VIII Brazilian Symposium on GeoInformatics, Campos do Jordão, Brazil, November 19-22, 2006, INPE, p. 249-258.

Um método de sobreposição de mapas mais eficiente para a TerraLib

Vinícius Lopes Rodrigues[1], Marcus Vinícius Alvim Andrade[1], Gilberto Ribeiro de Queiroz[2], Mirella Antunes de Magalhães[1]

[1]Universidade Federal de Viçosa, Depto. de Informática
Viçosa, Minas Gerais, Brasil, 36570-000

[2]INPE - Instituto Nacional de Pesquisas Espaciais,
Caixa Postal 515 – São José dos Campos, SP, Brasil, 12201

{vlopes,marcus,mirella}@dpi.ufv.br, gribeiro@dpi.inpe.br

Abstract. *This paper describes the implementation of a more efficient method to overlay two maps using the TerraLib library. As shown by some tests, this method is much more efficient than the original method and this fact is more evident when more complex maps are used.*

Resumo. *Este artigo descreve a implementação de um método mais eficiente para a sobreposição utilizando a biblioteca TerraLib. Como os testes mostraram, o método proposto é consideravelmente mais eficiente do que o método original e essa eficiência se torna mais evidente à medida que a complexidade dos mapas a serem sobrepostos aumenta.*

1. Introdução

Sistemas de Informações Geográficas (SIGs) são sistemas automatizados usados para armazenar, analisar e manipular dados geográficos, isto é, dados que representam objetos e fenômenos em que a localização geográfica é uma característica inerente à informação e indispensável para analisá-la [Câmara et al (2000), Rigaux et al (2002), Steinberg e Steinberg (2006)]. Estes sistemas envolvem problemas de várias áreas, como geometria computacional, computação gráfica, banco de dados, engenharia de software, etc.

Inserido neste contexto encontra-se o projeto TerraLib, que vem sendo desenvolvido por meio de uma parceria entre o INPE (Instituto Nacional de Pesquisas Espaciais), a Tecgraf/PUC-Rio (Grupo de Computação Gráfica da Universidade Católica do Rio de Janeiro) e a FUNCATE (Fundação de Ciência, Aplicações e Tecnologia Espaciais) [Daltio et al (2004), Queiroz (2003), TerraLib (2006)]. A TerraLib consiste em uma biblioteca de classes que possibilita o desenvolvimento de aplicativos geográficos customizados. Ela encontra-se disponível na Internet [TerraLib (2006)] como sistema de código aberto, permitindo o desenvolvimento, de forma colaborativa, de aplicativos e ferramentas geográficas.

O núcleo da TerraLib contém, como qualquer SIG, um módulo geométrico responsável por dar suporte às operações geométricas requisitadas pelas camadas superiores do SIG e dentre as diversas operações que compõem este módulo, uma das operações mais

importante é a que realiza a sobreposição [Berg et al (2000), Kriegel et al (1991), Rodrigues et al (2006), Rodrigues et al (2005)] entre dois polígonos. Utilizando essa operação, pode-se obter uma função para sobrepor dois mapas inteiros. Esta função é utilizada em situações em que é necessário combinar ou comparar dados armazenados em camadas de informações distintas. Por exemplo, considere uma consulta como "calcular a área de desmatamento ocorrido dentro de cada reserva indígena". Para executar esta análise, é necessário combinar uma camada de objetos poligonais (os limites das reservas indígenas) com outra (o mapa de desmatamento), para se obter uma nova camada, a partir da qual é possível realizar a análise desejada (área de desmatamento nas reservas indígenas). Veja a figura 1.

Figura 1. Exemplo de aplicação da sobreposição de mapas

Visto que esta operação normalmente é realizada diversas vezes podendo envolver mapas de tamanhos consideráveis então, por questões de eficiência do sistema como um todo, é importante que esta operação seja implementada de maneira mais eficiente. Diante disto, o objetivo deste trabalho é descrever a implementação de um método alternativo (mais eficiente) para a sobreposição de mapas na TerraLib.

2. Algoritmo de sobreposição atual

O algoritmo de sobreposição de mapas existente na TerraLib encontra-se implementado sobre um operador elementar, denominado *TeOverlay*, que efetua a sobreposição de dois conjuntos de polígonos. Esta operação (*TeOverlay*) é utilizada para compor uma outra, de mais alto nível, chamada *TeGeoOpOverlayIntersection*, que realiza a sobreposição de duas camadas de informação, preocupando-se não só com as novas geometrias geradas mas também com os atributos associados a elas. O aplicativo geográfico TerraView utiliza esta operação, fornecendo aos seus usuários a possibilidade de combinar informações provenientes de duas camadas distintas.

A sobreposição de polígonos (*TeOverlay*) por se tratar de uma implementação de mais baixo nível, encontra-se no núcleo da TerraLib (módulo *kernel*) enquanto a sobreposição de dois mapas (*TeGeoOpOverlayIntersection*), função de mais alto nível, encontra-se no módulo de algoritmos (módulo *function*s).

A TerraLib não realiza a sobreposição de polígonos de maneira generalizada. O que é determinado é a união, interseção ou a diferença entre polígonos, sendo este processamento realizado segundo o algoritmo proposto por Margalit e Knott (1989). Detalhes sobre a implementação deste operador podem ser consultados em Queiroz (2003).

A seguir, será descrita a operação de sobreposição de mapas implementada na parte de algoritmos da Terralib e utilizada no TerraView. Esta operação será tomada como base nos testes para a avaliação da eficiência do método proposto neste trabalho.

Dados dois mapas de entrada M_1 e M_2 a serem sobrepostos, suponha que o mapa M_1 possui uma menor quantidade de geometrias. O primeiro passo do processo é determinar os polígonos dos dois mapas que têm chance de se interceptarem, isto é, os polígonos cujas caixas envolventes (*bounding box*) se sobrepõem. Esta tarefa é realizada através de consultas SQL, que utilizam o gerenciador de banco de dados que está sendo utilizado pela aplicação (por exemplo, *MySQL, PostgreSQL, Oracle,* etc). Mais precisamente, para cada polígono P_i de M_1, é realizada uma consulta espacial entre os polígonos de M_2 para obter os polígonos cujas caixas envolventes sobrepõem a caixa envolvente de P_i.

O próximo passo é utilizar a função *TeOverlay* da TerraLib para obter a interseção entre cada polígono P_i e os polígonos que podem interceptá-lo e que foram obtidos na etapa anterior, utilizando o algoritmo de Margalit e Knott. Para realizar esta sobreposição, após um pré-processamento para orientar os polígonos adequadamente, os pontos de interseção entre P_i e os outros polígonos são computados e depois as fronteiras dos polígonos são fragmentadas considerando os pontos de interseção obtidos.

Daí, os fragmentos da fronteira de um polígono são classificados em relação aos outros polígonos para determinar se o fragmento está “fora”, “dentro” ou na “fronteira” do polígono. Para isso, é aplicado o teste para verificar se um ponto do fragmento está “fora”, “dentro” ou na “fronteira” do respectivo polígono. Finalmente, os polígonos de saída são construídos considerando cada fragmento de acordo com a operação desejada (união, interseção ou diferença).

É importante notar que primeiro passo deste processo envolvendo operações de consulta ao banco de dados pode tomar um tempo considerável, principalmente, naquelas situações em que os mapas a serem sobrepostos contenham muitos polígonos com várias interseções. Um exemplo onde tal situação ocorre é quando se deseja realizar a sobreposição de dois mapas temáticos de uma mesma região.

Além disso, analisando a etapa da sobreposição propriamente dita, pode-se observar que um mesmo polígono pode ser processado mais de uma vez, uma para cada chamada a *TeOverlay* envolvendo os candidatos a interseção.

Na seção a seguir, é apresentada uma alternativa mais eficiente para implementação da operação de sobreposição de mapas na TerraLib.

3. Algoritmo de sobreposição alternativo (mais eficiente)

Para tornar o processo de sobreposição mais eficiente, a idéia básica foi adaptar o método de sobreposição existente na Terraview redefinindo o processo de obtenção das interseções entre os polígonos que compõem os dois mapas a serem sobrepostos com o intuito de reduzir o número de consultas ao banco de dados. Para obter as interseções foi utilizado o algoritmo baseado no método de varredura proposto por Bentley e Ottman [Berg et al (2000), Daltio et al (2004), Preparata e Shamos (1989), Rezende e Stolfi (1994)], descrito na seção 3.1.

Desta forma, o primeiro passo do método proposto é efetuar uma consulta ao banco de dados para obter todos os segmentos dos dois mapas a serem sobrepostos e com isso, construir dois conjuntos de segmentos. Daí, o método de varredura é utilizado para obter todos os pontos de interseção entre os segmentos destes dois conjuntos e posteriormente, os métodos da TerraLib são utilizados para construir as regiões correspondentes à sobreposição utilizando os pontos de interseção previamente computados.

Essas regiões são obtidas utilizando o algoritmo da TerraLib que efetua a sobreposição de dois polígonos sendo que este algoritmo foi adaptado para evitar que os pontos de interseção sejam novamente calculados. Assim, o algoritmo passa a receber como parâmetros os dois polígonos a serem sobrepostos e também os pontos de interseção entre eles. Note que este processo deve ser aplicado a cada par de polígonos que se interceptam e para identificar estes pares de polígonos foi construída uma matriz para armazenar a lista de pontos de interseção entre cada par de polígonos. Mais precisamente, se os mapas a serem sobrepostos têm m_1 e m_2 polígonos então é construída uma matriz de dimensão m_1 x m_2 sendo que na posição na posição (i,j) é armazenada uma lista contendo as interseções entre o polígono i e o polígono j.

Daí, para cada par de polígonos que se interceptam, isto é, para cada posição (i,j) da matriz cuja lista não é vazia, é utilizada a função adaptada da TerraLib que realiza a sobreposição entre os polígonos (i,j) sendo que o objetivo desta função é fragmentar a fronteira dos dois polígonos e combinar as respectivas regiões. Como dito acima, a função da TerraLib que realiza esta tarefa foi alterada para receber os pontos de interseção previamente calculados evitando assim o cálculo destas interseções.

É importante observar que para os pares de polígonos que não se interceptam, isto é, para as posições da matriz cuja lista é vazia, é necessário verificar se um dos polígonos está inteiramente contido dentro do outro; essa operação é realizada utilizando a função da TerraLib que verifica a sobreposição entre as caixas envolventes dos dois polígonos.

3.1 Algoritmo para determinar as interseções entre segmentos

A idéia básica deste algoritmo consiste em imaginar a existência de uma reta vertical r que varre o plano da esquerda para a direita, ou seja, do menor valor da abscissa para o maior valor e durante esta varredura alguns eventos são identificados e tratados, veja a figura 2. Cada um destes eventos serve de gatilho para que um conjunto de operações seja realizado. Nesta implementação, um evento é representado por um ponto e pelo(s) segmento(s) que se relaciona(m) com este. Para cada evento, existem três listas de segmentos, descritas a seguir:

1 – Segmentos à esquerda: quando o ponto do evento é o extremo direito de um segmento, este segmento é inserido nessa lista.

2 – Segmentos à direita: quando o ponto do evento é o extremo esquerdo de um segmento, este segmento é inserido nessa lista.

3 – Segmentos de interseção: quando o ponto do evento é um ponto de interseção, os segmentos que se interceptaram originando esse ponto são inseridos nessa lista.

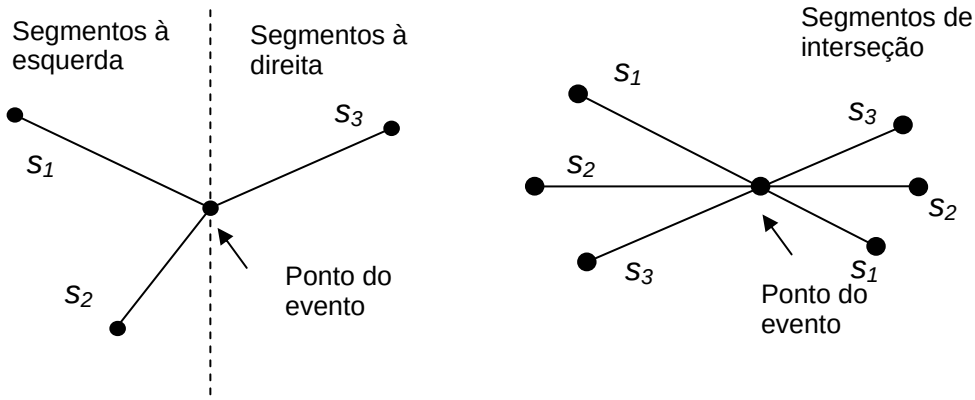


Figura 2. Exemplos de eventos

Os eventos são armazenados uma lista *E*, denominada agenda de eventos, que são mantidos ordenados em ordem crescente pelo valor da abscissa de modo que eles sejam tratados na seqüência adequada. No início do processamento, essa agenda é inicializada inserindo-se os eventos associados aos segmentos de entrada.

Além da agenda *E*, o algoritmo também utiliza uma lista *L* para indicar os segmentos ativos a cada instante. Um segmento é dito ativo quando ele é interceptado pela reta de varredura; mais precisamente, quando a abscissa da reta de varredura está entre as abscissas dos vértices extremos do segmento. Estes segmentos ativos definem o conjunto de segmentos a serem tratados a cada evento. Os segmentos na lista *L* são ordenados pelo valor da coordenada *y* de cada segmento, considerando a posição da reta de varredura. Note que, uma alteração na posição relativa dos elementos (segmentos) na lista *L* ocorre somente quando há um evento; isto é, a ordenação dos segmentos é alterada apenas quando um novo segmento começa (e é inserido na lista *L*), quando um segmento termina (e é retirado da lista *L*) ou quando há uma interseção. Veja figura 2.

Na prática, a varredura realizada pela reta *r* consiste em percorrer a lista de eventos e para cada elemento da lista deve-se executar um conjunto de operações para tratar os seus segmentos associados conforme descrito a seguir (para ilustrar estas operações considere a figura 2):

1 – Segmentos à esquerda: estes segmentos devem ser desativados, isto é, o segmento deve ser removido da lista *L*; neste caso, é verificado se os segmentos que eram os vizinhos de cima e de baixo do respectivo se interceptam..

2 – Segmentos à direita: estes segmentos se tornam ativos, isto é, o segmento é inserido na lista *L*; neste caso, é verificado se o respectivo segmento intercepta o segmento imediatamente acima dele (vizinho de cima) e se intercepta o segmento imediatamente abaixo dele (vizinho de baixo).

3 – Segmentos de interseção: quando a linha de varredura *r* alcança um evento que tem a lista de segmentos de interseção não vazia, os segmentos da lista devem ser trocados de posição dois a dois na lista *L*, trocando o último segmento com o primeiro, o penúltimo com o segundo, e assim sucessivamente. Por exemplo, suponha que os segmentos s_1 e s_3 se interceptam num ponto *p* como mostrado na figura 2. Quando o evento correspondente a esta interseção for tratado então o segmento que estava em baixo, antes da interseção, passará a estar em cima e vice-versa. Com essa mudança, a relação de vizinhança entre os segmentos também é alterada e, portanto, é necessário verificar se o segmento que agora está em cima intercepta o seu novo vizinho de cima e se o segmento que agora está em baixo intercepta o seu novo vizinho de baixo.

Em todos os casos descritos acima, quando uma interseção é obtida, esta interseção é inserida na agenda de eventos para que este evento venha a ser tratado; além disso, o ponto de interseção e os dois segmentos que o geraram são inseridos numa lista que, ao final do processo, conterá todas as interseções existentes entre os segmentos.

Por questões de eficiência, a lista de segmentos ativos *L* é armazenada utilizando uma árvore Red-Black [Cormen et al (1990)] que permite obter facilmente os vizinhos "de cima" e "de baixo" de um segmento.

Uma questão importante na implementação desse algoritmo é que existem algumas situações especiais que devem receber um tratamento diferenciado e que exigem um certo esforço de implementação.

Maiores detalhes da implementação desse algoritmo podem ser obtidos em [Rodrigues et al (2006), Rodrigues et al (2005)], onde esse mesmo algoritmo é apresentado em uma abordagem para o tratamento exato de mapas.

4. Resultados

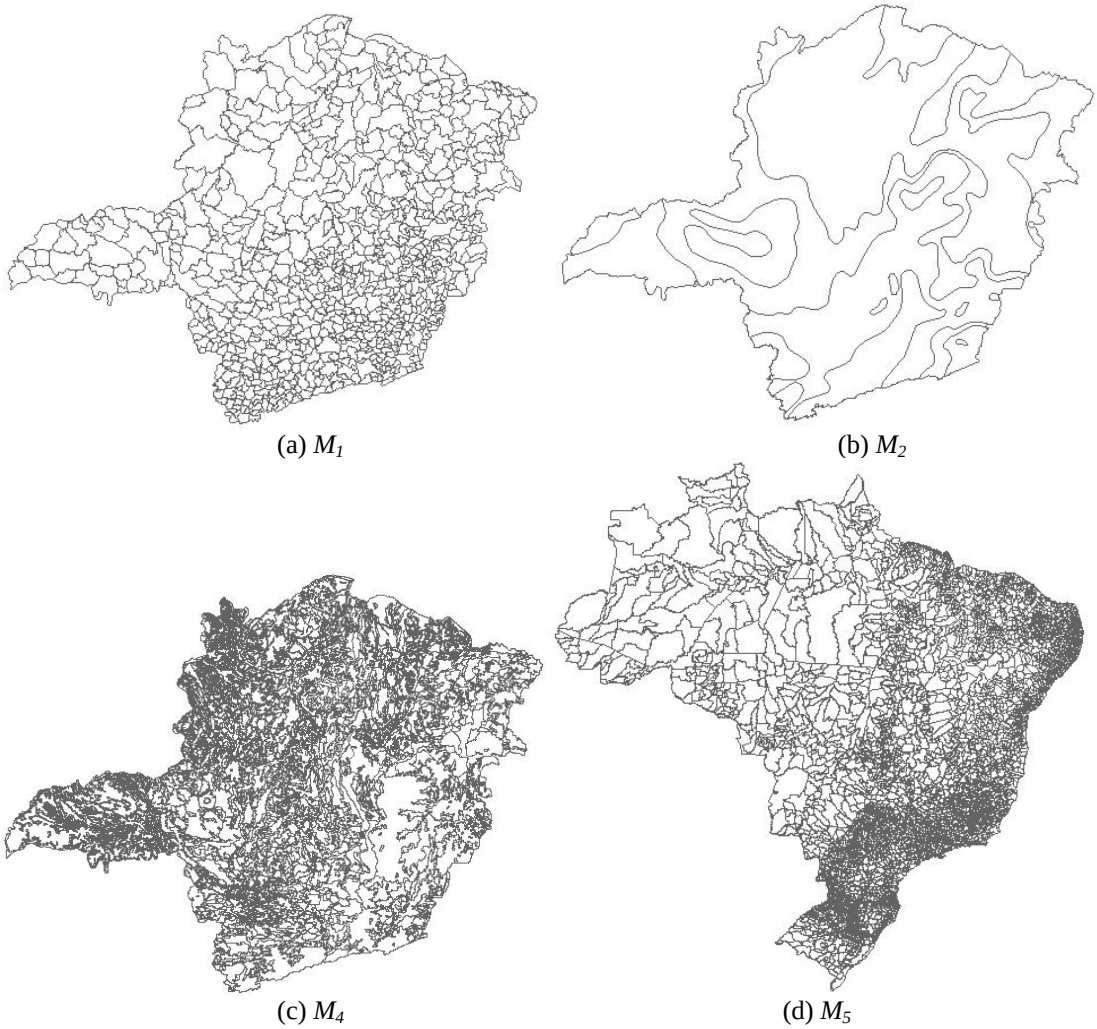
Os testes de avaliação do método apresentado neste trabalho foram realizados utilizando um computador com processador AMD Athlon™ 1.31 GHz com 512Mb de memória RAM e com sistema operacional Microsoft Windows XP. A implementação alternativa foi desenvolvida em Microsoft Visual C++ 6.0 com os dados armazenados em um banco de dados Microsoft Access no formato Terralib.

Para a realização dos testes, foram utilizados os seguintes mapas (veja a figura 3):

Tabela 1. Mapas utilizados nos testes de comparação

Número	Mapa	Número de regiões	Número de segmentos
M_1	Municípios MG	853	61237
M_2	Municípios MG (transladado[1])	853	61237
M_3	Temperatura MG	20	4659
M_4	Tipos de Solos MG	3067	858696
M_5	Municípios Brasil	5560	571747
M_6	Municípios Brasil (transladado[2])	5560	571747

[1] Transladado 0,5 unidade para a direita na projeção NoProjection do TerraView
[2] Transladado 1,2 unidade para a direita na projeção NoProjection do TerraView



(a) M_1 (b) M_2

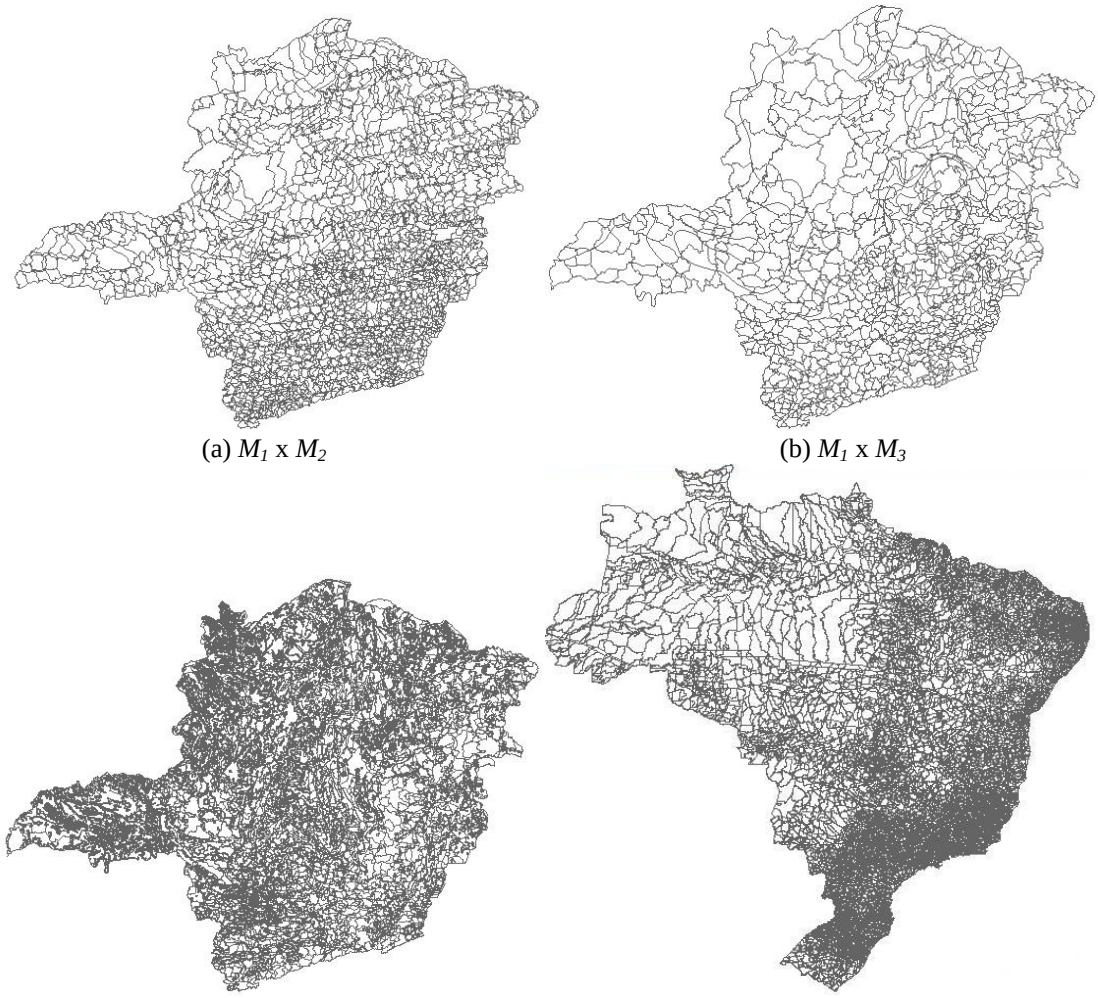
(c) M_4 (d) M_5

Figura 3. Mapas utilizados nos testes de comparação

Com esses mapas, foram realizados os seguintes casos de teste (os mapas resultantes são mostrados na figura 4):

Tabela 2. Resultados dos testes de comparação

	Número		Tempo (ms)	
Sobreposição	**Interseções**	**Polígonos de saída**	**Original**	**Proposto**
M_1 x M_2	15382	3885	184516	25037
M_1 x M_3	3678	1323	21591	18626
M_1 x M_4	39317	7437	433884	172007
M_5 x M_6	110232	25657	1554776	307913



(a) M_1 x M_2 (b) M_1 x M_3

(c) M_1 x M_4 (d) M_5 x M_6

Figura 4. Resultados Obtidos

Como se pode perceber, os tempos obtidos utilizando o método proposto são consideravelmente melhores do que os obtidos utilizando o algoritmo original. E esta diferença se torna mais evidente à medida que o tamanho (número de segmentos e regiões) dos mapas aumenta. Por exemplo, a sobreposição do mapa de municípios do Brasil com o mesmo mapa transladado foi realizada pelo algoritmo original em aproxidamente 26 minutos enquanto o método proposto obteve o mesmo resultado em apenas 5 minutos.

5. Conclusão e trabalhos futuros

Este trabalho descreve a implementação de um outro método para a realização da sobreposição de dois mapas utilizando a biblioteca TerraLib que utiliza o algoritmo de varredura para a obtenção das interseções. Como os testes comprovaram, o método proposto é consideravelmente mais eficiente do que o método original. Esta maior eficiência ocorre principalmente porque o método proposto elimina a realização das diversas consultas espaciais ao banco de dados que são realizadas pelo método original para

determinar os polígonos que podem se interceptar. Além disso, o método original obtém os pontos de interseção entre os polígonos por partes enquanto o método proposto obtém todos os pontos de interseção processando todos os segmentos em grupo. Isto evita que um mesmo segmento seja processado mais de uma vez

Como trabalhos futuros, a intenção é implementar o algoritmo de sobreposição proposto por Kriegel et al (1991) que também se baseia no método da varredura de forma semelhante ao algoritmo de interseção utilizado neste trabalho. A expectativa é que os resultados venham a ser ainda melhores do que os obtidos neste trabalho.

Agradecimentos

Este trabalho foi parcialmente financiado pelo Conselho Nacional de Desenvolvimento Científico e Tecnológico – CNPq e pela Fundação de Amparo à Pesquisa do Estado de Minas Gerais – FAPEMIG.

Referências

Berg, M., Kreveld, M. Van, Overmars, M., e Schwarzkopf, O. (2000), "Computational Geometry, Algorithms And Applications", Springer Verlag.

Câmara, G. et al. (2000), "Terralib: Technology In Support of GIS Innovation", II Workshop Brasileiro de Geoinformática, Geoinfo, Caxambu.

Cormem, T. H., Leiserson, C. E. e Rivest, R. L. (1990), "Introduction to Algorithms", The MIT Press.

Daltio, J., Andrade, M. V. A. e Queiroz, G. R. (2004), "Tratamento de Erros de Arredondamento na Biblioteca TerraLib", VI Simpósio Brasileiro de Geoinformática, Campos do Jordão, SP.

Kriegel, H. P., Brinkhoff, T. e Schneider R. (1991) "An Efficient Map Overlay Algorithm Based on Spatial Access Methods and Computational Geometry", Workshop Proceedings, Capri.

Margalit, A. e Knott, G. D. (1989), "An algorithm for computing the union, intersection or difference of two polygons", Computers & Graphics, v. 13, n. 2, p. 167-183.

Preparata, F. e Shamos, M. (1989), "Computational Geometry: An Introduction", Springer-Verlag.

Queiroz, G. R. (2003), "Algoritmos Geométricos para Bancos de Dados Geográficos: da Teoria à Prática na Terralib", Tese de Mestrado, Inpe.

Rezende, P. J. e Stolfi, J. (1994), "Fundamentos de Geometria Computacional", IX Escola de Computação.

Rigaux, P., Scholl, M. e Voisard, A. (2002), "Spatial Databases with Applications to GIS", Morgan Kaufmann.

Rodrigues, V. L., Andrade, M. V. A., Queiroz, G. R. e Daltio, J. (2006), "Algoritmos exatos para interseção de segmentos e para sobreposição de mapas incorporados à biblioteca TerraLib", 32a Conferencia Latinoamericana de Informática, Santiago, Chile.

Rodrigues, V. L., Cavalier, A. S., Andrade, M. V. A. e Queiroz, G. R. (2005), "Algoritmos exatos para tratamento de mapas na Terralib", VII Simpósio Brasileiro de Geoinformática, Campos do Jordão, SP.

Steinberg, S. J. e Steinberg, S. L. (2006), "GIS : Geographic Information Systems for Social Sciences: Investigating Space and Place", SAGE.

TerraLib home page (2006), http://www.terralib.org.

TerraView home page (2006), http://www.dpi.inpe.br/terraview