:= ∅
para cada r ∈ rede_elética
faça se (*Length* (r) > 20.000) ⇒ Comprimento de um OG-1D
então B := B + r

5.5 SIGs: principais características relativas aos dados

5.5.1 Conceitos básicos e tipos de dados do *benchmark*

Esta seção caracteriza os dados em termos dos tipos necessários para a representação de aplicações georeferenciadas e, adicionalmente, procedimentos para se realizar a geração de dados sintéticos. Esta caracterização será independente da organização dos dados (estrutura dos arquivos, quantidade de dados armazenados, entre outros), de modo a permitir a execução da carga de trabalho do *benchmark* proposto segundo aplicações georeferenciadas específicas, ou seja, possibilitando a criação de versões do *benchmark*.

Em relação à carga de trabalho, pode-se adaptar: freqüência de execução de cada transação primitiva (inclusive nula), ordem de execução, tipo de "*bufferização*" (se o *buffer* deve ou não ser "limpo" após a execução de cada transação), e composição dos dados (localização – sobre quais arquivos cada transação irá atuar, e seletividade). Além disto, a carga de trabalho pode ser formada por transações complexas, derivadas de uma composição de transações primitivas. Já os dados podem ser formados por um subconjunto de dados reais de uma aplicação georeferenciada, ou ainda por dados sintéticos gerados a partir de uma previsão de composição real. Em relação aos dados, pode-se adaptar: tipos,

quantidade, distribuição, controle de seletividade e estrutura do banco de dados. A adaptação de aspectos relacionados às transações e aos dados tende a gerar resultados de desempenho que reflitam o desempenho real de um SIG durante a sua operação. Isto é muito útil, principalmente quando um gerente de sistemas deseja saber se uma determinada configuração de sistema computacional irá suportar uma dada aplicação convenientemente.

Entretanto, o desempenho medido segundo uma aplicação georeferenciada específica não pode ser usado para propósitos de comparação de desempenho. Para isto, deve-se executar a carga de trabalho do *benchmark* proposto em uma aplicação alvo modelada a partir do levantamento de características comumente encontradas em aplicações georeferenciadas. Isto é efetuado na seção 6.2, onde uma aplicação alvo composta de dados sintéticos é descrita. Nesta aplicação, inclusive, algumas características foram associadas de forma a provocar uma alta degradação de desempenho nos SIGs a serem testados.

Os tipos de dados considerados pelo *benchmark*, já abordando o formato de armazenamento dos dados espaciais (*raster* ou vetorial), são:

- **dados convencionais:** dados comumente encontrados em SGBDs convencionais. Os tipos considerados pelo *benchmark* são: *integer*, *long integer*, *float*, *string* e *date* (seção 3.4).

- **célula:** dado espacial armazenado no formato *raster*. Para este tipo de dado deve-se associar uma categoria e armazenar as suas coordenadas centrais x e y (por exemplo em relação à sua posição na matriz de células).

- **ponto:** dado espacial de dimensão zero no formato vetorial. Para este tipo de dado deve-se armazenar as suas coordenadas x e y.

- **linha:** dados espaciais unidimensionais armazenados no formato vetorial. Uma conexão retilínea entre dois pontos é denominada segmento de linha. A interligação de vários segmentos de linha forma uma linha. Deve-se armazenar para cada linha, ordenadamente, as coordenadas de todos os pontos que a compõem, sendo necessário fixar o número de pontos, orientação e comprimento.

- **polígono:** dado espacial bidimensional armazenado no formato vetorial. Para este tipo de dado deve-se armazenar ordenadamente as coordenadas de todos os pontos que o compõem. Deve-se fixar o número de pontos (vértices), orientação e área.

Para dados espaciais do tipo *linha* e *polígono* é permitida a utilização estruturada de dados espaciais mais simples (*segmentos de linha* e *pontos*) para efetuar a descrição de suas geometrias, como no modelo Arc-Node. Isto visa à redução do espaço de armazenamento destes dados, uma vez que se elimina qualquer armazenamento redundante.

Para qualquer um dos tipos de dados descritos anteriormente é permitido o armazenamento em forma compactada. Isto pode reduzir o espaço de armazenamento de maneira significativa. Entretanto, é necessário que a técnica de compactação de dados seja transparente no sentido de não requerer qualquer esforço adicional de programação ou de interação com o SIG, além de preservar a precisão dos dados. Para dados espaciais no formato *raster*, o termo codificação de células corresponde ao processo de compactação de dados, sendo que as principais técnicas são: *run-length encoding, value point encoding* e *chain codes*. O uso destas técnicas, entretanto, prejudica a determinação de relacionamentos topológicos.

Por último, a descrição destes tipos de dados serve apenas como referência, tendo por objetivo padronizar o conteúdo dos dados presentes no *benchmark*. A troca de qualquer um destes tipos é permitida, desde que seja por um tipo mais abrangente (que englobe o intervalo de variação do tipo de dados trocado).

Dados gráficos não serão considerados pelo *benchmark*. Estes dados, apesar de muito importantes para alguns tipos de aplicações georeferenciadas, são mais utilizados para efeitos de visualização, e deste modo são tratados principalmente por funções de processamento de imagens. Por via de regra, dados gráficos são implicitamente relacionados aos dados espaciais e convencionais, através de cada objeto geográfico contido no banco de dados. Quando dados gráficos existem isoladamente, estes geralmente adicionam apenas uma informação visual a uma consulta específica, inexistindo no sentido de fazer parte de um predicado em uma consulta geográfica. Assim, uma consulta geográfica do tipo "Quais as casas próximas da Unicamp que possuem mais de quatro quartos, e estão à venda a um preço entre 30.000 e 400.000 reais ?", pode retornar como resposta, adicionalmente à visualização dos dados espaciais e convencionais, uma foto em perspectiva de cada casa que satisfaça a consulta, para que se possa ter uma noção da arquitetura e estilo do imóvel. Consultas em SIGs dificilmente envolveriam algo como "Quais as casas perto da Unicamp parecidas com a foto Casa-1 ?". Além disto, alguns SIGs permitem a omissão do armazenamento de dados gráficos relativos a dados espaciais e convencionais, uma vez que dados gráficos podem ser derivados a partir destes últimos.

5.5.2 Geração de dados convencionais

A geração de dados convencionais sintéticos é amplamente discutida em [Per90]. Este trabalho, por sua vez, simplifica as discussões a respeito da geração de dados convencionais descritas por este autor. Os procedimentos descritos neste trabalho para a geração de dados convencionais visam à formulação de consultas geográficas envolvendo ambos predicados convencionais e espaciais. Isto é freqüentemente efetuado em aplicações georeferenciadas, tornando necessário um certo grau de seletividade controlada dos dados convencionais, de forma a testar a habilidade do otimizador de consultas em selecionar o predicado cujo domínio em questão é mais restrito. Entretanto, problemas específicos encontrados na geração de dados convencionais não serão tratados, tal como o controle de seletividade em junções de vários níveis no modelo relacional. A seguir, são descritas

várias formas de distribuição de acordo com o tipo e domínio do dado convencional. Basicamente são usados dois tipos de distribuição: distribuição uniforme única (sem repetição de valores) e distribuição não uniforme segundo a lei de *Zipf* (seção 3.4). Os algoritmos relativos às distribuições citadas são encontrados em [Per90], sendo facilmente adaptados para o contexto deste trabalho.

Em aplicações georeferenciadas, atributos tais como nomes de cidades, ruas e bairros, possuem uma repetição quase nula, sendo assim indicada a geração de valores únicos. Entretanto, indica-se uma distribuição não uniforme para as iniciais dos valores destes atributos. Assim, por exemplo, pode-se gerar um conjunto de nomes de cidades cujas iniciais sejam distribuídas de forma não uniforme, onde a letra X ocupa um baixo *rank*, enquanto as letras A, N e S ocupam um alto *rank*. Um *string* na forma $X\,Y_1Y_2..Y_n\,Z_1Z_2..Z_m$, onde o tamanho do *string* é igual a $n+m+1$, garante ambas premissas de unicidade e distribuição não uniforme. Para isto, deve-se preencher seqüencialmente os caracteres $Y_1Y_2..Y_n$ com números, sendo o valor de n calculado de acordo com a quantidade de dados do atributo. Os caracteres $Z_1Z_2..Z_m$ não são significativos, sendo preenchidos com um conjunto de caracteres fixos quaisquer, com o intuito apenas de formar o tamanho final desejável do atributo. Já o caractere X deve ser gerado aleatoriamente, por exemplo segundo uma distribuição *Zipf*.

Os valores de decaimento mais utilizados para distribuições *Zipf* são descritos na tabela 5.2, sendo que para $z = 0$ obtém-se uma distribuição uniforme. A seletividade de dados distribuídos segundo uma distribuição *Zipf* é parcialmente controlada, de acordo com as freqüências geradas para os *ranks*. Como exemplo, supondo que são gerados apenas 10 valores distintos para um atributo, através de uma distribuição *Zipf* com $z = 1$, pode-se obter uma seletividade de 51%, relativos aos *ranks* 1 e 2.

Rank	Freqüências			
	z = 0	z = 0.5	z = 1	z = 3
1	0.1	0.1992	0.34	0.8351
2	0.1	0.1408	0.17	0.1044
3	0.1	0.1150	0.11	0.0309
4	0.1	0.0996	0.09	0.0130
5	0.1	0.0891	0.07	0.0067
6	0.1	0.0813	0.06	0.0038
7	0.1	0.0753	0.05	0.0024
8	0.1	0.0704	0.04	0.0016
9	0.1	0.0664	0.04	0.0012
10	0.1	0.0629	0.03	0.0009
Total =	1.0	1.0000	1.00	1.0000

Tabela 5.2 Exemplo de uma distribuição *Zipf*.

Outros tipos de atributos presentes em aplicações georeferenciadas são compostos de valores numéricos (*integer*, *long integer* ou *float*), tais como população, número de hospitais e temperatura. Para atributos compostos de valores numéricos do tipo *integer* cujo domínio é restrito (por exemplo, entre 1 e 15, para números de hospitais) é indicada a geração aleatória segundo uma distribuição não uniforme. Assim, consegue-se gerar valores repetidos e em maior número para os valores tidos mais freqüentes.

Para atributos compostos de valores numéricos do tipo *integer* cujo domínio é abrangente e para atributos compostos de valores numéricos do tipo *long integer* ou *float* é indicada a divisão dos valores possíveis em faixas. Primeiramente, é efetuada a geração aleatória da faixa, segundo uma distribuição não uniforme. Em seguida, para a faixa selecionada é gerado aleatoriamente um valor contido em seu domínio. Assim, consegue-se obter um maior número de valores para as faixas tidas como mais freqüentes, sendo que dentro da faixa o tipo de distribuição é irrelevante. Como exemplo, para o atributo população (*long integer*) de um objeto geográfico cidade cujos valores possíveis podem variar entre 1.000 e 13.000.000, pode-se determinar 6 faixas:

faixa 1 = 5.000.000 < i ≤ 13.000.000
faixa 2 = 1.000.000 < i ≤ 5.000.000
faixa 3 = 500.000 < i ≤ 1.000.000
faixa 4 = 100.000 < i ≤ 500.000
faixa 5 = 10.000 < i ≤ 100.000
faixa 6 = i ≤ 10.000

Sabendo-se que poucas cidades possuem população superior a 1.000.000, pode-se dar um baixo *rank* para a faixa 1. Similarmente, para cada faixa é atribuído um *rank* de acordo com o número de cidades com população contida no intervalo da faixa. Desta forma, gera-se diferentes freqüências de valores para cada faixa (distribuição não uniforme), sendo, entretanto, gerado aleatoriamente o valor dentro de cada faixa, sem nenhum controle de distribuição.

Sugere-se uma distribuição não uniforme para cada componente de atributos do tipo *date*, ou seja para dia, mês e ano. Assim, consegue-se uma rica combinação de seletividade.

Atributos convencionais pouco usados como argumentos em cláusulas de seleção devem ser gerados aleatoriamente sem nenhum controle de seletividade, ou ainda gerados segundo um valor constante. Isto visa apenas ao preenchimento do valor do atributo, como um *string*, por exemplo.

Adotando-se os procedimentos descritos para a geração de dados convencionais, no contexto de aplicações georeferenciadas, é possível gerar-se dados convencionais para cada objeto geográfico, em qualquer um dos tipos de dados descritos e respectivas distribuições.

5.5.3 Geração de dados espaciais no formato vetorial

Esta seção descreve aspectos relacionados à geração de dados espaciais sintéticos no formato vetorial (dados vetoriais). A geração destes dados difere consideravelmente da forma como são gerados dados convencionais. Nestes últimos, os valores de um atributo são gerados de acordo com o seu domínio, segundo uma distribuição uniforme ou não uniforme. Além disto, todos conceitos envolvendo a distribuição são bem definidos e conhecidos, tal como uma distribuição uniforme, a qual corresponde à geração de valores de acordo com um freqüência única para todos os valores distintos do domínio. A geração de dados vetoriais, entretanto, não é efetuada unicamente em função de seu domínio (valores que podem ser gerados para as coordenadas x e y). Aspectos relacionados à geometria do dado espacial são importantes, tais como forma geométrica (*shape*), tamanho (comprimento para linhas e área para polígonos), e complexidade (número de pontos que compõem a geometria). Este *benchmark* propõe uma distribuição controlada da localização e quantidade de dados para todos os tipos de dados vetoriais, e com exceção de pontos, também propõe uma distribuição controlada da forma geométrica, tamanho e complexidade.

A geração de dados vetoriais proposta é baseada nos conceitos de *extent* e *division*. Um *extent* consiste na área total abrangida pela aplicação. Já uma *division* consiste em uma área contida inteiramente no *extent*. Ambos, *extent* e *division*, podem possuir inúmeras formas de geometria. Entretanto, apenas a forma quadrática será considerada (figura 5.24).

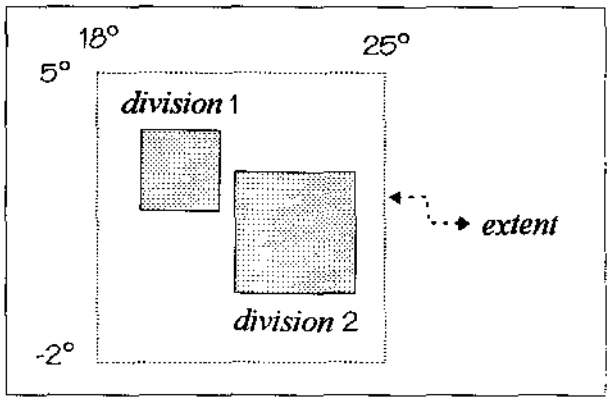


Figura 5.24 Conceitos *extent* e *division*.

Dados vetoriais devem ser gerados segundo disposições específicas de *divisions*. Desta forma, consegue-se controlar a localização dos dados, e conseqüentemente a seletividade destes. Visando à facilidade de formação de *divisions*, deve-se optar por uma divisão hierárquica do *extent* segundo o modelo *quadtree*. Isto simplifica os algoritmos de geração de dados, uma vez que mantém-se o formato quadrático para todas as *divisions* do *extent*. A figura 5.25 ilustra um *extent* contendo um conjunto específico de 40 *divisions*.

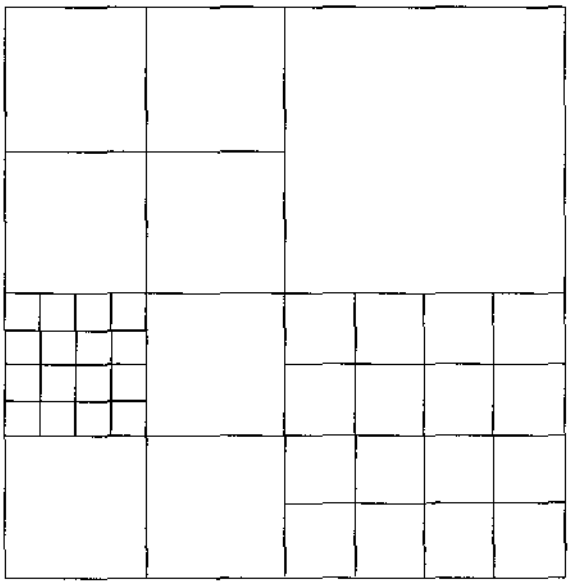


Figura 5.25 Divisão hierárquica do *extent*.

Para cada *division* de um *extent* deve-se determinar a quantidade de cada tipo de dado vetorial. Assim, consegue-se controlar a seletividade, uma vez que os dados de uma *division* (para cada tipo de dado), correspondem a x% do total de dados do *extent*, sendo x = (total de dados na *division* / total de dados no *extent*) * 100.

Em particular, a geração de pontos é simples, uma vez que estes não possuem forma geométrica. Para cada um dos n pontos de uma *division*, deve-se gerar aleatoriamente as coordenadas x e y de tal forma que seus valores estejam contidos no interior[2] da *division*.

A geração de linhas será efetuada com base em três fatores: *shape*, tamanho e complexidade. Os *shapes* visam caracterizar a orientação das linhas (figura 5.26). Dependendo da orientação escolhida, aumenta-se a percentagem de área relativa ao *dead space* dentro do *MBR* usado para representar uma linha. Quanto maior esta percentagem, maior a probabilidade de se escolher uma linha como resposta a uma consulta à qual esta não satisfaz, fazendo necessário sua remoção na fase de refinamento. Os *shapes* C e D (orientação diagonal descendente e ascendente) maximizam a área relativa ao *dead space*, enquanto os *shapes* A e B (orientação vertical e horizontal) minimizam esta área. A descrição de *containers*, tal como *MBR*, é feita na seção 6.1. O tamanho das linhas deve ser fixado em um intervalo entre [5%,95%] do tamanho do lado da *division*. Dependendo do tamanho relativo da *division* e do tamanho das linhas dentro da *division*, pode-se obter linhas ditas grandes ou pequenas. O tamanho das linhas pode influir no desempenho de métodos de acesso espaciais, de maneira análoga à descrita para polígonos em [Cox91]. Por fim, a complexidade corresponde ao número de pontos que as linhas geradas terão. Quanto maior o número de pontos de uma linha, maior a complexidade para se detectar relacionamentos topológicos, tais como interseção e cruzamento.

[2] as coordenadas não devem pertencer à fronteira (seção 5.2.5).

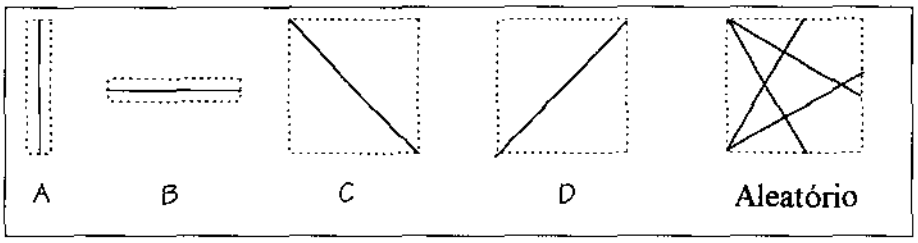


Figura 5.26 *Shapes* de linhas.

Para cada um dos fatores *shape*, tamanho e complexidade pode-se configurar se estes são fixos ou aleatórios. Assim, por exemplo, para uma *division* pode-se gerar linhas com *shape* aleatório, tamanho aleatório no intervalo [10%, 90%] e número fixo de pontos igual a 2. Por outro lado, pode-se gerar linhas com *shape* D, tamanho aleatório no intervalo [20%,80%] e número de pontos aleatórios no intervalo [2,5]. Estes dois exemplos são ilustrados na figura 5.27. A geração aleatória de tamanho e complexidade requer somente que seja fixado o intervalo no qual os valores são válidos. Já para a geração aleatória de *shapes* fica subentendido a possibilidade de se utilizar qualquer tipo de *shape*, além dos shapes A, B, C e D.

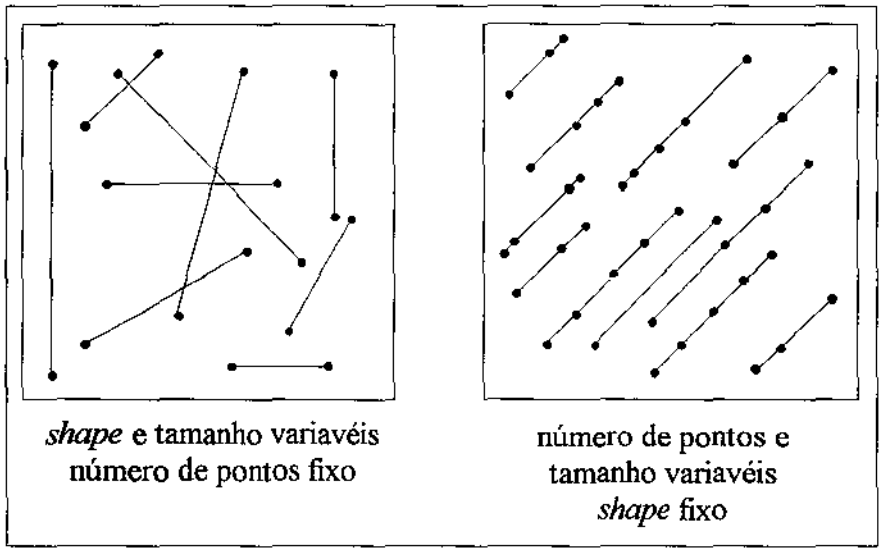


Figura 5.27 Linhas: exemplos de variações de *shape*, tamanho e número de pontos.

Os passos necessários para a geração de linhas são:

- **passo 1:** gera-se aleatoriamente um ponto contido no interior da *division*. Este ponto consiste no primeiro ponto final da linha.

- **passo 2:** traça-se um círculo de raio igual ao tamanho requerido a partir do primeiro ponto gerado. Escolhe-se um ponto contido nos limites do círculo que esteja dentro da *division* e respeite o *shape* escolhido. O ponto escolhido consiste no segundo

ponto final da linha. Caso isto não seja possível, descarta-se o primeiro ponto gerado e inicia-se novamente a seqüência.

- **passo 3:** para cada linha gerada cujo número de pontos requerido seja maior que dois, gera-se aleatoriamente n pontos ao longo dos pontos finais (na reta que liga o primeiro e o segundo ponto final da linha), de modo que estes não ocupem as mesmas coordenadas.

De maneira similar à geração de linhas, a geração de polígonos será efetuada com base nos fatores *shape*, tamanho e complexidade. Entretanto, uma vez definidos a quantidade e o *shape* dos polígonos na *division*, o tamanho destes é automaticamente determinado. Em outras palavras, para um *shape* específico, quanto maior a quantidade de polígonos em uma *division*, menor será o tamanho destes polígonos. Os *shapes* visam caracterizar a orientação dos polígonos, e assim controlar a percentagem de área relativa ao *dead space* dentro do *MBR* usado para representar o polígono. Os shapes dos conjuntos A, C e B (figura 5.28), respectivamente, geram polígonos com 0%, 25% e 50% de área relativa ao *dead space*. Cada um destes tipos de *shapes* gera sempre dois polígonos, de mesma área e formato. Por fim, a complexidade corresponde ao número de pontos que os polígonos gerados terão.

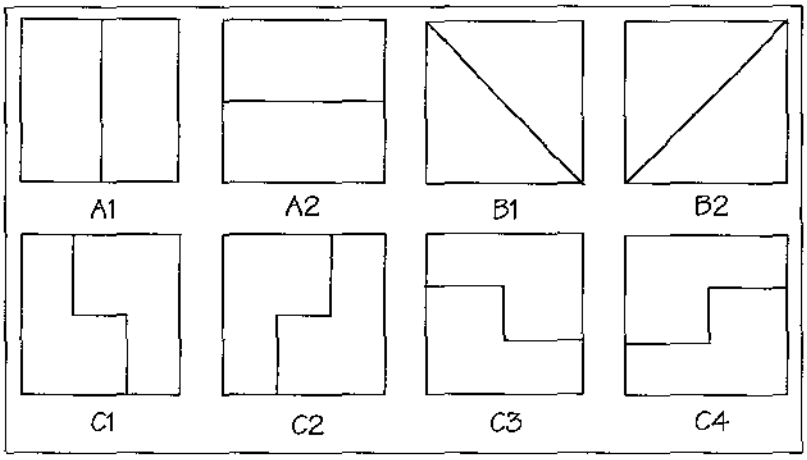


Figura 5.28 *Shapes* de polígonos.

Os passos necessários para a geração de polígonos são:

- **passo 1:** determina-se a quantidade de polígonos de uma *division*. Esta pode ser nula, ou uma potência de 2 (2^k), sendo que k deve ser um número inteiro, ímpar e maior que 0. Caso a quantidade escolhida seja nula, não é gerado nenhum polígono para a *division*. Para uma quantidade de polígonos não nula, divide-se a *division* segundo o modelo *quadtree* em 2^{k-1} partes iguais. Para cada uma destas partes gera-se dois polígonos segundo o *shape* escolhido.

- **passo 2:** determina-se a complexidade dos polígonos aos pares, para os polígonos gerados a partir de um mesmo *shape*. Assim, distribui-se somente uma vez o número de pontos excedente ao longo da divisa destes polígonos (excetuando-se os pontos finais da divisa). Estes pontos não devem ser repetidos, ou seja, não devem possuir as mesmas coordenadas. Para os *shapes* do conjunto A, o número mínimo de pontos é 4 por polígono, sendo estes distribuídos nos extremos dos dois retângulos formados pelo *shape*. Para os *shapes* do conjunto B, o número mínimo de pontos é 3 por polígono, sendo estes distribuídos nos extremos dos dois triângulos formados pelo *shape*. Para os *shapes* do conjunto C, o número mínimo de pontos é 6 por polígono, sendo estes distribuídos nos extremos de cada reta formada pelo *shape*. O número mínimo e a distribuição dos pontos para os *shapes* dos conjuntos A, B e C são ilustrados na figura 5.29.

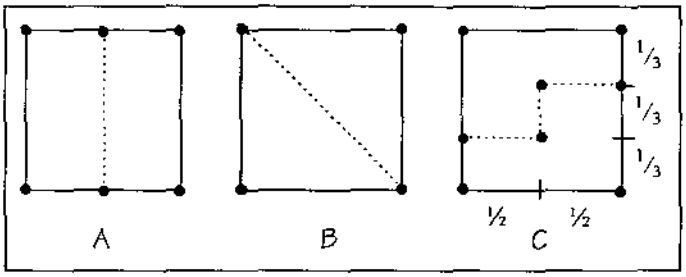


Figura 5.29 Distribuição de pontos nos *shapes* de polígonos.

A complexidade de um polígono pode ser fixa ou aleatória. No primeiro caso, fixa-se um número inteiro que corresponde ao número de pontos de todos os polígonos da *division*. No segundo caso, fixa-se um intervalo no qual o número de pontos deve ser gerado, tal como [6,12]. Deve-se observar, no entanto, o número mínimo de pontos que cada *shape* requer. Em relação aos *shapes* gerados para uma *division*, pode-se escolher um *shape* específico, ou uma lista de *shapes* possíveis. Assim, pode-se gerar somente o *shape* C1, ou o conjunto de *shapes* C (C1, C2, C3 e C4), ou ainda os *shapes* A1, B2 e C3. Quando vários *shapes* são permitidos, deve-se escolher aleatoriamente um dos *shapes* do conjunto. Por fim, vale novamente citar que o tamanho dos polígonos gerados depende da quantidade de polígonos da *division* e do tamanho relativo da *division*. O uso de diferentes *shapes*, entretanto, não altera o tamanho dos polígonos. Este tamanho permanece o mesmo para todos os polígonos da *division*, uma vez que todos os *shapes* descritos geram dois polígonos de mesma área, e sempre estão localizados segundo as partes geradas pelo modelo *quadtree*, ocupando assim um quadrado de mesma área.

Como exemplo, a figura 5.30 ilustra a geração de polígonos para duas *divisions* distintas, sendo que para a primeira (à esquerda) gera-se somente *shapes* B1, com número de pontos aleatórios no intervalo [4,9]. Já para a segunda *division*, gera-se aleatoriamente os *shapes* dos polígonos (usando-se os conjuntos A, B e C), todos com um número fixo de 7 pontos. A quantidade de polígonos para as *divisions* do exemplo é 8.

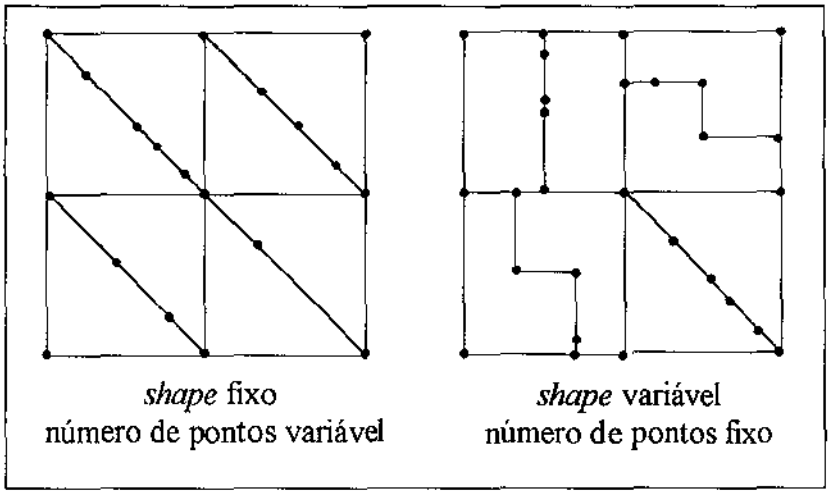


Figura 5.30 Polígonos: exemplos de variações de *shape* e número de pontos.

A geração de polígonos segundo os passos descritos sempre gera polígonos não superpostos. Isto é típico de aplicações geográficas baseadas em temas, nos quais cada conjunto contíguo de polígonos de mesma categoria forma um polígono específico. Para evitar-se que dois polígonos adjacentes possuam uma mesma categoria, sugere-se uma distribuição aleatória única de categoria, como exemplo através da geração seqüencial de número inteiros.

A determinação da quantidade de pontos, linhas e polígonos de cada *division* pode ser efetuada com base no conceito de densidade. Vários tipos de densidade existem, sendo a mais comum a densidade relativa não geográfica. Esta descreve o percentual dos dados que está contido em uma dada *division* em relação ao total de dados contidos no *extent*. Este tipo de densidade pode sempre ser determinado, uma vez que se conhece a quantidade total de dados no *extent* e na *division*, sendo este o ponto chave para se obter uma seletividade controlada.

Entretanto, pode-se querer gerar dados segundo outras densidades. Como exemplo, pode-se citar a densidade relativa geográfica, a qual consiste na razão (total de dados) / (área do *extent*). Assim, pode-se gerar pontos para uma dada *division* de forma que se obtenha uma densidade de 5 pontos/km^2. Para polígonos, no entanto, deve-se respeitar a regra da potência – 2^k, tornando a escolha desta densidade menos flexível.

A figura 5.31 ilustra para dois *extents*, os conceitos de densidade relativa geográfica e não geográfica. Neste exemplo, obtém-se a mesma densidade geográfica e não geográfica para *divisions* de tamanhos distintos.

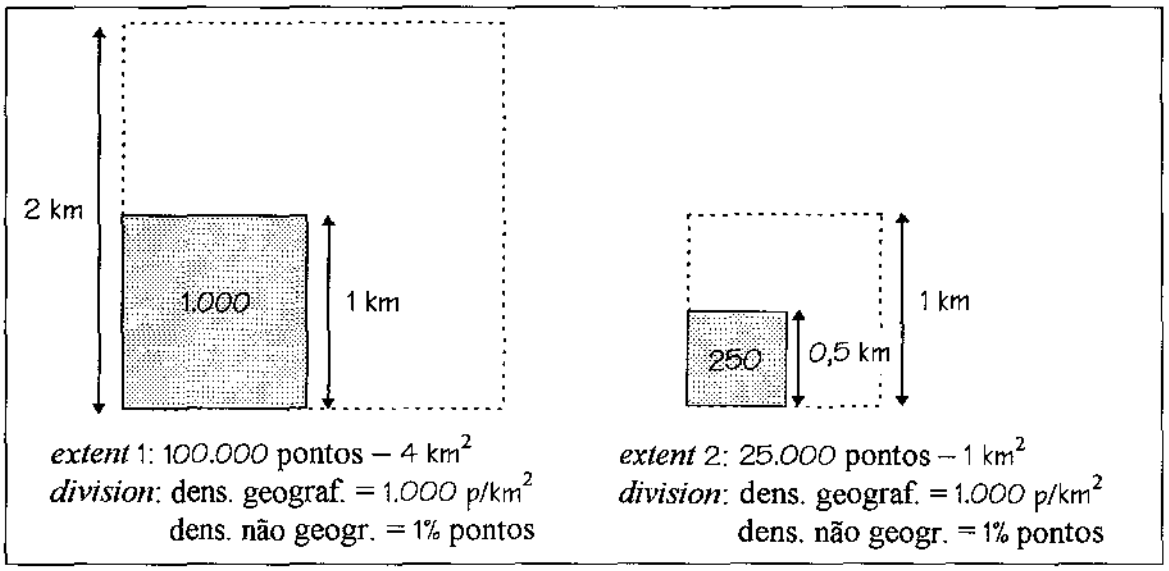


Figura 5.31 Conceitos de densidade relativa geográfica e não geográfica.

5.5.4 Geração de dados espaciais no formato *raster*

Dados *raster* são predominantemente usados para representar dados temáticos. Para isto, categorias são associadas às células de forma a se identificar a localização de características específicas de um tema. Este tipo de dado dá ênfase no conteúdo de áreas geográficas mais que nos seus limites. Assim, propõe-se como primeira forma de geração destes dados, uma geração baseada exclusivamente na percentagem que cada categoria ocupa no *extent*. Para isto, deve-se gerar aleatoriamente a categoria de cada célula, sendo que no final as percentagens de cada tipo de categoria sejam alcançadas. Deste modo, a seletividade de dados *raster* é controlada utilizando-se as percentagens aplicadas às categorias. O uso de percentagens, entretanto, deve ser feito de modo a se produzir quantidades inteiras, evitando-se assim aproximações. Adicionalmente, deve-se fixar a resolução espacial das células e os limites do *extent*.

Um método simples de se garantir a geração correta das percentagens de cada categoria é baseado em representação de lista encadeada, da seguinte forma:

- **passo 1:** fixa-se a quantidade de células por tipo de categoria, em função do tamanho do *extent* e da percentagem de ocupação. Cada categoria distinta deve ocupar um nó separado em uma fila encadeada. Os nós desta fila possuem além da categoria, a quantidade de células desta categoria.

- **passo 2:** gera-se aleatoriamente um número k no intervalo 1..n, onde n representa o número de nós presentes na fila. Inicialmente, n corresponde à quantidade de categorias distintas do tema.

- **passo 3:** procura-se na fila o k-ésimo nó, e associa-se para a célula em questão o valor da categoria deste nó. Em seguida, decrementa-se a quantidade de células deste nó de 1. Quando a quantidade de células de um nó atingir o valor 0, este deve ser removido da fila. Repete-se os passos 2 e 3 até que nenhum nó exista na fila.

Outra forma de se gerar dados *raster*, de modo que sejam gerados OGs-2D bem definidos em todo *extent*, consiste em executar a transação de conversão de dados - *VectorRaster* (seção 5.3.10) para transformar os polígonos gerados segundo as regras descritas na seção anterior para o formato *raster*. A seletividade dos dados *raster* é derivada diretamente do tamanho dos polígonos convertidos, sendo no entanto sujeita à aproximações decorrentes do processo de conversão.

Capítulo 6

Estudo de caso

6.1 SIGs: arquitetura do SGBD x desempenho

O gerenciamento de dados geográficos em SIGs requer o acoplamento de novas técnicas aos diversos componentes de uma arquitetura de um SGBD convencional, além da inserção de novos componentes a esta arquitetura. Estas técnicas e componentes visam organizar os dados de modo que estes sejam armazenados e recuperados eficientemente, respeitando possíveis inter-relacionamentos entre dados espaciais e convencionais. A figura 6.1 ilustra um exemplo de arquitetura de um SGBD não convencional, proposta por [Ooi90]. Esta arquitetura é estruturada em 3 níveis: nível de interface, responsável pela comunicação entre o sistema e o meio externo; nível intermediário, que dentre outras funções deve prover a otimização das consultas feitas pelo usuário; e o nível físico, representando a camada interna de um SGBD, onde encontram-se as funções responsáveis pelo armazenamento e recuperação dos dados em disco.

O processamento de uma consulta geográfica, por exemplo, envolve o acionamento de vários componentes de um SGBD, os quais podem degradar significativamente o desempenho, caso técnicas adequadas não tenham sido implantadas ou o relacionamento entre os componentes desta arquitetura se fizer de forma inapropriada. Primeiramente, uma consulta geográfica necessita ser expressa em uma linguagem de consulta que, além de incorporar operadores convencionais (operadores aritméticos, relacionais e lógicos), incorpore operadores com poder de expressão de propriedades topológicas (tais como interseção e inclusão), de propriedades geométricas (tais como área e comprimento) e de propriedades espaciais (tal como união, a qual gera como resultado o objeto geográfico resultante da união de vários outros objetos geográficos).

Operadores topológicos, geométricos e espaciais, classificados como operadores não convencionais, são a base da formação de predicados espaciais. Operadores aritméticos (tais como adição (+), subtração (-) e multiplicação (*)), relacionais (tais como maior (>),

menor (<), igual (=) e diferente (#)) e lógicos (tais como E (and) e OU (or)) são classificados como operadores convencionais, sendo os dois primeiros a base da formação de predicados convencionais, e o último usado para unir os diversos predicados existentes em uma consulta geográfica.

Linguagens de consultas convencionais não oferecem operadores topológicos, geométricos e espaciais, impossibilitando a realização de uma enorme gama de consultas. Exemplos de linguagens de consulta geográficas[1] propostas na literatura são: *Geo-QUEL* [BS77], *QBE-GeoBase* [BB81], *MapQuery* [Fra82], *Spatially Extended SQL – SESQL* [AS86], *PostQUEL* [SR87], *Geographic Query Language – GeoQL* [Ooi90] e *Spatial SQL* [Ege94].

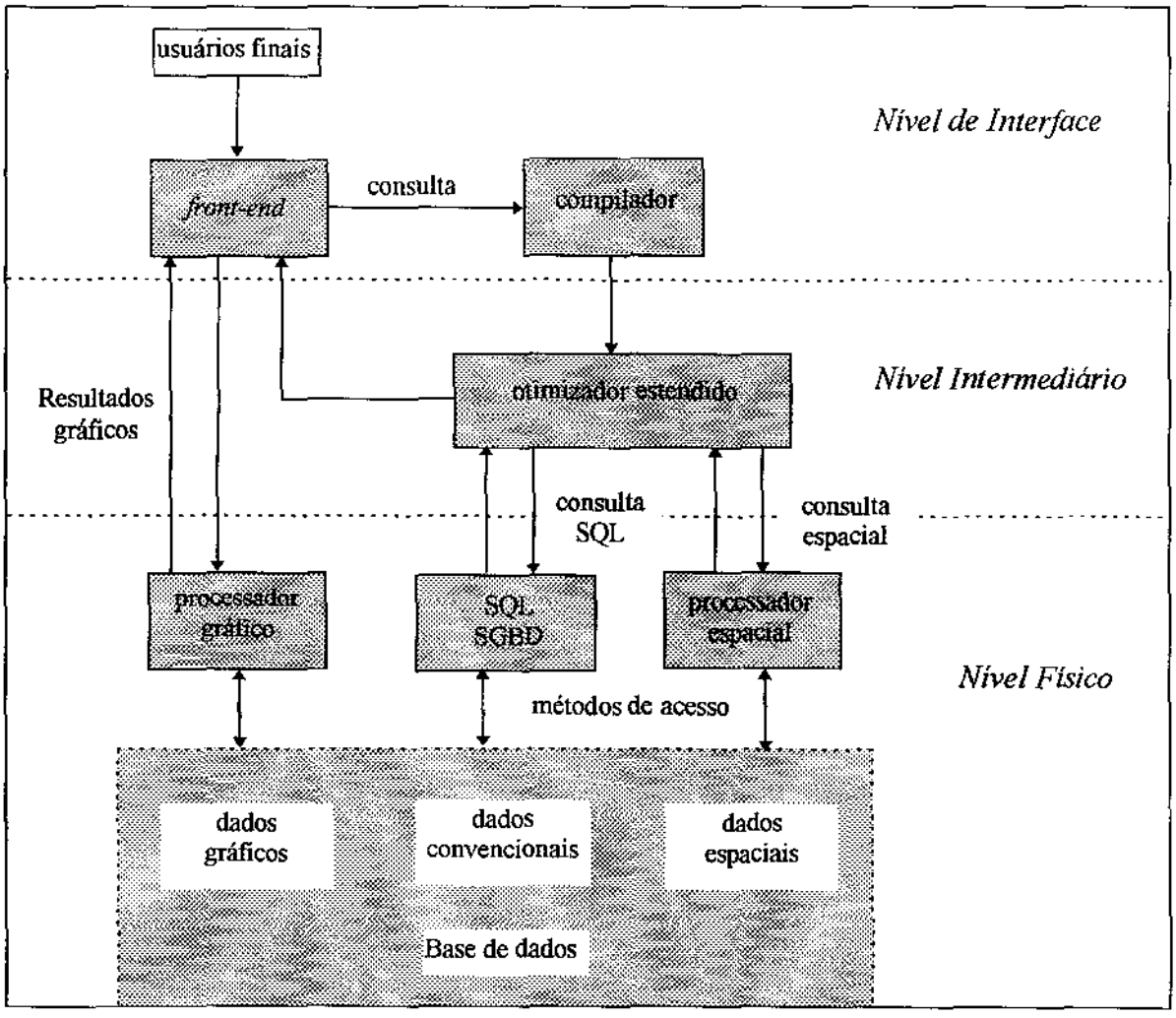


Figura 6.1 Arquitetura de um SGBD não convencional.

[1] comumente denominadas linguagens de consultas espaciais.

Como exemplo, a consulta "Quais cidades são atravessadas pelo rio Tietê e possuem uma população superior a 1.000.000 de habitantes ?" é codificada na linguagem GeoQL:

```
SELECT   CIDADE.nome
FROM     CIDADE, RIO
WHERE    RIO. nome = "Tietê" and
         CIDADE.população > 1.000.000 and
         RIO intersects CIDADE
```

Após a consulta geográfica ser expressa em uma linguagem de consulta apropriada (geográfica), esta é submetida ao compilador desta linguagem, o qual tem por finalidade validar a consulta especificada pelo usuário e transformá-la em uma estrutura apropriada, a ser avaliada pelo otimizador de consulta. Desta forma, o compilador também deve ser estendido de modo a incorporar predicados espaciais presentes em consultas geográficas. Mesmo que a linguagem de consulta e o compilador de um SGBD convencional sejam estendidos para suportar predicados espaciais, a consulta não será efetuada eficientemente, pois o otimizador de consultas, métodos de acesso, e tipos de dados deste SGBD são inapropriados para a manipulação de dados geográficos, sendo projetados somente para o suporte à dados convencionais.

Uma vez estando a consulta geográfica em um formato apropriado que possa ser analisado pelo otimizador de consultas, este avalia todos os predicados envolvidos na consulta, procurando escolher a melhor estratégia a ser utilizada para satisfazê-la. De maneira análoga ao compilador, o otimizador de consultas deve ser estendido para suportar predicados espaciais. Deve-se lembrar que a ordem em que os predicados são processados tem significado decisivo para a eficiência da resposta à consulta, uma vez que estes predicados podem atuar sobre diferentes tamanhos e tipos de conjuntos de dados.

Para facilitar o acesso aos dados armazenados, um SGBD deve fornecer estruturas que providenciem um caminho otimizado aos dados com base em um conjunto definido de predicados sobre os atributos. Estas estruturas são chamadas de métodos de acesso. Para um SGBD convencional, um típico exemplo de método de acesso é a árvore-B^+ (B^+-*Tree*), a qual é construída para suportar os operadores convencionais de igualdade, inferioridade e superioridade entre seus atributos. Outro exemplo é a estrutura *hash* para determinação de igualdade.

Entretanto, a disposição geográfica de dados espaciais impede a simples utilização de métodos de acesso convencionais. O acoplamento de métodos de acesso capazes de otimizarem o acesso aos dados espaciais (chamados de métodos de acesso espaciais) é um requisito essencial para se garantir um desempenho aceitável. Alguns métodos de acesso espaciais encontrados na literatura são: *R-Tree, R-Tree Greene,* R^+*-Tree, R*-Tree, R*-Tree R, Cell Tree, K-d Tree, Spatial K-d Tree, Grid File, Quadtree, MX-CIF Quadtree e Point Quadtree* [Cox91].

[Cox91], em seu trabalho de dissertação, realizou vários testes de desempenho de métodos de acesso espaciais, em particular dos métodos derivados de *multiway trees*, *R-Tree*, *R-Tree Greene*, R^+*-Tree*, *R*-Tree* e *R*-Tree R*. Dentre os fatores determinantes de desempenho constatou-se fatores relacionados exclusivamente aos dados, tais como distribuição, dinâmica, tipos e tamanho dos dados espaciais, e fatores relacionados às consultas, uma vez que nem todos os possíveis tipos de consultas são suportados eficientemente por cada método de acesso.

Nos experimentos foram gerados 14 arquivos de dados, os quais possuíam diferentes configurações de tipos, tamanho (no caso de retângulos) e distribuição. A quantidade de dados (10.000) foi mantida constante em todos os arquivos. Apenas dados espaciais na forma de pontos e retângulos foram gerados, uma vez que todos os métodos de acesso testados trabalham com o conceito de *MBR* (*Minimum Bounding Rectangle - Bounding Box)*. A caracterização do tamanho dos retângulos se fez a partir de duas classes distintas: retângulos pequenos (correspondendo no máximo a 2% do espaço total de cada dimensão do espaço) e retângulos grandes (correspondendo no mínimo a 50% do espaço total de cada dimensão do espaço). A distribuição dos dados foi caracterizada em uniforme e não uniforme. Na primeira, os dados foram gerados por meio de funções geradoras de números uniformes aleatórios (funções randômicas), mapeadas para o intervalo de 0 a 1.000. Dados do tipo ponto foram criados a partir da geração de dois números, já dados do tipo retângulo foram criados a partir da geração de quatro números, sendo os dois primeiros gerados para representar o centro do *MBR* nos eixos x e y, respectivamente; e os demais para representar a distância do centro do *MBR* à sua extremidade nos eixos x e y. Estes dois últimos valores foram mapeados nos intervalos de 0 a 10 e 0 a 250, de acordo com o tamanho dos retângulos criados. Na distribuição não uniforme, a criação de dados do tipo ponto foi feita a partir da determinação dos pontos referentes ao contorno de polígonos quadriláteros. Estes polígonos foram construídos com a geração de quatro pares de números aleatórios referentes a seus vértices. Os retângulos pequenos sob distribuição não uniforme foram gerados segundo a curva senóide, enquanto os retângulos grandes através da curva co-senóide.

A carga de trabalho definida nos experimentos consistia em apenas três tipos de consultas. O primeiro tipo, *point queries*, consistia na determinação de todos os retângulos R tal que um dado ponto P estivesse contido em R, ou seja, uma especialização da transação de busca topológica *Embody* (seção 5.3.3). O segundo tipo, *intersection range queries*, consistia na determinação de todos os retângulos R que intersectavam um dado retângulo R1, ou seja, uma especialização da transação de busca topológica *Intersect*. O terceiro e último tipo, *contingence range queries*, consistia na determinação de todos os retângulos R que estavam contidos em um dado retângulo R1, ou seja, uma especialização da transação de busca topológica *Inside*. Estes três tipos de consulta foram executados, para cada arquivo, segundo três dinâmicas de dados distintas: inserção completa dos 10.000 dados espaciais, remoção de 5.000 dados espaciais, e inserção e remoção de 2.000 dados espaciais.

Como conclusões, [Cox91] verificou que o método *R-Tree* era mais indicado para prover suporte a conjunto de dados compostos de retângulos grandes. Para conjuntos de dados compostos de pontos e retângulos pequenos combinados, o método *R*-Tree* e *R*-Tree R* se mostraram mais apropriados. Adicionalmente, o fato do desempenho obtido nas consultas ter sido preservado nas três dinâmicas de dados comprovou a eficiência das estruturas dinâmicas utilizadas por estes métodos, os quais suportaram a atualização dos dados espaciais. Estas conclusões foram tiradas com base no número de acessos a disco, a única medida de desempenho extraída. Assim, percebe-se que a utilização de um determinado método de acesso pode intercaladamente ocasionar bons e maus resultados de desempenho.

Deve-se distinguir as forma de representação interna dos dados espaciais da forma de representação utilizada pelos métodos de acesso. Pode-se dizer que a ênfase das duas representações são opostas, pois enquanto a representação interna preocupa-se em representar o objeto geográfico através de sua composição (especialização), de forma que possa ser representado e manipulado eficientemente sem perda de precisão, na representação através de métodos de acesso há a preocupação em abstrair (generalizar) os objetos geográficos através da representação destes por formas geométricas mais simples. Para este propósito, os principais métodos de acesso espaciais, derivados de *multiway trees*, utilizam o conceito de *container*. O tipo mais difundido de *container* é o *MBR*, já citado anteriormente, o qual consiste no menor retângulo que circunscreve o objeto geográfico em questão. Outro tipo de *container*, pouco utilizado, é o MBC (*Minimum Bounding Circle)*, o qual consiste no menor círculo que circunscreve o objeto geográfico em questão.

A utilização de *containers* para representar objetos geográficos em métodos de acesso espaciais visa simplificar operações de superposição entre formas geométricas complexas, o que é muito custoso, tornando a operação de superposição uma simples superposição de formas geométricas simples, tal como retângulos. Assim, a busca em métodos de acesso baseados em *containers*, a qual é efetuada através de sucessivas operações de superposição envolvendo os nós das folhas da estrutura do método de acesso, é efetuada mais rapidamente, uma vez que se o *container* do objeto geográfico não fizer parte da resposta da consulta então o objeto geográfico por si só também não satisfará a consulta.

Entretanto, os objetos geográficos selecionados por um método de acesso não podem ser considerados, a priori, como respostas precisas a uma consulta, sem uma posterior análise de sua geometria. Isto ocorre porque *containers* introduzem uma parcela de erro, especialmente se a geometria do objeto geográfico for distribuída de forma irregular no *container* (como um letra L ou uma linha diagonal, por exemplo). Nestes casos o *container* representa apenas uma aproximação grosseira da geometria do objeto geográfico representado. Na figura 6.2 verifica-se que certa porção do espaço, contida dentro do *container*, não faz referência ao dado em si. Este espaço, denominado *dead space*, pode causar a recuperação deste objeto geográfico através da determinação de algum relacionamento espacial sobre esta área. Pode-se dizer que métodos de acesso espaciais

funcionam apenas como um filtro, selecionando rapidamente e da maneira mais restrita possível candidatos que possam satisfazer uma dada consulta.

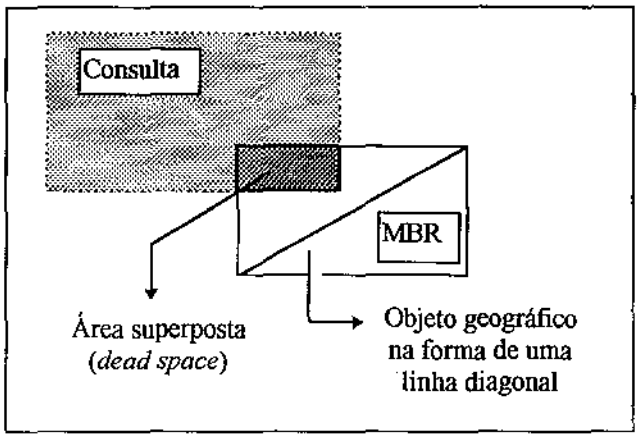


Figura 6.2 Exemplo de uma recuperação errada de um objeto geográfico com *MBR*.

Após a seleção dos possíveis candidatos, para cada um deles deve ser recuperada a representação precisa de sua geometria (o que pode ser muito demorado), com o propósito de se realizar novos cálculos analíticos. Esta fase é chamada de refinamento da consulta. Atualmente, existe uma grande discussão sobre a forma da representação dos objetos geográficos nas estruturas dos métodos de acesso espaciais. A utilização de aproximações muito simples, tal como *MBR*, com o intuito de rapidamente se obter um conjunto de candidatos, muitas vezes pode gerar um número elevado destes, para cada um dos quais deve-se efetuar o refinamento. Assim, o uso de formas geométricas um pouco mais complexas (mais não muito), tal como polígonos limitados por um certo número de vértices, pode produzir um desempenho melhor. Isto se deve, teoricamente, a um ganho de tempo decorrente do refinamento de poucos candidatos, em detrimento de uma busca rápida destes candidatos. Resultados de desempenho preliminares divulgados por [GRY95] mostram que esta alternativa pode melhorar o desempenho de árvores *R*-Tree* sem que modificações bruscas sejam feitas nos algoritmos de inserção, remoção e consulta. Já [KHS91] propõe a decomposição de objetos geográficos complexos em objetos geográficos menores. Cada objeto geográfico menor seria então aproximado por um *container* específico, tal como um trapezóide ou retângulo. Alguns resultados preliminares de desempenho são reportados por este autor, mostrando algumas vantagens desta técnica.

Aplicações convencionais possuem como tipos de dados (ou atributos) basicamente valores numéricos e alfanuméricos. A representação destes tipos de dados nos SGBDs não requer nenhuma estratégia específica dos sistemas de banco de dados, uma vez que fazem parte do conjunto básico fornecido pelas linguagens de programação nas quais SGBDs são codificados: *integer*, *float*, *char*, *string*, dentre outros.

Aplicações geográficas, por sua vez, para representar a geometria de objetos geográficos utilizam tipos de dados que modelam os conceitos de ponto, linha e polígono

no espaço bidimensional (dados espaciais). Estes tipos são representados basicamente em dois formatos de dados: *raster* e vetorial (seção 2.3). Sendo estes tipos inexistentes nas linguagens de programação, para que SGBDs convencionais possam provê-los, a fim de que sejam utilizados pelas aplicações geográficas como tipos de dados básicos, estes devem possuir um mecanismo de representação destes dados nos tipos mais simples já comentados. Embora a abstração de dados multidimensionais (dados espaciais) em dados escalares (dados convencionais como *integer* e *string*) possa ser efetuada por estratégias distintas, o suporte às operações sobre estes dados, acesso e recuperação, são fatores decisivos na escolha da representação interna dos dados espaciais pelos SGBDs.

Exemplificando a influência da estratégia de representação interna de dados no desempenho de um SGBD não convencional, deve-se supor uma aplicação geográfica com a representação de um mapa através de uma escala em dezenas de metros. Assim, uma avenida de 1 km de extensão por 10 m de largura pode ser representada pelo conjunto de 100 pontos, onde cada ponto através de sua coordenada x,y, representa 100 m^2. Segundo esta forma de representação, toda vez que for preciso recuperar ou gravar um dado deste tipo, será feita a leitura/gravação de 100 pares de valores numéricos. Além da grande quantidade de espaço necessária para representação de um único dado, este processo de leitura/gravação é extremamente ineficiente. Quanto ao suporte às operações espaciais, esta estratégia em nada facilita a determinação dos relacionamentos espaciais entre objetos geográficos. A determinação de relacionamentos como interseção e proximidade, por exemplo, ocasionam algoritmos complexos.

Assim, a representação de dados espaciais utilizando-se apenas os tipos básicos convencionais torna-se inapropriado. Um SIG deve, portanto, fornecer tipos especiais para a representação de dados espaciais. O tamanho e a forma de organização desta representação podem influenciar no desempenho. Adicionalmente, simplificações podem ser adotadas em nível de aplicação para minimizar esforços de armazenamento e de tempo de busca. Um exemplo é a utilização de um ponto para representar uma cidade ou um lago, dependendo da abstração requerida. Assim, um objeto geográfico pode ter distintas representações internas em função do nível de detalhe desejável. Um exemplo de método de armazenamento espacial específico para longas seqüências de pontos, chamado *V-Tree*, é encontrado em [MCD94, Med95]. Segundo estes autores, transações em SIGs muitas vezes visam recuperar apenas partes de um objeto geográfico grande, o que implica em acrescentar técnicas especiais, em particular no método de acesso e no armazenamento interno, de modo a recuperar rapidamente estas partes sem ter que recuperar inteiramente os dados espaciais. Já [Lu+91] descreve um arquitetura genérica para o gerenciamento do armazenamento de dados geográficos.

A visualização de objetos geográficos como resultado a uma consulta requer a busca de dados gráficos associados a todos os dados espaciais / convencionais envolvidos, ou a sua derivação a partir destes. Em adição, esta resposta é altamente dependente de um contexto associado. Isto significa que a resposta a uma consulta do tipo "Quais as 10 cidades mais próximas de Campinas ?" não deve retornar isoladamente apenas os dados gráficos relativos aos objetos geográficos (cidades) que satisfazem a consulta. Isto não faria

sentido. Também não faria sentido a visualização gráfica destas cidades em uma mapa continental, desde que o objetivo desta consulta enfoca claramente um contexto regional. Assim, a visualização gráfica do resultado de uma consulta geográfica requer a visualização tanto dos dados espaciais e convencionais associados diretamente aos objetos geográficos que satisfazem a consulta, quanto dos dados espaciais e convencionais associados aos objetos geográficos que visam posicionar geograficamente a resposta da consulta (contexto associado).

Devido à complexa estrutura inerente à arquitetura de um SGBD não convencional, a análise independente de componentes isolados, apesar de ser importante no sentido de se apontar deficiências e virtudes nestes componentes, pode não refletir o desempenho global de um SIG. O *benchmark* proposto por este trabalho visa exercitar todos os componentes de um SGBD não convencional, tornando-se deste modo uma técnica voltada à análise do desempenho global de SIGs. Segundo [GRY95], testes envolvendo dados reais de aplicações georeferenciadas demonstraram que existe uma relação direta entre o número de páginas lidas e o tempo de resposta para se completar determinadas transações, o que sugere uma tendência *I/O bound*. A próxima seção apresenta uma aplicação alvo sintética na qual a carga de trabalho definida no capítulo anterior poderá atuar.

6.2 Aplicação alvo sintética

Esta seção descreve uma aplicação alvo georeferenciada composta de dados sintéticos. O inter-relacionamento entre as transações e os dados é definido rigorosamente através da caracterização da estrutura do banco de dados, tipos de dados, quantidade de dados, distribuição, controle de seletividade, tipo de *bufferização*, arquivos sobre os quais cada transação irá atuar e seletividade das transações. A caracterização dos dados espaciais é efetuada exclusivamente para dados no formato vetorial. A utilização de dados no formato *raster*, entretanto, pode ser estendida através da conversão dos dados vetoriais para o formato *raster* (transação *VectorRaster*) e posterior aplicação das transações.

A utilização de dados sintéticos permite a adição de características especiais, de modo a explorar deficiências encontradas ou pressentidas em aplicações georeferenciadas, as quais podem degradar significativamente o desempenho. Para isto, segue-se os procedimentos descritos na seção 5.5. Em especial, a geração de linhas e polígonos é efetuada de forma a variar a quantidade, *shape*, tamanho e complexidade. Assim, por exemplo, *shapes* que maximizam o percentual de *dead space*, assim como *divisions* com quantidade elevada de pontos, linhas e polígonos são explorados pelas transações, com o intuito de degradar o desempenho.

A geração de dados vetoriais proposta é baseada nos conceitos de *extent* e *division* (seção 5.5.3). O *extent* considerado pela aplicação alvo consiste em um quadrado de 1.000 km de lado, formando uma área de 1.000.000 km^2. A figura 6.3 ilustra a divisão

hierárquica do *extent* adotada para a geração dos dados espaciais vetoriais de todos os objetos geográficos desta aplicação alvo. O *extent* considerado possui 7 *divisions*, nomeadas de A a G. As *divisions* A, F e G são resultantes de uma primeira divisão do *extent* segundo o modelo *quadtree*. Já as *divisions* B, C, D e E são resultantes de uma segunda divisão do *extent*.

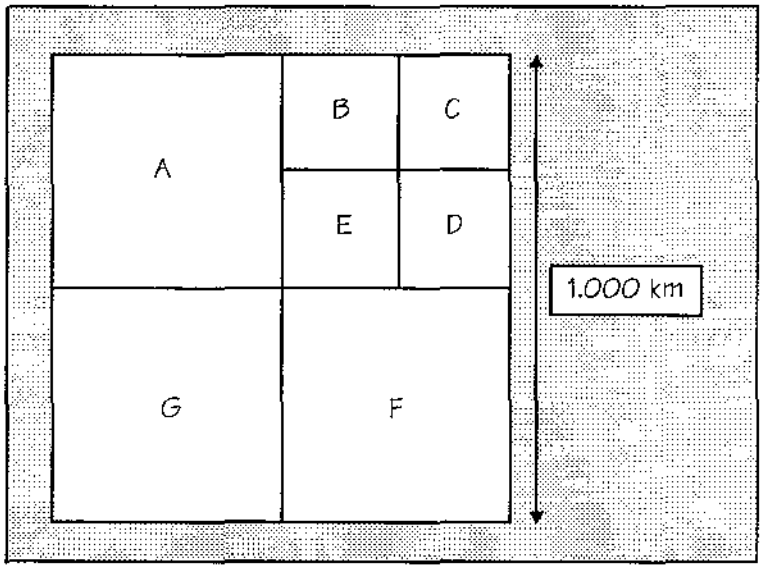


Figura 6.3 Aplicação alvo: *extent* e *divisions*.

O banco de dados considerado engloba quatro classes de objetos geográficos distintos: cidades, estradas, solos e vegetação. Para cada uma destas classes são descritos um conjunto de dados espaciais e convencionais. Estes dados podem ser armazenados em um mesmo arquivo ou em vários arquivos distintos, desde que sejam mantidos os tipos de dados, quantidade e distribuição. As classes acima descritas refletem classes de objetos geográficos encontradas comumente em aplicações georeferenciadas. Entretanto, para as classes cidades, estradas e solos, a quantidade de objetos geográficos é propositadamente superestimada. Assim, por exemplo, a quantidade de objetos geográficos fixada para a classe cidades é de 1.000.000, uma quantidade superior à encontrada na prática.

A geração dos objetos geográficos de cada classe deve ser efetuada por *division*, sendo primeiramente gerados objetos geográficos para a *division* A, depois para a B, depois para a C e assim sucessivamente até a última *division* G. Para cada objeto geográfico gerado deve-se gerar todos os dados convencionais deste. Assim, pode-se dizer que os algoritmos de geração de dados são paralelos, sendo os algoritmos de geração de dados convencionais orientados segundo os algoritmos de geração de dados espaciais.

Os dados relativos às classes e suas distribuições são descritos a seguir.

Classe Cidades: composta de objetos geográficos cujos dados espaciais são formados exclusivamente por pontos armazenados segundo o formato vetorial. Para cada ponto deve-se armazenar as suas coordenadas x e y (*float*). Adicionalmente, para cada cidade também deve-se armazenar os seguintes dados convencionais: chave (*long integer*), nome (*string[30]*), data de fundação (*date*), população (*long integer*), número de hospitais (*integer*) e altitude (*float*). Em especial, o atributo chave visa à identificação unitária das cidades geradas. Somente para esta classe serão descritas transações voltadas à análise de atributos convencionais. No total devem ser geradas 1.000.000 de cidades, eqüivalendo aproximadamente a 51,5 *Mbytes*. A tabela 6.1 descreve a quantidade de cidades, a seletividade, a razão cidades/km^2 e o espaço de armazenamento requerido por *division*. Os dados convencionais de cada cidade ocupam 46 *bytes*, sendo 4 *bytes* para a chave, 30 *bytes* para o nome, 2 *bytes* para a data de fundação, 4 *bytes* para a população, 2 *bytes* para número de hospitais e 4 *bytes* para a altitude. Já os dados espaciais de cada cidade ocupam 8 *bytes*, 4 *bytes* para cada uma das coordenadas x e y. Assim, no total, cada cidade ocupa 54 *bytes*.

division	quantidade	seletividade	cidades/km^2	Mbytes
A	150.000	15%	0,6	7,72
B	10.000	1%	0,16	0,51
C	400.000	40%	6,4	20,6
D	250.000	25%	4	12,87
E	100.000	10%	1,6	5,15
F	10.000	1%	0,04	0,51
G	80.000	8%	0,32	4,12
Total =	1.000.000	100%	1	51,48

Tabela 6.1 Caracterização da geração de cidades.

A geração de dados convencionais segue as discussões efetuadas na seção 5.5.2.

- O atributo chave deve ser gerado aleatoriamente no intervalo 1..1.000.000, segundo uma distribuição uniforme única.

- O atributo nome deve ser gerado segundo o modelo $X Y_1Y_2 ..Y_7 Z_1Z_2 .. Z_{22}$. O primeiro caractere X deve ser gerado segundo uma distribuição *Zipf*, com z = 1. Para isto, os valores e *ranks* considerados são: letra A (48%), letra B (24%), letra C (16%) e letra D (12%). Os caracteres Y_n devem ser gerados seqüencialmente a partir do valor 1. Os caracteres Z_m devem ser simplesmente preenchidos, como exemplo por #.

- O atributo data deve ser gerado aleatoriamente segundo uma distribuição uniforme para dia, mês e ano. Entretanto, para cada um destes componentes é adotado uma percentagem e domínio distintos. Os componentes dia, mês e ano devem ser gerados, respectivamente, nos intervalos 1..25, 1..10 e 1976..1995, onde cada valor distinto corresponde a 4%,10% e 5%, na mesma ordem.

- O atributo população deve ser gerado segundo uma distribuição *Zipf*, com z = 1, para as seguintes faixas de valores: faixa 1 (5.000.000 < i ≤ 13.000.000), faixa 2 (1.000.000 < i ≤ 5.000.000), faixa 3 (500.000 < i ≤ 1.000.000), faixa 4 (100.000 < i ≤ 500.000), faixa 5 (10.000 < i ≤ 100.000) e faixa 6 (i ≤ 10.000). Os *ranks* das faixas 1..6 são, respectivamente, 6,8%, 8,2%, 10,2%, 13,6%, 20,4% e 40,8%. Uma vez gerado a faixa, é gerado aleatoriamente um valor dentro da faixa, sem nenhum controle de distribuição.

- O atributo número de hospitais deve ser gerado segundo uma distribuição *Zipf*, com z = 1. Para isto, os valores e *ranks* considerados são: 0 (34%), 1 (17%), 2 (11%), 3 (9%), 4 (7%), 5 (6%), 6 (5%), 7 (4%), 8 (4%) e 9 (3%).

- O atributo altitude deve ser gerado segundo uma distribuição *Zipf*, com z = 1, para as seguintes faixas de valores: faixa 1 (400 < i ≤ 1.000), faixa 2 (300 < i ≤ 400), faixa 3 (200 < i ≤ 300), faixa 4 (100 < i ≤ 200) e faixa 5 (i ≤ 100). Os *ranks* das faixas 1..5 são, respectivamente, 8,8%, 10,9%, 14,6%, 21,9% e 43,8%. Uma vez gerado a faixa, é gerado aleatoriamente um valor dentro da faixa, sem nenhum controle de distribuição.

Classe Estradas: composta de objetos geográficos cujos dados espaciais são formados por linhas armazenadas segundo o formato vetorial. Para cada linha deve-se armazenar ordenadamente os pontos que a formam. Adicionalmente, para cada estrada deve-se armazenar os seguintes dados convencionais: chave (*long integer*), nome (*string[30]*) e comprimento (*float*). Em especial, o atributo chave visa à identificação unitária das estradas geradas. No total devem ser geradas 1.000.000 de estradas, eqüivalendo aproximadamente a 121,1 *Mbytes*. A tabela 6.2 descreve a quantidade de estradas, o *shape*, o tamanho, a complexidade, a seletividade, a razão estradas/km^2 e o espaço de armazenamento requerido por *division*. Os dados convencionais de cada estrada ocupam 38 *bytes*, sendo 4 *bytes* para a chave, 30 *bytes* para o nome e 4 *bytes* para o comprimento. Já os dados espaciais de cada estrada ocupam 16 ou 400 *bytes*, para uma complexidade de 2 e 50 pontos respectivamente. Assim, no total, as estradas ocupam 54 ou 438 *bytes*, dependendo da complexidade.

division	quantidade	*shape*	tamanho	complexidade	seletividade	estradas/ km^2	*Mbytes*
A	150.000	aleatório	[5-95]	2	15%	0,6	7,72
B	10.000	A e B	[5-10]	2	1%	0,16	0,51
C	400.000	C e D	[35-65]	2	40%	6,4	20,6
D	250.000	aleatório	[70-95]	2	25%	4	12,87
E	100.000	aleatório	[5-95]	50	10%	1,6	41,77
F	10.000	A e B	[70-95]	50	1%	0,04	4,18
G	80.000	C e D	[70-95]	50	8%	0,32	33,42
Total =	1.000.000			11,12	100%	1	121,07

Tabela 6.2 Caracterização da geração de estradas.

A geração de dados convencionais é idêntica à realizada para a classe cidades. Em especial, o atributo comprimento consiste em um atributo derivado, devendo este ser calculado a partir dos pontos limítrofes de cada linha gerada.

Classe Vegetação: composta de objetos geográficos cujos dados espaciais são formados por polígonos armazenados segundo o formato vetorial. Os pontos que formam cada polígono devem ser armazenados ordenadamente. Polígonos poderão ter no mínimo 3 pontos (*shape* B – figura 5.28) e no máximo 50 pontos. Adicionalmente, para cada objeto geográfico desta classe deve-se armazenar os seguintes dados convencionais: chave (*integer*) e área (*float*). Em especial, o atributo chave visa à identificação unitária dos objetos geográficos gerados para esta classe. O atributo chave também deve ser usado para simular categoria. No total devem ser gerados 38 objetos geográficos, eqüivalendo aproximadamente a 12,92 *Kbytes*. Esta classe corresponde a polígonos ditos grandes, devido à pequena quantidade de polígonos por *division*. A tabela 6.3 descreve a quantidade, o *shape*, a complexidade, a seletividade, o tamanho dos polígonos e o espaço de armazenamento requerido por *division*. Os dados convencionais ocupam 6 *bytes*, sendo 2 *bytes* para a chave e 4 *bytes* para a área. Já os dados espaciais ocupam 24, 32, 48 ou 400 *bytes*, para uma complexidade de 3, 4, 6 e 50 pontos respectivamente. Assim, no total, os objetos geográficos desta classe ocupam 30, 38, 54 ou 406 *bytes*, dependendo da complexidade.

division	quantidade	*shape*	complexidade	seletividade	tamanho (km^2)	*Kbytes*
A	2	B2	3	5,26%	125.000	0,06
B	8	A	50	21,05%	7812,5	3,17
C	8	C	50	21,05%	7812,5	3,17
D	8	B	50	21,05%	7812,5	3,17
E	8	aleatório	50	21,05%	7812,5	3,17
F	2	C2	6	5,26%	125.000	0,11
G	2	A2	4	5,26%	125.000	0,07
Total =	38		42,79	100%	26.316	12,92

Tabela 6.3 Caracterização da geração de vegetação.

Como citado anteriormente, a geração dos objetos geográficos de cada classe deve ser efetuada por *division*, sendo primeiramente gerados objetos geográficos para a *division* A, depois para a B, e assim sucessivamente até a *division* G. Entretanto, a atribuição do valor do atributo chave para os polígonos das *divisions* A e F deve ser efetuada no sentido esquerda para direita e a atribuição do valor do atributo chave para os polígonos da *division* G deve ser efetuada no sentido de cima para baixo, conforme ilustrado na figura 6.4.

A geração de dados convencionais é idêntica à realizada para a classe cidades. Em especial, o atributo área consiste em um atributo derivado, devendo este ser calculado a partir dos dados espaciais de cada polígono gerado.

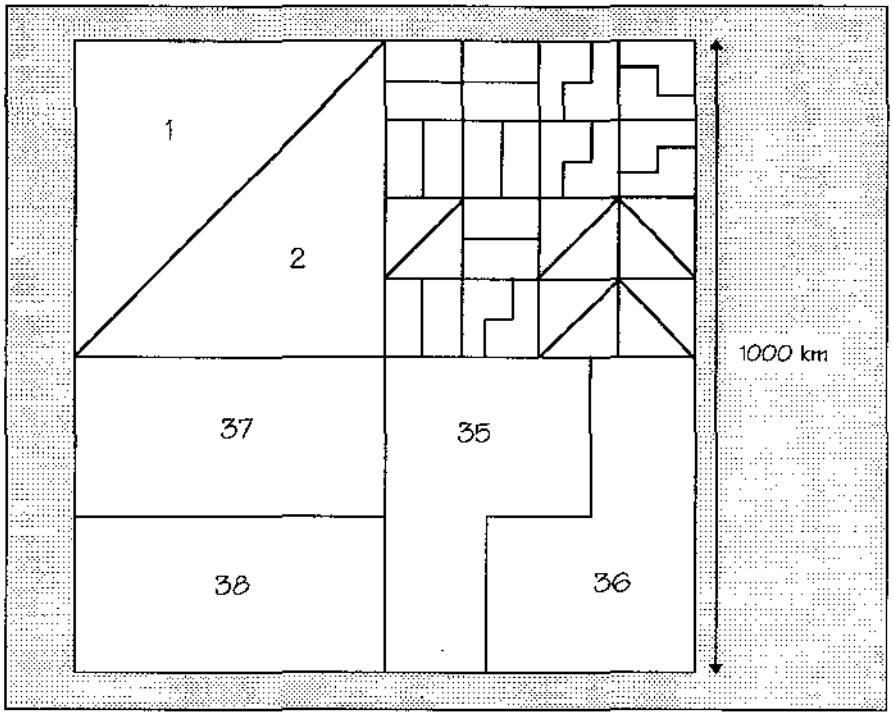


Figura 6.4 Atribuição de chaves.

Classe Solos: composta de objetos geográficos cujos dados espaciais são formados por polígonos armazenados segundo o formato vetorial. Os pontos que formam cada polígono devem ser armazenados ordenadamente. Polígonos poderão ter no mínimo 3 pontos (*shape* B – figura 5.28) e no máximo 50 pontos. Em adição, para cada objeto geográfico desta classe deve-se armazenar os seguintes dados convencionais: chave (*long integer*) e perímetro (*float*). Em especial, o atributo chave visa à identificação unitária dos objetos geográficos gerados para esta classe. O atributo chave também deve ser usado para simular categoria. No total devem ser gerados 2.097.152 objetos geográficos, eqüivalendo a 635 *Mbytes*. Esta classe gera polígonos ditos pequenos, devido à grande quantidade de polígonos por *division*. A tabela 6.4 descreve a quantidade, o *shape*, a complexidade, a seletividade e o espaço de armazenamento requerido por *division*. O tamanho e a razão polígonos/km^2 são iguais para todos os objetos geográficos desta classe, sendo seus valores 0,48 km^2 e 2,1 respectivamente. Os dados convencionais ocupam 8 *bytes*, sendo 4 *bytes* para a chave e 4 *bytes* para a área. Já os dados espaciais ocupam 24, 32, 48 ou 400 *bytes*, para uma complexidade de 3, 4, 6 e 50 pontos respectivamente. Assim, no total, os objetos geográficos desta classe ocupam 32, 40, 56 ou 408 *bytes*, dependendo da complexidade.

A geração de dados convencionais é idêntica à realizada para a classe cidades. Em especial, o atributo perímetro consiste em um atributo derivado, devendo este ser calculado a partir dos dados espaciais de cada polígono gerado.

division	quantidade	shape	complexidade	seletividade	Mbytes
A	524.288	B2	50	25,00%	204
B	131.072	A	4	6,25%	5
C	131.072	C	6	6,25%	7
D	131.072	B	3	6,25%	4
E	131.072	aleatório	6	6,25%	7
F	524.288	C2	50	25,00%	204
G	524.288	A2	50	25,00%	204
Total =	2.097.152		38,69	100%	635

Tabela 6.4 Caracterização da geração de solos.

A quantidade de *bytes* descrita para os objetos geográficos de cada uma das classes serve apenas como uma estimativa do espaço de armazenamento requerido, uma vez que o uso de técnicas de compactação, assim como a estruturação de dados espaciais no formato vetorial pode reduzir drasticamente as quantidades estimadas (seção 5.5.1). Na descrição de cada classe, o importante é a caracterização precisa dos tipos de dados, quantidade de objetos, além da forma de geração dos dados.

Não será permitido o armazenamento explícito de qualquer relacionamento espacial entre as classes cidades, estradas, vegetação e solos. A inserção de relacionamentos de adjacência, inclusão e interseção, por exemplo, apesar de importante para aplicações que fazem uso muito freqüente destes relacionamentos sem a necessidade de atualização periódica (uma vez que colabora para diminuir o tempo de resposta, tornando não necessário o cálculo destes relacionamentos durante a execução de uma transação), implica em um grande aumento no espaço de armazenamento requerido, muitas vezes inviáveis na prática. SIGs devem possuir a capacidade de determinar de forma *ad-hoc* qualquer tipo de relacionamento espacial. Para isto, estes devem ser auxiliados por eficientes métodos de acesso espaciais e funções analíticas.

A aplicação alvo aqui descrita visa à análise de sistemas sem carga de trabalho externa ao *benchmark*. Para isto, no momento de execução das transações deve-se assegurar que a carga de trabalho externa seja reduzida. Para reportar o desempenho propõe-se o uso das medidas de tempo de resposta e quantidade de *I/O*. O espaço de medição para o tempo de resposta deve englobar desde o envio dos primeiros *bytes* da transação (acionamento da transação) até a visualização dos resultados gerados. Entretanto, para se isolar a influência de placas gráficas, sugere-se duas medições de tempo de resposta: uma após a execução da transação e outra após a visualização dos dados gerados pela transação. A seguir é esquematizado o esquema de medição proposto (figura 6.5). Note que a diferença (T2-T1) isola o tempo gasto somente para se visualizar a transação. O tempo necessário para se obter os parâmetros das transações, caso necessário, não deve ser computado e nem incluído no esquema de medição. Parâmetros devem ser obtidos anteriormente à execução da transação, na fase de preparação desta.

{*reset time*: $T = 0$}
transaction (...) $\Rightarrow X$
{*get time T1, get I/O* – execução}
display (X, default)
{*get time T2, get I/O* – visualização}

Figura 6.5 Esquema de medição.

A seguir são descritos os dados sobre os quais a carga de trabalho do *benchmark* agirá. Para cada transação será especificado a seletividade requerida, segundo a distribuição de dados previamente efetuada. A ordem na qual as transações são executadas é irrelevante, uma vez que o desempenho é reportado individualmente para cada transação. Entretanto, antes da execução de cada transação o *buffer* deve ser limpo. Isto é efetuado para inibir o aproveitamento dos dados utilizados e gerados por uma transação anterior.

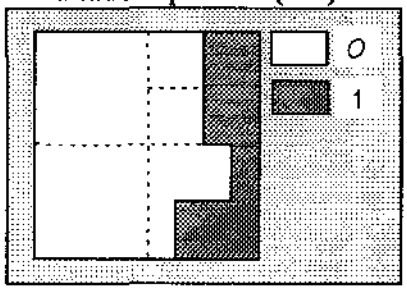
Além disto, não será apresentada nenhuma medida de desempenho que agrupe os diversos resultados gerados em um único número. A utilização de tal medida, comumente utilizada para tornar a comparação de desempenho mais simples, faz necessário a atribuição de pesos a cada transação da carga de trabalho. Estes pesos tendem a refletir a freqüência relativa de execução, ou seja, a importância de cada transação. A atribuição de pesos a um conjunto relativamente grande de transações (como exemplo, para a carga de trabalho definida nesta dissertação) é difícil de ser validado e portanto não será efetuado.

T1: Reclassificação

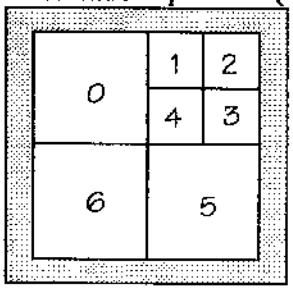
Esquema de medição:

(T1.1) *Reclass* (vegetação, {(0,{1-10, 27-35, 37-38}), (1,{11-26, 36})})
(T1.2) *Reclass* (solos, {(0,{1-524.288}), (1,{524.289-655.360}),
(2,{655.361-786.432}), (3,{786.433-917.504}),
(4,{917.505-1.048.576}), (5,{1.048.577-1.572.864}),
(6, {1.572.865-2.097.152})})

Resultado esperado: (T1.1)



Resultado esperado: (T1.2)



T2: Superposição

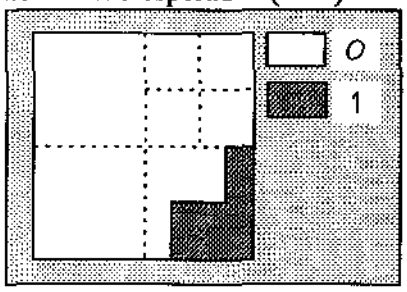
fase de preparação:

Reclass (vegetação, {(0,{1-10, 27-35, 37-38}), (1,{11-26, 36})}) ⇒ R1 **(idem (T1.1))**
Reclass (solos, {(0,{1-1.048.576, 1.572.865-2.097.152}),
(1,{1.048.577-1.572.864})}) ⇒ R2
Inside (*Division*-E, solos, *true*) ⇒ R3

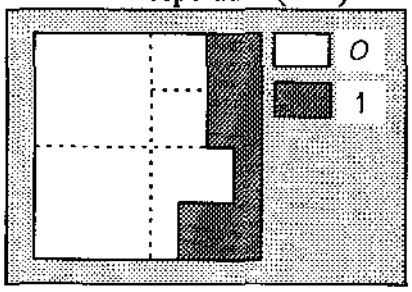
Esquema de medição:

(T2.1) *Overlay* (vegetação, solos, *conventional*)
(T2.2) *Overlay* (vegetação, vegetação, *add*)
(T2.3) *Overlay* (solos, solos, *add*)
(T2.4) *Overlay* (R1, R2, *and*)
(T2.5) *Overlay* (R1, R2, *or*)
(T2.6) *Overlay* (vegetação, R3, *cutter*)

Resultado esperado: (t2.4)



Resultado esperado: (t2.5)

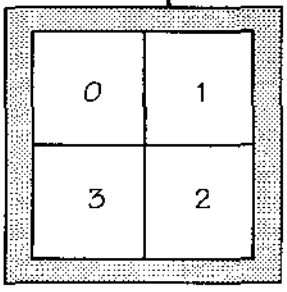


T3: Análise de ponderação

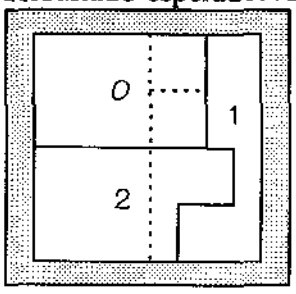
fase de preparação:

Reclass (solos, {(0,{1-2}), (1,{3-34}), (2,{35-36}), (3,{37-38})}) ⇒ R1
Reclass (solos, {(0,{1-10, 27-34}), (1,{11-26, 36}), (2,{35, 37-38})}) ⇒ R2

Resultado esperado: R1



Resultado esperado: R2



Esquema de medição:

(T3.1) *Weigh* (R1, R2, *add*, *weight*)

R1: cat_1=0, cat_2=1, cat_3=2, cat_4=3 | R2: cat_1=0, cat_2=1, cat_3=2

weight = R2.cat_1 (2^6), R2.cat_2 (2^5), R1.cat_1 (2^4), R2.cat_3 (2^3), R1.cat_2 (2^2), R1.cat_3 (2^1) e R1.cat_4 (2^0)

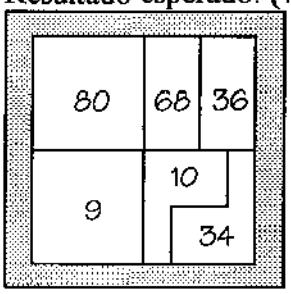
(T3.2) *Weigh* (R1, R2, *table*, *weight*)

R1: cat_1=0, cat_2=1, cat_3=2, cat_4=3 | R2: cat_1=0, cat_2=1, cat_3=2

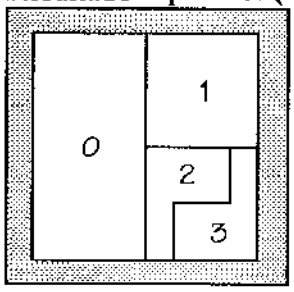
	R1.cat_1	R1.cat_2	R1.cat_3	R1.cat_4
R2.cat_1	0	1	1	1
R2.cat_2	1	1	3	2
R2.cat_3	3	2	2	0

Tabela 6.5 *Weight*: t3.2.

Resultado esperado: (T3.1)



Resultado esperado: (T3.2)



T4: Análise de proximidade

fase de preparação:

Select (vegetação.chave = 31, vegetação) ⇒ R1 [*Division*-E]
Select (solos.chave = 589.824, solos) ⇒ R2 [*Division*-B]
Select (solos.chave = 720.896, solos) ⇒ R3 [*Division*-C]
Select (solos.chave = 851.968, solos) ⇒ R4 [*Division*-D]
Select (solos.chave = 983.040, solos) ⇒ R5 [*Division*-E]
Inside (*Division*-B, estradas, *true*) ⇒ R6
Inside (*Division*-C, estradas, *true*) ⇒ R7
Inside (*Division*-C, cidades, *true*) ⇒ R8
Inside (*Division*-F, cidades, *true*) ⇒ R9
R10 = centro do *extent*

Esquema de medição:

(T4.1) *SimpleBuffer* (R1, 10.000, *exterior*)
(T4.2) *SimpleBuffer* (R1, 10.000, *interior*)
(T4.3) *SimpleBuffer* (R1, 10.000, *both*)
(T4.4) *MultipleBuffer* (R1, 10.000, 5, *exterior*)
(T4.5) *MultipleBuffer* (R1, 10.000, 5, *interior*)
(T4.6) *MultipleBuffer* (R1, 10.000, 5, *both*)
(T4.7) *SimpleBufferMultiple* ({R2, R3, R4, R5}, 10.000)
(T4.8) *MultipleBufferMultiple* ({R2, R3, R4, R5}, 10.000, 5)
(T4.9) *SimpleBufferMultiple* (R6, 100)
(T4.10) *SimpleBufferMultiple* (R7, 10.000)
(T4.11) *MultipleBufferMultiple* (R6, 100, 5)
(T4.12) *MultipleBufferMultiple* (R7, 10.000, 5)
(T4.13) *SimpleBufferMultiple* (R8, 10.000)
(T4.14) *SimpleBufferMultiple* (R9, 100)
(T4.15) *SimpleBuffer* (R10, 10.000, *square*)

T5: Interior e fronteira de OGs-2D

fase de preparação:

Select (vegetação.chave = 31, vegetação) ⇒ R1 [*Division*-E]

Esquema de medição:

(T5.1) *GetInterior* (R1)
(T5.1) *GetBoundary* (R1)

T6: Transações topológicas *booleanas*

fase de preparação:

Select (vegetação.chave = 1, vegetação) ⇒ R1 [*Division*-A]
Select (vegetação.chave = 2, vegetação) ⇒ R2 [*Division*-A]
Select (vegetação.chave = 36, vegetação) ⇒ R3 [*Division*-F]
Select (solos.chave = 262.144, solos) ⇒ R4 [*Division*-A]
Select (estradas.chave = 155.000, estradas) ⇒ R5 [*Division*-B]
Select (estradas.chave = 685.000, estradas) ⇒ R6 [*Division*-D]
Select (solos.chave = 851.968, solos) ⇒ R7 [*Division*-D]
Inside (*Division*-A, cidades, *true*) ⇒ R8
Inside (*Division*-A, estradas, *true*) ⇒ R9

Inside (*Division*-A, solos, *true*) ⇒ R10
Inside (*Division*-C, solos, *true*) ⇒ R11
Inside (*Division*-B, estradas, *true*) ⇒ R12
Inside (*Division*-D, estradas, *true*) ⇒ R13
R14 ⇒ R12 - R5

Esquema de medição:

(T6.1) *Meet* (R1, R2) ⇒ *true*
(T6.2) *NotMeet* (R1, R2) ⇒ *false*
(T6.3) *Meet* (R1, R3) ⇒ *false*
(T6.4) *NotMeet* (R1, R3) ⇒ *true*
(T6.5) *Cover* (*Division*-A, R8, *true*) ⇒ *true*
(T6.6) *Cover* (*Division*-A, R9, *true*) ⇒ *true*
(T6.7) *Cover* (*Division*-A, R10, *true*) ⇒ *true*
(T6.8) *Cover* (*Division*-F, R8, *true*) ⇒ *false*
(T6.9) *Cover* (*Division*-F, R9, *true*) ⇒ *false*
(T6.10) *Cover* (*Division*-F, R10, *true*) ⇒ *false*
(T6.11) *NotCover* (*Division*-A, R8, *true*) ⇒ *false*
(T6.12) *NotCover (Division*-A, R9, *true*) ⇒ *false*
(T6.13) *NotCover (Division*-A, R10, *true*) ⇒ *false*
(T6.14) *NotCover (Division*-F, R8, *true*) ⇒ *true*
(T6.15) *NotCover (Division*-F, R9, *true*) ⇒ *true*
(T6.16) *NotCover (Division*-F, R10, *true*) ⇒ *true*
(T6.17) *Equal* ({R4, R10}) ⇒ *true*
(T6.18) *Equal* ({R4, R11}) ⇒ *false*
(T6.19) *NotEqual* ({R4, R10}) ⇒ *false*
(T6.20) *NotEqual* ({R4, R11}) ⇒ *true*
(T6.21) *Overlap* (R6, R13) ⇒ ?
(T6.22) *NotOverlap* (R5, R12) ⇒ *false*
(T6.23) *NotOverlap* (R5, R14) ⇒ ?
(T6.24) *Span* (R7, R13) ⇒ ?
(T6.25) *NotSpan* (R7, R13) ⇒ ?

T7: Transações de busca topológica

fase de preparação:

Select (estradas.chave = 155.000, estradas) ⇒ R1 [*Division*-B]
Select (vegetação.chave = 35, vegetação) ⇒ R2 [*Division*-F]
Select (solos.chave = 262.144, solos) ⇒ R3 [*Division*-A]
Select (solos.chave = 1.310.720, solos) ⇒ R4 [*Division*-F]

Select (solos.chave = 1.835.008, solos) ⇒ R5 [*Division-G*]
Select (cidades.chave = 360.000, cidades) ⇒ R6 [*Division-C*]
Select (estradas.chave = 685.000, estradas) ⇒ R7 [*Division-D*]
Select (vegetação.chave = 1, vegetação) ⇒ R8 [*Division-A*]
Select (vegetação.chave = 37, vegetação) ⇒ R9 [*Division-G*]
Select (solos.chave = 851.968, solos) ⇒ R10 [*Division-D*]
Inside (*Division-D*, cidades, *true*) ⇒ R11

Esquema de medição:

(T7.1) *Nearest* (R1, cidades)
(T7.2) *Nearest* (R1, estradas)
(T7.3) para cada $c \in$ R11 faça *Nearest* (c, cidades)
(T7.4) para cada $c \in$ R11 faça *Nearest* (c, estradas)
(T7.5) *Adjacent* (R2, vegetação)
(T7.6) *Adjacent* (R3, solos)
(T7.7) *Adjacent* (R4, solos)
(T7.8) *Adjacent* (R5, solos)
(T7.9) *NotAdjacent* (R2, vegetação)
(T7.10) *NotAdjacent* (R3, solos)
(T7.11) *Inside* (*Divisions-A/B/C/D/E*, cidades, *true*) ⇒ seletividade de 91%
(T7.12) *Inside* (*Division-C*, cidades, *true*) ⇒ seletividade de 40%
(T7.13) *Inside* (*Division-B*, estradas, *true*) ⇒ seletividade de 1%
(T7.14) *Inside* (*Division-F*, solos, *true*) ⇒ seletividade de 25%
(T7.15) *NotInside* (*Divisions-A/B/C/D/E*, cidades, *true*) ⇒ seletividade de 9%
(T7.16) *NotInside* (*Division-C*, cidades, *true*) ⇒ seletividade de 60%
(T7.17) *NotInside* (*Division-B*, estradas, *true*) ⇒ seletividade de 99%
(T7.18) *NotInside* (*Division-G*, solos, *true*) ⇒ seletividade de 75%
(T7.19) *Embody* (R4, {vegetação, solos})
(T7.20) *Embody* (R1, {vegetação, solos})
(T7.21) *Embody* (R6, {vegetação, solos})
(T7.22) *NotEmbody* (R4, {vegetação, solos})
(T7.23) *NotEmbody* (R1, {vegetação, solos})
(T7.24) *NotEmbody* (R6, {vegetação, solos})
(T7.25) *Intersect* (R7, estradas)
(T7.26) *Intersect* (R8, estradas)
(T7.27) *Intersect* (R9, estradas)
(T7.28) *Intersect* (R2, estradas)
(T7.29) *Intersect* (R7, solos)
(T7.30) *Intersect* (R8, solos)
(T7.31) *Intersect* (R9, solos)
(T7.32) *Intersect* (R2, solos)
(T7.33) *NotIntersect* (R7, estradas)

(T7.34) *NotIntersect* (R8, estradas)
(T7.35) *NotIntersect* (R7, solos)
(T7.36) *NotIntersect* (R8, solos)
(T7.37) *Cross* (R10, estradas)
(T7.38) *NotCross* (R10, estradas)

T8: Transações topológicas escalares

fase de preparação:

Select (estradas.chave = 685.000, estradas) ⇒ R1 [*Division*-D]
Select (solos.chave = 851.968, solos) ⇒ R2 [*Division*-D]
Inside (*Division*-D, solos, *true*) ⇒ R3
Inside (*Division*-D, estradas, *true*) ⇒ R4

Esquema de medição:

(T8.1) *WhereIntersect* (R1, R3)
(T8.2) *WhereIntersect* (R2, R4)

T9: Transações escalares

fase de preparação:

Select (cidades.chave = 360.000, cidades) ⇒ R1 [*Division*-C]
Select (cidades.chave = 960.000, cidades) ⇒ R2 [*Division*-G]
Select (estradas.chave = 685.000, estradas) ⇒ R3 [*Division*-D]
Select (estradas.chave = 155.000, estradas) ⇒ R4 [*Division*-B]

Esquema de medição:

(T9.1) *MinDistance* (R1, R2)
(T9.2) *MinDistance* (R1, R3)
(T9.3) *MinDistance* (R3, R4)

T10: Uso de atributo derivado x operação analítica

fase de preparação:

Select (estradas.chave ≥ 1 *and* estradas.chave ≤ 1.000.000, estradas) ⇒ R1 [*Extent*]
Select (vegetação.chave≥ 1 *and* vegetação.chave ≤ 38, vegetação) ⇒ R2 [*Extent*]
Select (solos.chave≥ 1 *and* solos.chave ≤ 2.097.152, solos) ⇒ R3 [*Extent*]

Esquema de medição:

(T10.1) para cada x ∈ R1 faça *Length* (x)
(T10.2) para cada x ∈ R2 faça *Area* (x)
(T10.3) para cada x ∈ R3 faça *Perimeter* (x)
(T10.4) para cada x ∈ R1 faça estradas.comprimento
(T10.5) para cada x ∈ R2 faça vegetação.área
(T10.6) para cada x ∈ R3 faça solos.perímetro

T11: Transações baseadas em conjunto

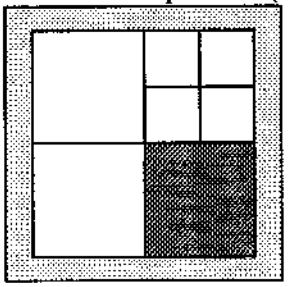
fase de preparação:

Select (solos.chave = 1, solos) ⇒ R1 [*Division*-A]
Select (vegetação.chave = 1, vegetação) ⇒ R2 [*Division*-A]
Inside (*Division*-F, solos, *true*) ⇒ R3

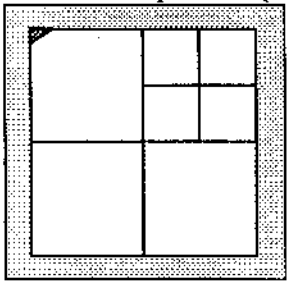
Esquema de medição:

(T11.1) *Union* (R3)
(T11.2) *Intersection* ({R1, R2})
(T11.3) *Difference* (R1, R2) ⇒ ∅
(T11.4) *Difference* (R2, R1)

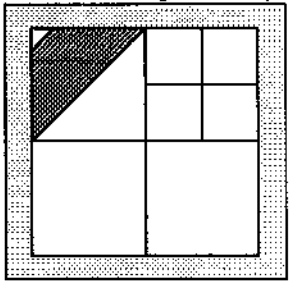
Resultado esperado: (t11.1)



Resultado esperado: (t11.2)



Resultado esperado: (t11.4)



T12: Transações diversas

fase de preparação:

Inside (*Division*-B, cidades, *true*) ⇒ R1

Esquema de medição:

(T12.1) x := ∅
para cada p ∈ R1
faça para cada q ∈ {R1-p}
faça *Segment* (p, q) ⇒ z
x := x + z
Save (x)

(T12.2) *LoadSystem*
(T12.3) *Select* (cidades.data.mês = 12, cidades) ⇒ 0%
(T12.4) *Select* (cidades.data.ano = 1985, cidades) ⇒ 5%
(T12.5) *Select* (cidades.nome[1] = "D", cidades) ⇒ 12%
(T12.6) *Select* (cidades.população > 10.000 *and* cidades.população ≤ 100.000, cidades) ⇒ 20,4%
(T12.7) *Select* (cidades.hospitais = 0, cidades) ⇒ 34%
(T12.8) *Select* (cidades.altitude ≤ 100, cidades) ⇒ 43,8%
(T12.9) *Select* (cidades.data.dia ≥ 1 *and* cidades.data.dia ≤ 15, cidades) ⇒ 60%
(T12.10) *Select* (cidades.nome[1] = "A" *or* cidades.nome[1] = "B" *or*
cidades.nome[1] = "C", cidades) ⇒ 88%
(T12.11) *Select* (cidades.data.mês ≥ 1 *and* cidades.data.mês ≤ 10, cidades) ⇒ 100%

T13: Seleção convencional x seleção espacial

Seleção de objetos geográficos que satisfaçam ambos predicados convencional e espacial.

Pc = predicado convencional
Pe = predicado espacial

(T13.1) Pc | Pe (R:= Pc ∩ Pe ⇒ Resultado final)
Pc: *Select* (cidades.nome[1] = "A" *or* cidades.nome[1] = "B" *or*
cidades.nome[1] = "C", cidades) ⇒ 88%
Pe: *Inside* (*Division*-B, cidades, *true*) ⇒ 1%

(T13.2) Pc | Pe (R:= Pc ∩ Pe ⇒ Resultado final)
Pc: *Select* (cidades.data.ano = 1985, cidades) ⇒ 5%
Pe: *Inside* (*Divisions*-A/B/C/D/E, cidades, *true*) ⇒ 91%

(T13.3) Pc | Pe (R:= Pc ∩ Pe ⇒ Resultado final)
Pc: *Select* (cidades.data.ano = 1985 *or* cidades.data.ano = 1986, cidades) ⇒ 10%
Pe: *Inside* (*Division*-E, cidades, *true*) ⇒ 10%

(T13.4) Pc | Pe (R:= Pc ∩ Pe ⇒ Resultado final)
Pc: *Select* (cidades.data.mês ≥ 1 *and* cidades.data.mês ≤ 9, cidades) ⇒ 90%
Pe: *Inside* (*Divisions*-A/C/D/E, cidades, *true*) ⇒ 90%

Capítulo 7

Conclusões

Esta dissertação teve por objetivo propor a carga de trabalho e caracterizar os dados de um *benchmark* voltado à análise de desempenho de SIGs. Visando à criação de um *benchmark* representativo, foi elaborado um questionário que serviu de base para a determinação das transações e do conjunto de dados utilizado por aplicações georeferenciadas reais. Este questionário foi de propósito geral e englobou aspectos relacionados ao *hardware*, *software* e *liveware*, servindo não somente para o levantamento das características espaciais de aplicações georeferenciadas, mas também para caracterizar o segmento de geoprocessamento nas principais instituições brasileiras.

A carga de trabalho do *benchmark* é composta por um conjunto de transações primitivas, especificadas em alto nível, que podem ser utilizadas para a formação de transações mais complexas. Estas transações primitivas propostas são predominantemente orientadas aos dados espaciais, sendo, a priori, independentes do formato de dados utilizado (*raster* ou vetorial). No entanto, algumas transações definidas são voltadas especificamente para o formato *raster*, ou ainda utilizam ambos formatos conjuntamente. A carga de trabalho proposta é baseada em transações realizadas durante a fase de execução do sistema. Em outras palavras, não são consideradas, por exemplo, operações realizadas durante a fase de captura de dados.

A caracterização dos dados do *benchmark* foi efetuada em termos dos tipos de dados necessários para a representação de aplicações georeferenciadas, e adicionalmente procedimentos para se realizar a geração de dados sintéticos. Esta caracterização é independente da forma de organização dos dados (estrutura de arquivos, quantidade de dados armazenados, entre outros), de modo a permitir a execução da carga de trabalho do *benchmark* segundo aplicações georeferenciadas específicas.

Finalmente, para permitir a comparação de desempenho entre SIGs distintos, foi definida uma aplicação alvo na qual a carga de trabalho do *benchmark* proposto pode ser executada. Esta aplicação é composta por dados sintéticos, sendo voltada para representar uma aplicação georeferenciada de grande porte. Assim, a comparação de desempenho entre SIGs distintos é possível, uma vez que foram definidos com rigor tanto aspectos

relacionados às transações ("*bufferização*", seletividade e arquivos sobre os quais cada transação irá atuar) quanto aspectos relacionados aos dados (estrutura do banco de dados, tipos de dados, quantidade, controle de seletividade e distribuição).

Desta forma, as principais contribuições decorrentes da realização deste trabalho são:

- determinação de um conjunto expressivo de transações primitivas que permite a derivação de outras transações complexas presentes em aplicações georeferenciadas.

- formalização das transações, facilitando o seu entendimento e a sua portabilidade para linguagens de consultas de SIGs comerciais.

- tratamento transparente e uniforme de transações que envolvam os formatos *raster* e vetorial.

- caracterização da geração de dados sintéticos, especialmente para dados espaciais.

Como primeira extensão a este trabalho está prevista a execução da carga de trabalho do *benchmark* segundo a aplicação definida no capítulo 6 para diferentes SIGs. Desta forma, poderá ser testado o desempenho destes SIGs segundo distintas configurações de *hardware* / sistema operacional / SGBD / SIG. Uma segunda extensão é a utilização das transações primitivas para teste de representatividade de linguagens de consulta geográficas. Outras extensões incluem o acoplamento de transações temporais à carga de trabalho do *benchmark*, a execução da carga de trabalho do *benchmark* segundo aplicações georeferenciadas específicas, como exemplo no projeto SAGRE – Telebrás, e a integração do modelo proposto por [Per90], para a geração de dados convencionais, ao modelo proposto por este trabalho, para a geração de dados espaciais.

Bibliografia

[AA94] ALENCAR, V.C.A. AQUINO, M.S. Um Sistema de Informações Geográficas para o Planejamento de Rotas de Ônibus. In: GIS Brasil 94 – Congresso e Feira para Usuários de Geoprocessamento, Curitiba, PR, Brasil. *Anais*, 1994. Usos não Convencional de SIGs, p.1-10.

[Agu95] AGUIAR, C.D. *Integração de Sistemas de Banco de Dados Heterogêneos em Aplicações de Planejamento Urbano.* Campinas: DCC – Unicamp, 1995. (Tese de Mestrado).

[Alv89] ALVES, D.S. Modelo de Dados para Sistemas de Informação Geográfica. São Paulo: Escola Politécnica – USP, 1989. (Tese de Doutorado).

[And+90] ANDERSON, T.L., *et al.* The HiperModel Benchmark. In: *Lecture Notes in Computer Science*, v.416, p.317-331. Spring Verlag, 1990.

[Ano+85] ANON, *et al.* A Measure of Transaction Processing Power. *Datamation*, v.31, n.7, p.112-118, April 1985.

[Arg+93] ARGONDIZIO, E.L., *et al.* Conversão de Mapeamento Urbano – Uma Metodologia para o Projeto SAGRE. In: IV Conferência Latino Americana sobre Sistemas de Informação Geográfica e II Simpósio Brasileiro de Geoprocessamento, São Paulo, SP, Brasil. *Anais*, 1993. p.207-219.

[Aro89] ARONOFF, S. *Geographic Information Systems: A Management Perspective.* WDL Publications, Ottawa, Canada, 1989.

[AS86] ABEL, D.J., SMITH, J.L. A Relational GIS Database Accommodating Independent Partionings of the Region. In: 2nd International Symposium on Spatial Data Handling, Seattle, Washington. *Proceedings*, 1986.

[AS94] ASKOV, D.C., STUTZ F.P. Freeway Alignment and Community Residents' Receptivity: A GIS Distance Buffering Application. In: GIS/LIS 94, Phoenix, Arizona, USA. *Proceedings*, 1994. p.14-23.

[AX94] ALEXANDER, M. XIANG, W. Crime Pattern Analysis Using GIS. In: GIS/LIS 94, Phoenix, Arizona, USA. *Proceedings*, 1994. p.1-3.

[Bat+93] BATISTELLA, M., *et al.* Banco de Dados Geocodificados com Base Municipal para a Região Nordeste do Brasil. In: IV Conferência Latinoamericana sobre Sistemas de Informação Geográfica e II Simpósio Brasileiro de Geoprocessamento, São Paulo, SP, Brasil. *Anais*, 1993. p.89-101.

[Bat+94] BATISTELLA, M., *et al.* As Atividades de Geoprocessamento no Núcleo de Monitoramento Ambiental da Embrapa. In: GIS Brasil 94 – Congresso e Feira para Usuários de Geoprocessamento, Curitiba, PR, Brasil. *Anais*, 1994. Meio Ambiente e Recursos Naturais, p.58-67.

[BB81] BARRERA, R., BUCHMANN, A. Schema Definition and Query Language for a Geographical Data Base System. *Computer Architecture for Pattern Analysis and Image Database Management*, p.250-256, 1981.

[BD84] BORAL, H., DEWITT, D.J. A Methodology for Database System Performance Evaluation. In: SIGMOD Conference, Boston, MA, USA. *Proceedings*, 1984.

[Ber87] BERRY, J.K. Fundamental Operations in Computer-Assisted Map Analysis. *International Journal of Geographical Information Systems*, v.1, n.2, p.119-136. 1987.

[Ber+91] BERKUM, J.J.A., *et al.* Aspects of Computer Simulations in an Instructional Context. *Education and Computing*, v.6, p.231-239, 1991.

[Bou88] BOURSIER, P. Analysis of Urban Geographic Queries. In: N.M. Thalmann and D. Thalmann, editors, *New Trends in Computer Graphics*, p.601-610. Spring-Verlag, 1988.

[BS77] BERMAN, R., STONEBRAKER, M. Geo-QUEL: A System for the Manipulation and Display of Geographic Data. *Computer Graphics*, v.11, n.2, p.186-191, 1977.

[BT88] BITTON, D., TURBYFILL, C. A Retrospective on the Wisconsin Benchmark. In: M. Stonebraker, editor, *Readings in Database Systems*, p.280-289. Morgan Kaufmann Publishers, Inc., San Mateo, CA, USA, 1988.

[Car86] CARSTENSEN, L.W. Relational Land Information Systems Development Using Relational Databases and Geographic Information Systems. In: Auto Carto, London, England. *Proceedings*, 1986. p.507-516.

[Car89] CARTER, J.K. On Defining the Geographic Information System. In: W.J. Ripple, editor, *Fundamentals of Geographic Information Systems: A Compendium*, p.3-7. The American Society for Photogrammetry and Remote Sensing and the American Congress on Surveying and Mapping, USA, 1989.

[CB92] CIFERRI, R.R., BEZERRA, W.S. *Informativo de Marketing: TPC Benchmarks.* Itautec Informática, 1992. (Technical Report).

[CDN93] CAREY, M.J., DEWITT, D.J., NAUGHTON, J.F. The OO7 Benchmark. *SIGMOD RECORD*, v.22, n.2, p.12-21, 1993.

[Cod81] CODD, E.F. Data Models in Database Management. In: Workshop on Data Abstraction Databases and Conceptual Modeling. *Proceedings*, 1981.

[Cor94] CORNACCHIONI, L.A.B. Utilização do GIS em Recursos Naturais na Cia Suzano de Papel e Celulose. In: GIS Brasil 94 – Congresso e Feira para Usuários de Geoprocessamento, Curitiba, PR, Brasil. *Anais*, 1994. Florestal e Agrícola, p.1-5.

[Cox91] COX JÚNIOR, F.S. *Análise de Métodos de Acesso a Dados Espaciais Aplicados a Sistemas Gerenciadores de Banco de Dados.* Campinas: DCC – Unicamp, 1991. (Tese de Mestrado).

[Cow90] COWEN, D.J. GIS versus CAD versus DBMS: What are the Differences ?. In: D.J. Peuquet and D.F. Marble, editors, *Introductory Readings in Geographic Information Systems*, p.52-61. Taylor & Francis Ltd., London, England, 1990.

[CS92] CATTELL, R.G.G., SKEEN, J. Object Operations Benchmark. *ACM Transactions on Database Systems*, v.17, n.1, p.1-31, March 1992.

[CSV94] CAVALCANTI, A.E.C., SILVA, I.C., VIEIRA FILHO, M.M. Experiência da Celpe no Processo de Automação de Geoprocessamento para o Controle da Distribuição de Energia no Estado de Pernambuco. In: GIS Brasil 94 – Congresso e Feira para Usuários de Geoprocessamento, Curitiba, PR, Brasil. *Anais*, 1994. Concessionárias, p.18-26.

[Dei90] DEITEL, H.M. *Operating Systems*. Addison-Wesley Publishing Company, USA, 1990. 229p.

[Dew85] DEWITT, D.J. Benchmarking Databases Systems: Past Effort and Future Directions. *Database Engineering*, v.8, n.1, p.2-9, March 1985.

[DF94] DAVIS JÚNIOR, C.A., FONSECA, F.T. Geoprocessamento em Belo Horizonte: Aplicações. In: GIS Brasil 94 – Congresso e Feira para Usuários de Geoprocessamento, Curitiba, PR, Brasil. *Anais*, 1994. Municipal, p.1-10.

[DC88] DICKINSON, H., CALKINS, H.W. The Economic Evaluation of Implementing a GIS. *International Journal of Geographical Information Systems*, v.2, n.4, p.307-327. 1988.

[Dym94] DYMON, U.J. Mapping Severe Weather Alert Systems in Alabama and Georgia. In: GIS/LIS 94, Phoenix, Arizona, USA. *Proceedings*, 1994. p.229-233.

[Ege94] EGENHOFER, M.J. Spatial SQL: A Query and Presentation Language. *IEEE Transactions on Knowledge and Data Engineering*, v.6, n.1, p.86-95, February 1994.

[EH84] EFFELSBERG, W., HAERDER, T. Principles of Database Buffer Management. *ACM Toods*, v.9, n.4, December 1984.

[Ele94] ELETRICIDADE DE SÃO PAULO S/A. *Especificação funcional do sistema AUTOCART (projeto SIGRADE), 1994*. (Technical Report).

[EN89] ELMASRI, R., NAVATHE, S.B. Fundamentals of Database Systems. The Benjamin/Cummings Publishing Company, Inc., 1989. 802p.

[Fer78] FERRARI, D. *Computer Systems Performance Evaluation*. Prentice Hall, New Jersey, USA, 1978.

[Fer86] FERREIRA, A.B.H. *Novo Dicionário da Língua Portuguesa*. Editora Nova Fronteira, Brasil, 1986. 1838p.

[Fra82] FRANK, A. MapQuery: Data Base Query Language for Retrieval of Geometric Data and their Graphical Representation. *Computer Graphics*, v.16, n.3, p.199-270, 1982.

[Fra91] FRANKLIN, W.M.R. Computer Systems and Low-Level Data Structures for GIS. In: D.J. Maguire, M.F. Goodchild and D.W. Rhind, editors, *Geographical Information .Systems, Principles and Applications*, p.215-225, v.1. Longman Group UK Limited, England, 1991.

[GC94] GOMEL, D., CAMPOS, R.F. Da Praça ao Bureau – Definição dos Caminhos na Rede de Elementos Histórico-Culturais com Auxílio do SIG. In: GIS Brasil 94 – Congresso e Feira para Usuários de Geoprocessamento, Curitiba, PR, Brasil. *Anais*, 1994. Usos não Convencional de SIGs, p.25-34.

[GR87] GOODCHILD, M.F., RIZZO, B.R. Performance Evaluation and Work-load Estimation for Geographic Information Systems. *International Journal of Geographical Information Systems*, v.1, n.1, p.67-76. 1987.

[Gra91] GRAY, J. *The Benchmark Handbook for Database and Transaction Processing Systems*. Morgan Kaufmann Publishers, Inc., San Mateo, CA, USA, 1991. 334p.

[GRY95] GOLSTEIN, J., RAMAKRISHNAN, R., YU, J. Using Constraints to Query R*-Trees. Department of Computer Sciences, University of Wisconsin, Madison, 1995. 27p. (Technical Report).

[Gui80] GUIMARÃES, C.C. *Princípios de Sistemas Operacionais*. Editora Campus, Rio de Janeiro, Brasil, 1980. 222p.

[Güt94] GÜTING, R.H. An Introduction to Spatial Database Systems. *The VLDB Journal*, v.3, n.4, 1994. p.357-399.

[KHS91] KRIEGEL, H.P., HORNM H., SCHIWIETZ, M. The Performance of Object Decomposition Techniques for Spatial Query Processing. In: *Lecture Notes in Computer Science*, v.525, p.257-276. Spring Verlag, 1991.

[KSR91] KOHLER, W., SHAH, A., RAAB, F. *Overview of TPC Benchmark C: The Order-Entry Benchmark*. Transaction Processing Performance Council, December, 1991. (Technical Report).

[KS94] KORTH, H.F., SILBERSCHATZ, A. *Sistemas de Banco de Dados*. Makron Books do Brasil Editora Ltda, Brasil, 1994. 748p.

[Jey88] JEYANANDAN, D. A Systems Approach to Land Information Systems. *Surveying and Mapping*, v.48, n.3, p.161-171, September 1988.

[Lu+91] LU, H. *et al.* Storage Management in Geographical Information Systems. In: *Lecture Notes in Computer Science*, v.525, p.451-470. Spring Verlag, 1991.

[Mag81] MAGALHÃES, G.C. *Improving the Performance of Database Systems*. Toronto: Department of Computer Sciences – University of Toronto, 1981. (Tese de Doutorado).

[Mag91] MAGUIRE, D.J. An Overview and Definition of GIS. In: D.J. Maguire, M.F. Goodchild and D.W. Rhind, editors, *Geographical Information Systems, Principles and Applications*, p.9-20, v.1. Longman Group UK Limited, England, 1991.

[Mag93] MAGALHÃES, G.C. Projeto SAGRE – Sistema Automatizado de Gerência de Rede Externa – Telebrás – Telecomunicações Brasileiras S/A. *Fator GIS*, n.3, p.26-28, 1993.

[Mag94] MAGNABOSCO, S.M. Roteamento Turístico com Uso de SIG no Paraná. In: GIS Brasil 94 – Congresso e Feira para Usuários de Geoprocessamento, Curitiba, PR, Brasil. *Anais*, 1994. Usos não Convencional de SIGs, p.21-24.

[Mag+94] MAGALHÃES, G.C, *et al.* Especificação Técnica de Conversão de Dados – Proposta da Telebrás – Projeto SAGRE. In: GIS Brasil 94 – Congresso e Feira para Usuários de Geoprocessamento, Curitiba, PR, Brasil. *Anais*, 1994. Concessionárias, p.43-52.

[Mar90] MARBLE, D.F. Geographic Information Systems: An Overview. In D.J. Peuquet and D.F. Marble, editors, *Introductory Readings in Geographic Information Systems*, p.8-17. Taylor & Francis Ltd., 1990.

[MCD94] MEDIANO, M.R., CASANOVA, M.A., DREUX, M. V-Trees – A Storage Method for Long Vector Data. In: 20th VLDB Conference, Santiago, Chile. *Proceedings*, 1994.

[Mcl90] MCLAREN, R.A. *The Art of Benchmarking GIS Technology*. Know Edge Ltd., Edinburgh, 1990. (Technical Report).

[Med95] MEDIANO, M.R. *V-Trees: Um método de Armazenamento para Dados Vetoriais Longos*. Rio de Janeiro: DI – Pontificia Universidade Católica do Rio de Janeiro, 1995. (Tese de Mestrado).

[MM93] MEDEIROS, C.M.B., MAGALHÃES, G.C. *Rule Application in GIS – a Case Study*. DCC – Unicamp, 1993. 19p. (Technical Report).

[Mor+94] MORELLI, R., *et al.* Sistema de Informações Geográficas para Previsão de Safras. In: GIS Brasil 94 – Congresso e Feira para Usuários de Geoprocessamento, Curitiba, PR, Brasil. *Anais*, 1994. Florestal e Agrícola, p.33-38.

[MP94] MEDEIROS, C.M.B, PIRES, F. Databases for GIS. *ACM Sigmod Record*, v.23, n.1, p.107-115, 1994.

[Mul85] MULLER, J.C. Geographic Information Systems: A Unifying Force for Geography. *The Operational Geographer*, n.8, p.41-43, 1985.

[Oli93] OLIVEIRA, R.L. *Transparência de Modelos em Sistemas de Bancos de Dados Heterogêneos*. Campinas: DCC – Unicamp, 1993. (Tese de Mestrado).

[Ooi90] OOI, B.C. Efficient Query Processing in Geographic Information Systems. In: *Lecture Notes in Computer Science*, v.471. p.1-209. Spring Verlag, 1990.

[Ope91] OPENSHAW, S. Developing Appropriate Spatial Analysis Methods for GIS. In: D.J. Maguire, M.F. Goodchild and D.W. Rhind, editors, *Geographical Information Systems, Principles and Applications*, p.389-402, v.1. Longman Group UK Limited, England, 1991.

[Per90] PEREIRA, R.F. *Análise de Desempenho de Banco de Dados utilizando Benchmarks Especializados*. Campinas: DCC – Unicamp, 1990. (Tese de Mestrado).

[Peu90] PEUQUET, D.J. A Conceptual Framework and Comparison of Spatial Data Models. In: D.J. Peuquet and D.F. Marble, editors, *Introductory Readings in Geographic Information Systems*, p.251-285. Taylor & Francis Ltd., London, England, 1990.

[Pir95] PIRES, F. Um Ambiente de Modelagem de Dados Geográficos. To be published, 1995.

[PM90] PEUQUET, D.J., MARBLE, D.F. *Introductory Readings in Geographic Information Systems*. Taylor & Francis Ltd., London, England, 1990. 371p.

[PMS93] PIRES, F., MEDEIROS, C.M.B., SILVA, A.B. Modelling Geographic Information Systems Using an Object Oriented Framework. In: 13th International Conference on the Chilean Computer Science Society, La Serena, Chile. *Proceedings*, 1993. p.217-232.

[RC94] ROSSETO, C.F. CUNHA, C.B. A Aplicação do Geoprocessamento na Roteirização de Veículos. In: GIS Brasil 94 – Congresso e Feira para Usuários de Geoprocessamento, Curitiba, PR, Brasil. *Anais*, 1994. Usos não Convencional de SIGs, p.35-44.

[Rhi88] RHIND, D.W. A GIS Research Agenda. *International Journal of Geographical Information Systems*, v.2, n.1, p.23-28. 1988.

[Rip89] RIPPLE, W.J. *Fundamentals of Geographic Information Systems: A Compendium*. The American Society for Photogrammetry and Remote Sensing and the American Congress on Surveying and Mapping, USA, 1989. 248p.

[RKC87] RUBENSTEIN, W.B., KUBICAR, M.S., CATTELL, R.G.G. Benchmarking Simple Database Operations. *SIGMOD RECORD*, v.16, n.3, p.387-394, May 1987.

[Roc94] ROCHA, H.O. Aplicações do Geoprocessamento na Avaliação da Aptidão Agrícola das Terras. In: GIS Brasil 94 – Congresso e Feira para Usuários de Geoprocessamento, Curitiba, PR, Brasil. *Anais*, 1994. Florestal e Agrícola, p.39-47.

[Rod+94] RODRIGUES, M., *et al.* SIG na Gestão de Patrimônio Imobiliário. In: GIS Brasil 94 – Congresso e Feira para Usuários de Geoprocessamento, Curitiba, PR, Brasil. *Anais*, 1994. Painéis, p.46-52.

[Ros95] ROSA, F.S. Softwares de Geoprocessamento: Quem é quem. *Fator GIS*, n.8, p.21-25, 1995.

[RY94] REVELL, N., YOUSSEF, W. *Database Performance Evaluation – A Methodological Approach*. School of Informatics, Department of Business Computing, The City University, 1994. 15p. (Technical Report).

[SC94] SOUZA, S.P., CALIJURI, M.C. Estudos Ambientais para Minimização do Assoreamento da Bacia Hidrográfica do Rio Manso (Minas Gerais) Utilizando-se os Sistemas de Informações Geográficas. In: GIS Brasil 94 – Congresso e Feira

para Usuários de Geoprocessamento, Curitiba, PR, Brasil. *Anais*, 1994. Painéis, p.53-57.

[SD92] STONEBRAKER, M, DOZIER, J. *Sequoia 2000: Large Capacity Servers to Support Global Change Research*. Electronics Research, March 1992. (Technical Report).

[SE90] STAR, J., ESTES, J. *Geographic Information Systems – An Introduction*. Prentice Hall, Inc., New Jersey, USA, 1990. 303p.

[Sie94] SIEBERT, U. SIG na Sanepar: Os Primeiros Passos. *Fator GIS*, n.7, p.29-31, 1994.

[Sil94] SILVA, A.B. *Sistemas Geo-Referenciados de Informação: Uma Introdução*. Instituto de Geociências da Unicamp, Outubro, 1994. (Technical Report).

[Smi+89] SMITH, T.R., *et al.* Requirements and Principles for the Implementation and Construction of Large-Scale Geographic Information Systems. In: W.J. Ripple, editor, *Fundamentals of Geographic Information Systems: A Compendium*, p.19-37. The American Society for Photogrammetry and Remote Sensing and the American Congress on Surveying and Mapping, USA, 1989.

[Sou+93] SOUZA, J.M., *et al.* Uma Arquitetura Organizacional para Sistemas de Informação Geográfica Orientados a Objetos. In: IV Conferência Latino-americana sobre Sistemas de Informação Geográfica e II Simpósio Brasileiro de Geoprocessamento, São Paulo, SP, Brasil. *Anais*, 1993. p.187-204.

[SP94] SILVA, J.C.C., PENHA, U.C. O Geoprocessamento na COMIG: Experiências na Área de Geologia e Recursos Minerais. In: GIS Brasil 94 – Congresso e Feira para Usuários de Geoprocessamento, Curitiba, PR, Brasil. *Anais*, 1994. Meio Ambiente e Recursos Naturais, p.51-57.

[SR87] STONEBRAKER, M. ROWE, L. The Postgress Data Model. In: 13th VLDB Conference, Brighton, England. *Proceedings*, 1987. p.83-96.

[Sto+93] STONEBRAKER, M., *et al.* The Sequoia 2000 Storage Benchmark. In: ACM SIGMOD Conference, Washington, DC, USA. *Proceedings*, 1993. p.2-11.

[Sto94] STONEBRAKER, M. *The Sequoia 2000 Project*. EECS Dept., University of California at Berkeley, Berkeley, 1994. 5p. (Technical Report).

[Tan92a] TANENBAUM, A.S. *Organização Estruturada de Computadores*. Prentice-Hall do Brasil, Rio de Janeiro, Brasil, 1992. 460p.

[Tan92b] TANENBAUM, A.S. *Modern Operating Systems*. Prentice-Hall Inc., New Jersey, USA, 1992. 730p.

[Tel91] TELECOMUNICAÇÕES BRASILEIRAS S/A. *Edital de Concorrência – Obtenção da Licença de Uso de Programas de Computador que Realizem o Gerenciamento de Informações Geográficas Integrados a Sistemas de Banco de Dados Comerciais*, Agosto 1991. (Technical Report).

[Tim94] TIMES, V.C. *MGeo: Um Modelo Orientado a Objetos para Aplicações Geográficas*. Recife: DI – UFPe, 1994. (Tese de Mestrado).

[TOB91] TURBYFILL, C., ORJI, C., BITTON, D. AS^{3}AP: An ANSI SQL Standard Scaleable and Portable Benchmark for Relational Database Systems. In: J. Gray editor, *The Benchmark Handbook for Database and Transaction Processing Systems*, p.167-207. Morgan Kaufmann Publishers, Inc., San Mateo, CA, USA, 1991.

[Tom91] TOMLIN, C.D. Cartographic Modelling. In: D.J. Maguire, M.F. Goodchild and D.W. Rhind, editors, *Geographical Information Systems, Principles and Applications*, p.361-374, v.1. Longman Group UK Limited, England, 1991.

[Tra89a] TRANSACTION PROCESSING PERFORMANCE COUNCIL. *TPC benchmark A – Draft 6-DR Proposed Standard*, August 1989. (Technical Report).

[Tra89b] TRANSACTION PROCESSING PERFORMANCE COUNCIL. *TPC Benchmark A Standard*, November 1989. (Technical Report).

[Tra92] TRANSACTION PROCESSING PERFORMANCE COUNCIL. *TPC Benchmark C, Standard Specification, Revision 1.0*, Edited by François Raab, August 1992. (Technical Report).

[Tur87] TURBYFILL, C. *Comparative Benchmarking of Relational Database Systems*. Ithaca: Department of Computer Sciences – Cornell University, NY, USA, 1987. (Tese de Doutorado).

[US90] UNLAND, R., SCHLAGETER, G. Object-Oriented Database Systems: Concepts and Perspectives. In: *Lecture Notes in Computer Science*, v.466, p.154-197. Spring Verlag, 1990.

[Web94a] WEBER, M.A.A. Mais Energia para São Paulo. Fator GIS, n.5, p.47, 1994.

[Web94b] WEBER, M.A.A. Geoprocessamento em Redes de Distribuição – Experiência Eletropaulo. In: GIS Brasil 94 – Congresso e Feira para Usuários de Geoprocessamento, Curitiba, PR, Brasil. *Anais*, 1994. Concessionárias, p.27-34.

[Wor92] WORBOYS, M.F. A Generic Model for Planar Geographical Objects. *International Journal of Geographical Information Systems*, v.6, n.5, p.353-372. 1992.

Apêndice A

Resumo das Transações Primitivas

Transações primitivas genéricas

Reclassificação

Reclass (O: Set (OG), NewCategories: TypeNC) : Set (OG)

Superposição

Overlay ([Not] O1, [Not] O2: Set (OG-2D), Operation: TypeOp) : Set (OG-2D)

Análise de Ponderação

Weigh (O1, O2: Set (OG-2D), Operation: TypeOp, Weight: TypeW) : Set (OG-2D)

Análise de proximidade simples

SimpleBuffer (O: OG, Distance: Float, Additional: TypeAd) : OG-2D

Análise de proximidade simples ao redor de múltiplos OGs

SimpleBufferMultiple (O: Set (OG), Distance: Float) : Set (OG-2D)

Análise de proximidade múltipla

MultipleBuffer (O: OG, Distance: Float, Levels: Integer, Additional: TypeAd) : Set (OG-2D)

Análise de proximidade múltipla ao redor de múltiplos OGs

MultipleBufferMultiple (O: Set(OG), Distance: Float, Levels: Integer) : Set (OG-2D)

Interior de um OG-2D

GetInterior (O: OG-2D) : OG-2D

Fronteira de um OG-2D

GetBoundary (O: OG-2D) : OG-1D

Transações primitivas topológicas *booleanas*

Adjacência

Meet (O1: OG, O2: Set (OG)) : Boolean

Não Adjacência

NotMeet (O1: OG, O2: Set (OG)) : Boolean

Inclusão

Cover (O1: OG, O2: Set (OG), Boundary: Boolean) : Boolean

Não Inclusão

NotCover (O1: OG, O2: Set (OG), Boundary: Boolean) : Boolean

Igualdade

Equal (O1: Set (OG)) : Boolean

Não Igualdade

NotEqual (O1: Set (OG)) : Boolean

Interseção

Overlap (O1: OG, O2: Set (OG)) : Boolean

Disjunção - Não Interseção

Disjoint (O1: OG, O2: Set (OG)) : Boolean
NotOverlap (O1: OG, O2: Set (OG)) : Boolean

Cruzamento

Span (O1: OG, O2: Set (OG)) : Boolean

Não Cruzamento

NotSpan (O1: OG, O2: Set (OG)) : Boolean

Transações primitivas de busca topológica

Busca por proximidade mínima

Nearest (O: OG, TypeOG: TO) : Set (OG)

Busca por adjacência

Adjacent (O: OG, TypeOG: TO) : Set (OG)

Busca por exclusão de adjacência

NotAdjacent (O: OG, TypeOG: TO) : Set (OG)

Busca por inclusão – relacionamento topológico está contido

Inside (O: OG, TypeOG: TO, Boundary: Boolean) : Set (OG)

Busca por exclusão de inclusão – relacionamento topológico está contido

NotInside (O: OG, TypeOG: TO, Boundary: Boolean) : Set (OG)

Busca por inclusão – relacionamento topológico contém

Embody (O: OG, TypeOG: TO, Boundary: Boolean) : Set (OG)

Busca por exclusão de inclusão – relacionamento topológico contém

NotEmbody (O: OG, TypeOG: TO, Boundary: Boolean) : Set (OG)

Busca por interseção

Intersect (O: OG, TypeOG: TO) : Set (OG)

Busca por exclusão de interseção

NotIntersect (O: OG, TypeOG: TO) : Set (OG)

Busca por cruzamento

Cross (O: OG, TypeOG: TO) : Set (OG)

Busca por exclusão de cruzamento

NotCross (O: OG, TypeOG: TO) : Set (OG)

Transações primitivas topológicas escalares

Interseção escalar

WhereIntersect (O1: OG, O2: Set(OG)) : Set ((x,y))

Transações primitivas escalares

Distância mínima entre dois OGs

MinDistance (O1, O2: OG) : Float

Área de um OG-2D

Area (O: OG-2D) : Float

Perímetro de um OG-2D

Perimeter (O: OG-2D) : Float

Comprimento de um OG-1D

Length (O: OG-1D) : Float

Transações primitivas baseadas em conjunto

União

Union (O: Set(OG)) : Set(OG)

Interseção

Intersection (O: Set(OG)) : Set(OG)

Diferença

Difference (O1, O2: Set(OG)) : Set(OG)

Transações primitivas que envolvem visualização gráfica

Visualização gráfica

Display (O: Set (OG), Bottom: Set ((x,y))

Transações primitivas diversas

Determinação de coordenadas geográficas

Bearing (O: OG) : Set ((x,y))

Geração de um OG-1D a partir de OGs-0D

Segment (O1, O2: OG-0D) : OG-1D

Carga do sistema

LoadSystem

Seleção convencional

Select (E: Exp, TypeOG: TO) : Set (OG)

Número de OGs

Number (O: Set(OG), TypeOG: TO) : Integer

Gravação

Save (O: Set (OG))

Transações primitivas específicas do formato raster

Agrupamento

GroupCell (M: Set(Cell)) : Set (OG)

Mudança da resolução espacial

ResolutionCell (M: Set(Cell), NewLengthCell: Integer) : Set (Cell)

Comparação de matrizes

CompareCell (M1, M2: Set(Cell)) : TypeCompare

Número de células

NumberCell (M: Set(Cell)) : LongInteger

Célula de valor mínimo

MinValueCell (M: Set(Cell)) : Set (Cell)

Valor médio de categoria

AverageCell (M: Set(Cell)) : Float

Transações mistas

Conversão do formato *raster* para o formato vetorial

RasterVector (M: Set(Cell)) : Set (OG)

Conversão do formato vetorial para o formato *raster*

VectorRaster (O: Set (OG), LengthCell: Integer, F: Function) : Set(Cell)

Superposição visual

VisualOverlay (O: Set (OG), M: Set (Cell))

Apêndice B

Questionário

Caro usuário de Sistemas de Informações Geográficas (SIGs),

Este questionário visa à identificação dos requisitos básicos que caracterizam típicas aplicações georeferenciadas. A caracterização destes requisitos propiciará a criação de um *benchmark*, o qual irá analisar o desempenho de diferentes SIGs . Este trabalho está sendo desenvolvido na *Universidade Estadual de Campinas, Unicamp*, como parte do trabalho de dissertação de mestrado. Para isto, pedimos gentilmente o preenchimento deste questionário e antecipadamente agradecemos a sua ajuda.

1. Dados Gerais.

Instituição: _____
Endereço: _____
Telefone: _____
Fax: _____
E-Mail: _____
Pessoa de contato: _____

2. Descreva as principais aplicações georeferenciadas de sua instituição.

(espaço para respostas)

3. Quais SIGs são utilizados para suportar as aplicações especificadas na resposta anterior ? (Selecione quantas alternativas forem necessárias)

[ ] Apic
[ ] Arc/Info
[ ] AtlasGIS
[ ] AtlasMap
[ ] Erdas
[ ] Grass
[ ] Idrisi
[ ] InfoCad
[ ] Intergraph MGE
[ ] MapInfo
[ ] SGI – Inpe
[ ] Spans
[ ] Spring
[ ] Tactician
[ ] Vision/GeoVision
[ ] Outros: _____

4. Indique os tipos de dados armazenados.
(Selecione quantas alternativas forem necessárias)

[ ] Convencionais (Alfanumérico, Numérico – Inteiro ou Real, *Booleano* e Data)
[ ] Imagem
[ ] Espacial no formato *raster* (quadrático)
[ ] Espacial no formato vetorial (Ponto/Linha/Polígono)
[ ] Outros: _____

5. Indique os tamanhos de célula freqüentemente utilizadas para o armazenamento *raster*.

1 - _____ km x km

2 - _____ km x km

6. Os dados são armazenados em: (Selecione somente uma alternativa)

[ ] Arquivos internos ao SIGs – sem o auxílio de um SGBD
[ ] Arquivos Flat – sem o auxílio de um SGBD
[ ] Um SGBD, o qual armazena todos os tipos de dados
Especifique (incluindo a versão): _____
[ ] Vários SGBDs, onde cada SGBD armazena um conjunto específico de tipos de dados
Especifique (incluindo a versão):
1 - SGBD: _____ Tipos de Dados: _____
2 - SGBD: _____ Tipos de Dados: _____
[ ] Outros: _____

7. Indique os tipos de plataformas de *hardware* utilizados para suportar as aplicações georeferenciadas de sua instituição. Descreva também o modelo de cada tipo de plataforma e a quantidade de máquinas correspondente (Exemplo: *plataforma*: PC, *modelo*: 486 DX2-66 MHz, *quantidade*: 4).

Plataforma de Hardware	*N#*	*Modelo*	*Quantidade*
PC	1		
	2		
Workstation	3		
	4		
Mainframe	5		
	6		
Outros	7		
	8		

8. Especifique as configurações das diferentes plataformas de *hardware* descritas no item anterior.

N# Plataforma de hardware	*Memória Principal (Mbytes)*	*Memória Secundária (Mbytes)*	*Memória Cache (Kbyes)*	*Processador (Tipo, MHz)*
1				
2				
3				
4				
5				
6				
7				
8				

9. As plataformas de *hardware* descritas são caracterizadas por serem:
(Selecione quantas alternativas forem necessárias).

[ ] Stand-Alone
[ ] Cliente-Servidor
[ ] Rede Ponto-a-Ponto
[ ] Outros: _____

10. Descreva os sistemas operacionais utilizados para suportar as aplicações georeferenciadas de sua instituição.
(Selecione quantas alternativas forem necessárias).

[ ] DOS (MS-DOS Like)
[ ] OS/2
[ ] UNIX versão: _____
[ ] VMS
[ ] Windows 3.1 (+ DOS-Like)
[ ] Windows NT
[ ] Windows for Workgroups
[ ] Outros: _____

11. Indique o número de pessoas trabalhando em cada uma das áreas abaixo:

Administração do Sistema: _____
Desenvolvimento de Aplicativos: _____
Entrada/Coleta de dados: _____
Usuários Finais: _____
Outros: _____

12. Indique as funções necessárias para as aplicações georeferenciadas de sua instituição. (Selecione quantas alternativas forem necessárias)

[ ] Coleta de Dados
[ ] Funções fornecidas por um SGBD, tais como: alteração, remoção, inserção e consulta de dados.
[ ] Funções Analíticas
[ ] Funções Cartográficas
[ ] Saída de Dados – *plottagem*, impressão ou visualização na tela

13. Indique os métodos utilizados para a aquisição de dados. (Selecione quantas alternativas forem necessárias)

[ ] Não é efetuado
[ ] Fotografia
[ ] Sensoriamento Remoto, Imagens de Satélite
[ ] Digitalização de Mapas, "*Scannerização*"
[ ] Observação em Campo/GPS
[ ] Outros: _____

14. Descreva sucintamente os principais problemas encontrados durante a fase coleta de dados.

(espaço para respostas)

15. Indique os passos executados durante a fase de pré-processamento dos dados coletados. Especifique as conversões necessárias, tanto implícitas quanto explícitas.

(espaço para respostas)

16. Qual a metodologia utilizada para a modelagem conceitual do banco de dados ?
(Selecione somente uma alternativa)

[ ] Não é utilizado nenhum modelo – ausência de modelo conceitual
[ ] Modelo Entidade-Relacionamento (ER)
[ ] Modelo Entidade-Categoria-Relacionamento (ECR)
[ ] Técnica de Modelagem de Objetos (OMT)
[ ] Outros: _____

17. Alterações efetuadas no modelo conceitual refletem automaticamente no modelo lógico e físico ?

[ ] Ausência de modelo conceitual
[ ] Sim
[ ] Não
[ ] em Termos
Especifique:
_____

18. Qual o modelo de dados lógico utilizado ?
(Caso o sistema seja heterogêneo, selecione quantas alternativas forem necessárias)

[ ] Não é utilizado nenhum modelo lógico
[ ] Modelo Relacional
[ ] Modelo de Rede
[ ] Modelo Hierárquico
[ ] Modelo Orientado a Objetos.
Especifique:
_____
[ ] Outros: _____

19. Quais os tipos de linguagem de consulta utilizados pelas aplicações georeferenciadas de sua instituição ?
(Selecione quantas alternativas forem necessárias)

[ ] SQL-Like
[ ] QBE-Like
[ ] Linguagem Visual
[ ] Estrutura de Menus
[ ] Outros: _____

20. Indique os operadores geométricos utilizados pelas aplicações de sua instituição.
Operadores geométricos retornam como resultado um valor escalar.
(Selecione quantas alternativas forem necessárias)

[ ] Área
[ ] Perímetro
[ ] Comprimento
[ ] Distância
[ ] Outros: _____

21. Indique os operadores topológicos utilizados pelas aplicações de sua instituição.
Operadores topológicos retornam como resultado um valor booleano (0 ou 1) e sempre são aplicados sobre 2 objetos geográficos.
(Selecione quantas alternativas forem necessárias)

[ ] Interseção
[ ] Disjunção
[ ] Inclusão
[ ] Adjacência
[ ] Proximidade
[ ] Corte/Através
[ ] Outros: _____

22. Indique os operadores que retornam um outro objeto (ou conjunto) como resultado.
(Selecione quantas alternativas forem necessárias)

[ ] Interseção – entre 2 objetos geográficos
[ ] União – entre 2 objetos geográficos
[ ] Complemento – de 1 objeto geográfico
[ ] Contorno/Borda – de 1 objeto geográfico
[ ] Interior – de 1 objeto geográfico
[ ] Vizinhança – de 1 objeto geográfico – retorna todos os objetos geográficos que estejam dentro de uma determinada área.
[ ] Outros: _____

23. É efetuado o agrupamento de predicados espaciais (operadores geométricos, topológicos, entre outros) e predicados convencionais (alfanuméricos, numéricos, *booleanos* e data) conjuntamente em consultas ?
(Selecione somente uma alternativa)

[ ] Sim
[ ] Não
[ ] em Termos
Especifique:
_____

24. Indique as operações efetuadas durante o gerenciamento dos dados armazenados – para todos os tipos de dados.
(Selecione quantas alternativas forem necessárias)

[ ] Extração de dados/Consulta de dados
[ ] Inserção de novos dados
[ ] Alteração de dados já armazenados
[ ] Remoção de dados antigos
[ ] Proteção de dados
[ ] Recuperação de dados em caso de queda do sistema
[ ] Outros: _____

25. A operação de consulta de dados é:
(Selecione somente uma alternativa)

[ ] Dependente de sua visualização, ou seja, sempre é gerado uma saída para a tela, para o plotter ou para a impressora
[ ] Independente de sua visualização – sendo que o resultado gerado é guardado em memória principal ou em memória secundária (discos)

26. Descreva as principais transações.

(espaço para respostas)

27. Existe a necessidade de controle de versões ? Quanto o fator tempo é importante para as aplicações georeferenciadas sendo desenvolvidas em sua instituição ?

(espaço para respostas)

28. Indique as funções analíticas utilizadas pelas aplicações georeferenciadas de sua instituição.
(Selecione quantas alternativas forem necessárias)

[ ] Cálculo de Área/Perímetro
[ ] Cálculo de Distância
[ ] Reclassificação de Polígonos
[ ] Determinação de interior/fronteira de objetos geográficos bidimensionais
[ ] Superposição
[ ] Superposição conjunta de dados espaciais no formato *raster* e no formato vetorial
[ ] Análise de Ponderação
[ ] Transformação *raster* ⇒ vetorial
[ ] Transformação vetorial ⇒ *raster*
[ ] Mudança de Resolução Espacial
[ ] Criação de Zonas de *Buffer*
[ ] Modelagem Digital de Terreno
[ ] Cálculo envolvendo Redes
[ ] Análise Geoestatística
[ ] Projeção de Mapas
[ ] Outros: _____

29. Indique como é feito o acréscimo de novas funções analíticas.
(Selecione quantas alternativas forem necessárias, caso a primeira opção não seja escolhida)

[ ] Não é possível o acréscimo de nova funções analíticas
[ ] por Programação
[ ] por Menus
[ ] Outros: _____

30. Indique, dentre as áreas abaixo, quais são necessárias às aplicações georeferenciadas de sua instituição. Também descreva, para cada uma delas, as atividades freqüentemente efetuadas. Caso haja a necessidade de uso de pacotes específicos, indique-os.

[ ] Processamento Digital de Imagens (PDI)

(espaço para respostas)

[ ] Simulação

(espaço para respostas)

31. Indique as principais tarefas efetuadas durante a fase de saída de dados.
(Selecione quantas alternativas forem necessárias)

[ ] Visualização gráfica dos dados
[ ] Visualização gráfica de dados implícitos relacionados a um dado objeto geográfico
[ ] Uso de múltiplas representações gráficas para um dado objeto geográfico, dependendo do contexto que ele está associado
[ ] *Zoom*
[ ] Edição gráfica de mapas
[ ] Uso de apenas uma janela
[ ] Uso de múltiplas janelas para a visualização dos dados, sem a necessidade de alteração simultânea destes dados nas diversas janelas ativas
[ ] Uso de múltiplas janelas para a visualização dos dados, com a necessidade de alteração simultânea destes dados em todas as janelas ativas
[ ] Visualização gráficas apenas em tons de cinza
[ ] Visualização gráfica colorida
[ ] Visualização gráfica baseada em normas cartográficas, com grande número de estilos e cores de linhas, além de regras bem definidas para a rotulação dos objetos
[ ] Configuração dinâmica de cor, textura e estilo de objetos geográficos
[ ] Plottagem
[ ] Impressão de relatórios alfanuméricos
[ ] Impressão de relatórios gráficos
[ ] Impressão/*plottagem* em lote (*batch*)
[ ] Exportação de dados *raster* em diversos formados de codificação
[ ] Outros: _____

32. Indique os modelos de placas gráficas utilizados.

(espaço para respostas)

33. Indique as funcionalidades providas pela interface.
(Selecione quantas alternativas forem necessárias)

[ ] Estrutura de menus – sem flexibilidade de configuração
[ ] Estrutura de menus – com flexibilidade de configuração
[ ] Uso de janelas, sendo permitido mudanças de posicionamento, tamanho e estilo
[ ] Associações dinâmicas de funções analíticas a botões
[ ] Linguagem de comando
[ ] Linguagem de programação/macros
[ ] Outros: _____

34. Indique os tipos de ajuda.
(Selecione quantas alternativas forem necessárias)

[ ] Ajuda on-line – tipo hipertexto
[ ] Ajuda on-line – texto contínuo
[ ] Manuais/Guias de Referência
[ ] Outros: _____

35. Espaço para comentários adicionais.

(espaço para respostas)

Por fim, nós gostaríamos novamente de agradecer a atenção concedida para o preenchimento deste questionário. Gostaríamos também de agradecer a colaboração de Ardemírio de Barros Silva (*Instituto de GeoCiências – Unicamp*), Claudia Bauzer Medeiros (*DCC – Unicamp*), Cristina Dutra de Aguiar (*DCC – Unicamp*) e Telma Regina Rodrigues (*USP – Ribeirão Preto*) pela ajuda concedida na elaboração deste questionário.

A devolução deste questionário deve ser enviada para os autores abaixo, no endereço, e-mail ou número de fax especificados.

Ricardo Rodrigues Ciferri / Geovane Cayres Magalhães (*DCC – Unicamp)*
Rua Jean Nassif Mokarzel, 40 apt 31 – Campinas – Barão Geraldo – Centro – Cep: 13084-480
Fone: (0192) 39-0446
E-Mail: rrc@dcc.unicamp.br, geovane@dcc.unicamp.br
Fax: (016) 633-1010 Ramal 550