

SB-index: Um Índice Espacial baseado em Bitmap para Data Warehouse Geográfico

Thiago Luís Lopes Siqueira^{1,2}, Ricardo Rodrigues Ciferri³, Valéria Cesário Times⁴

¹Programa de Pós-Graduação em Ciência da Computação
Universidade Federal de São Carlos (UFSCar)
Caixa Postal 676 – 13.565-905 – São Carlos – SP – Brasil

²Instituto Federal de Educação, Ciência e Tecnologia de São Paulo (IFSP) - Campus Salto

³Departamento de Computação – Universidade Federal de São Carlos (UFSCar)

⁴Centro de Informática – Universidade Federal de Pernambuco (UFPE)

prof.thiago@cefetsp.br, ricardo@dc.ufscar.br, vct@cin.ufpe.br

***Abstract.** In this paper, we propose an efficient index for geographic data warehouses: the SB-index, which enables multidimensional queries with spatial predicate and supports predefined spatial hierarchies. Performance evaluation results showed improvements which ranged from 25% to 99%.*

***Resumo.** Neste artigo, nós propomos o SB-index, um eficiente índice para data warehouse geográfico. Ele viabiliza consultas multidimensionais com predicados espaciais e provê suporte a hierarquias espaciais predefinidas. O melhoramento do desempenho, segundo nossos testes, variou de 25% a 99%.*

1. Introdução

Um *data warehouse* geográfico (DWG) é um banco de dados que permite a análise espacial aliada a consultas multidimensionais envolvendo enormes volumes de dados. As ferramentas SOLAP (*Spatial On-Line Analytical Processing*) promovem consultas sobre o DWG e auxiliam o suporte à decisão. Um DWG é construído sobre um banco de dados relacional usando tabelas de fatos e de dimensão, incluindo atributos espaciais que armazenam geometrias [Stefanovic et al. 2000] [Kimball e Ross 2002] [Fidalgo et al. 2004] [Malinowski e Zimányi 2008]. Nota-se a existência de hierarquias espaciais, e.g. região < nação < cidade < endereço, as quais viabilizam as operações SOLAP de *roll-up* e *drill-down* [Malinowski e Zimányi 2008]. Porém, o desempenho do processamento de consultas no DWG é drasticamente afetado pela existência de várias operações de junção entre tabelas e pela necessidade de computar um ou mais predicados espaciais. Logo, índices para diminuir o tempo de resposta das consultas são vitais no DWG.

Este artigo descreve um índice para DWG denominado SB-index (**S**patial **B**itmap **I**ndex) [Siqueira 2009]. O SB-index provê a execução eficiente de consultas analíticas multidimensionais com predicado espacial, bem como trata hierarquias espaciais predefinidas. Outra contribuição relevante deste artigo é a investigação dos efeitos da redundância de dados espaciais sobre o processamento de consultas no DWG. As seções 2 e 3 abordam o trabalho correlato e a fundamentação teórica, respectivamente. A Seção 4 detalha a proposta do SB-index, testado na Seção 5. A Seção 6 investiga o impacto da redundância de dados espaciais. A Seção 7 conclui o artigo.

2. Trabalhos correlatos

Os índices espaciais, como a R-tree [Guttman 1984], enfocam a computação eficiente de predicados espaciais [Gaede e Günther 1998], o que não é suficiente no DWG, onde também existem junções entre tabelas volumosas, e predicados convencionais e agrupamentos. Por outro lado, o índice Bitmap [Stockinger e Wu 2007] é alheio à multidimensionalidade, sendo empregado em data warehouses convencionais para indexar junções [O’Neil e Graefe 1995]. Não há registro do uso do índice Bitmap em DWG. A aR-tree [Papadias et al. 2001] é o único índice existente para DWG. Ela agrupa os objetos espaciais como a R-tree, estabelecendo uma **hierarquia espacial ad-hoc**. Cada entrada possui um retângulo envolvente mínimo (MBR) e o valor da função de agregação para todos os objetos incluídos no MBR. Porém, inúmeras aplicações de DWG utilizam **hierarquias espaciais predefinidas**, tais como região < nação < cidade < endereço. Os autores da aR-tree argumentam que ela não deve ser usada para indexar hierarquias predefinidas, apontando as visões materializadas [Harinarayan et al. 1996] [Stefanovic et al. 2000] como uma solução alternativa. Não existe, na literatura pesquisada, um índice para DWG que explore hierarquias predefinidas. Ao contrário da aR-tree, o nosso SB-index enfoca tais hierarquias.

3. Fundamentação Teórica

3.1. Data warehouse geográfico

Os esquemas de DWG da Figura 1 possuem a tabela de fatos *Lineorder* e as tabelas de dimensão *Part*, *Date*, *Supplier* e *Customer*. As tabelas de dimensão fornecem as perspectivas para análise do negócio tratado pela tabela de fatos, referenciando-a com chaves estrangeiras [Kimball e Ross 2002]. São atributos espaciais aqueles com sufixo *_geo*. Na Figura 1a, as tabelas de dimensão espaciais do esquema híbrido [Siqueira et al. 2008b] são *S_Address*, *C_Address*, *City*, *Nation* e *Region*, sendo que todas as geometrias armazenadas são distintas. Já na Figura 1b, as tabelas de dimensão espaciais são *Customer* e *Supplier*, forçando a existência de dados espaciais redundantes [Siqueira et al. 2008a, 2009]. A hierarquia espacial região < nação < cidade < endereço é válida em ambos os esquemas, tanto para a tabela *Supplier* quanto para *Customer*.

3.2. Índices

Seja X um atributo da relação R . Um **índice de projeção** [O’Neil e Graefe 1995] sobre X é uma seqüência de valores de X extraída de R e classificada pelo número da tupla. O **índice Bitmap** [Stockinger e Wu 2007] associa um vetor de bits a cada valor distinto v do atributo indexado X . Os vetores mantêm tantos bits quantas são as tuplas de R . Se na k -ésima tupla de R ocorre $X = v$, então o k -ésimo bit do vetor de bits associado a v tem valor 1. Caso contrário, o k -ésimo bit tem valor 0. Operações lógicas bit-a-bit são suficientes para resolver uma consulta multidimensional usando Bitmap. A cardinalidade do atributo, $|X|$, é o número de valores distintos que X pode assumir, determinando a quantidade de vetores de bits a serem criados. Constrói-se um **índice Bitmap de junção** [O’Neil e Graefe 1995] sobre o atributo C de uma tabela de dimensão para indicar o conjunto de tuplas da tabela de fatos que permite uma junção com um dado valor de C . Logo, os vetores de bits do índice Bitmap de junção indicam em quais tuplas da tabela de fatos ocorre cada um dos valores de C .

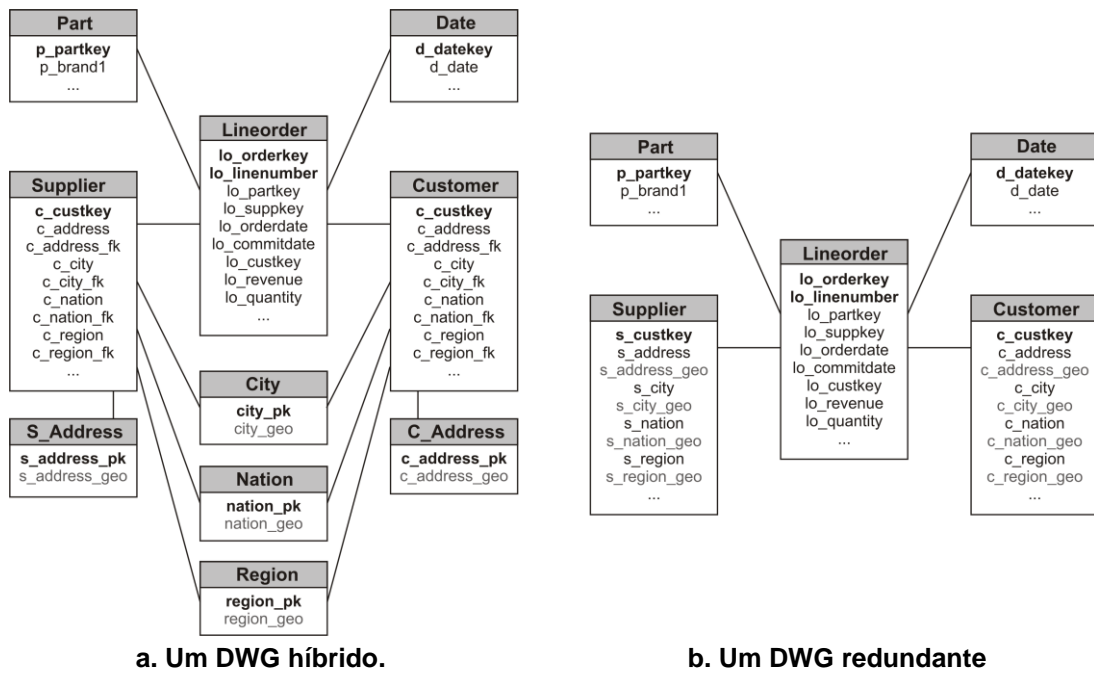


Figura 1. Dois esquemas de DWG.

4. SB-index

O SB-index consiste em uma adaptação do índice de projeção: além do valor da chave primária de uma tabela, cada entrada armazena também o MBR do objeto espacial correspondente. Definiu-se o tipo *sbitvector* (*spatial bit-vector*) como possuindo um número inteiro que representa a chave, e quatro números reais que representam o MBR. Por definição, o SB-index é um *array* do tipo *sbitvector*. Isto confere a cada entrada um tamanho em bytes dado por: $s = \text{sizeof}(\text{int}) + 4 \times \text{sizeof}(\text{double})$. Implementa-se o SB-index como um arquivo seqüencial, armazenando-o em disco. A Figura 2 ilustra a estrutura de dados do SB-index, cujas entradas têm um valor de chave primária da tabela de dimensão espacial City (Figura 1a), e o MBR do objeto espacial correspondente. Além disso, **B** é um índice Bitmap de junção sobre a chave primária da tabela de dimensão espacial. Por exemplo, vê-se que $city_pk = 1$ ocorre na 1ª e na 2ª tuplas da tabela de fatos. Obrigatoriamente, $SB\text{-index}[i]$ associa-se a $B[i]$. Desta forma, o MBR de $SB\text{-index}[0]$ também está retratado na 1ª e na 2ª tuplas da tabela de fatos. Requer-se a construção de um SB-index para cada nível da hierarquia espacial.

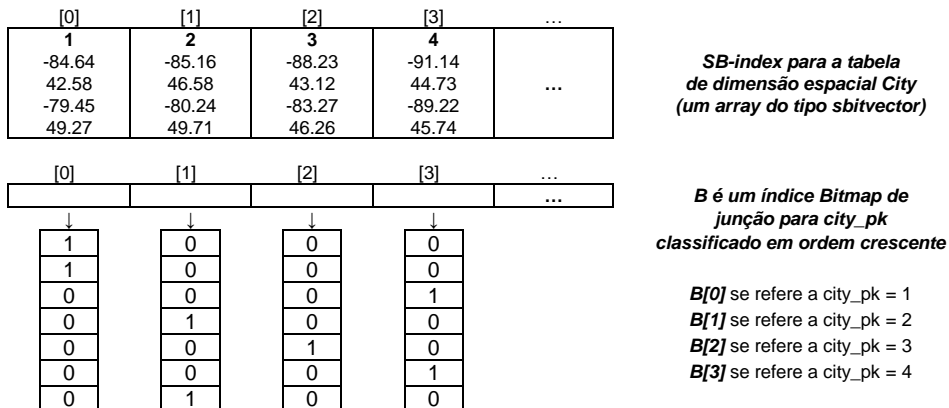


Figura 2. A estrutura de dados do SB-index.

As consultas SOLAP possuem predicados convencionais e espaciais. O processamento de consultas usando o SB-index pode ser resumido em três fases: (i) uma busca seqüencial no índice para verificar quais entradas satisfazem ao predicado espacial, colecionando as chaves das entradas que o satisfizerem; (ii) o refinamento dos candidatos junto ao SGBD e a transformação do predicado espacial em um predicado convencional usando os resultados do refinamento; e (iii) a reescrita da consulta contendo apenas predicados convencionais, e a sua subsequente avaliação junto ao índice Bitmap de junção. O Algoritmo 1 e a Tabela 1 detalham o processamento de consultas usando o SB-index.

Tabela 1. Parâmetros do Algoritmo SequentialSearch.

Parâmetro	Descrição
<i>idx</i>	arquivo do SB-index
<i>L</i>	n° máximo de entradas do tipo <i>sbitvector</i> que cabem em 1 página de disco
<i>R</i>	relacionamento espacial
<i>QW</i>	janela de consulta espacial <i>ad hoc</i>
<i>candidates</i>	seção da memória primária que mantém os valores das chaves primárias dos objetos espaciais candidatos a satisfazer o predicado espacial
<i>str_conventional_predicate</i>	<i>string</i> que descreve um predicado convencional
<i>T</i>	tabela de dimensão espacial
<i>pk</i>	chave primária de <i>T</i>
<i>sa</i>	atributo espacial de <i>T</i>
<i>query</i>	consulta reescrita, inicialmente sem o predicado espacial

Algoritmo 1: SequentialSearch(*idx*, *L*, *R*, *QW*, *candidates*,
 str_conventional_predicate, *T*, *pk*, *sa*, *query*, *answer*)

Saída: a resposta da consulta SOLAP

Declarações: *page*, *buffer*

```

1  Open (idx)
2  n ← 0
3  while not (eof(idx))
4    Read (idx, page)
5    Copy (page, buffer)
6    for i ← 0 to L do
7      if R(buffer[i], QW) then
8        candidates[n] ← buffer[i].pk
9        n ← n + 1
10   end-for
11  end-while
12  Close (idx)
13  for i ← 0 to n do
14    if (Execute_DBMS(select R(jc,sa) from T where pk=candidates[i]))
15      then Concatenate (str_conventional_predicate, candidates[i])
16  end-for
17  Concatenate (query, str_conventional_predicate)
18  answer ← ExecuteStarJoinBitmap (query)

```

O algoritmo *SequentialSearch* inicia abrindo o arquivo do SB-index e zerando o contador de candidatos (linhas 1 e 2). As linhas 3 até 11 mostram uma iteração que coleciona todas as possíveis respostas do predicado espacial, percorrendo todas as entradas do SB-index (busca seqüencial). Nesta iteração, lê-se uma página de disco do índice e copia-se a mesma em um buffer (linhas 4 e 5). Outra repetição percorre todas as entradas do buffer, testando se o MBR de cada entrada satisfaz o relacionamento espacial (linha 7). Caso o relacionamento seja verdadeiro, copia-se a chave primária da entrada para a coleção de candidatos (linha 8), e incrementa-se o contador de candidatos (linha 9). Após a leitura integral do índice, fecha-se o seu arquivo (linha 12). Então, ocorrerá o refinamento de cada uma das entradas da coleção de candidatos. Este refina-

mento também é uma iteração (linhas 13 a 16). Nela, o SGBD testa se é verdadeiro o relacionamento espacial entre a janela de consulta e o objeto espacial cuja chave está colecionada dentre os candidatos. Em caso verdadeiro, a chave é concatenada à string do predicado convencional. Depois de formada, a string do predicado convencional é concatenada à string da consulta (linha 17). Por fim, obtém-se o conjunto resposta da consulta contendo apenas predicados convencionais, executando-se a mesma por meio do acesso ao índice Bitmap de junção (linha 18). Observa-se que, embora o SB-index dependa do índice Bitmap de junção, ficando suscetível aos efeitos negativos da alta cardinalidade de um atributo, tais efeitos podem ser atenuados pelas técnicas de compressão, *binning* e codificação [Stockinger e Wu 2007].

5. Testes de Desempenho

O esquema híbrido mostrado na Figura 1a foi derivado do *Star Schema Benchmark* (SSB) [O’Neil, P. et al. 2009]. Nós incluímos dados espaciais reais do TIGER/Line (www.census.gov/geo/www/tiger), e geramos três DWG com os fatores de escala 2, 6 e 10. Todos possuem 5 regiões distintas, 5 nações por região, 10 cidades por nação e um número crescente de fatos. O número de endereços por cidade está vinculado ao fator de escala utilizado. A consulta Q2.3 do SSB foi adaptada para tornar-se um *roll-up* com predicado espacial. Conforme a Figura 3, substituiu-se o predicado sublinhado por um predicado espacial envolvendo uma janela de consulta espacial (QW). Quanto maior o nível de granularidade, maior a janela, que cobre uma fração específica do *extent*: 0,001% no nível endereço, 0,05% em cidade, 0,1% em nação e 1% em região. As janelas de um *roll-up* possuem o mesmo centróide. Porém, janelas de *roll-ups* distintos são disjuntas. Mais detalhes dos dados espaciais e das consultas podem ser encontrados em <http://gbd.dc.ufscar.br/papers/ctdsbc2010/Mapas&Janelas.pdf>.

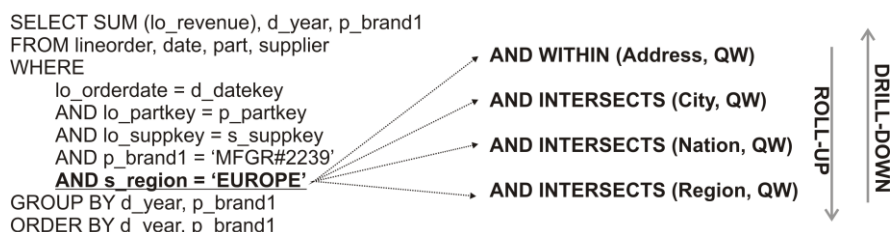


Figura 3. Consulta SOLAP para os testes de desempenho.

O SB-index foi implementado em C++ e compilado com gcc 4.1. Sua página de disco foi ajustada para 4 KB. Empregou-se o software FastBit (<https://sdm.lbl.gov/fastbit/>) para implementar os índices Bitmap de junção e manipulá-los para responder aos predicados convencionais. Usou-se a técnica de compressão WAH [Stockinger e Wu 2007] para atenuar os efeitos da alta cardinalidade. Os experimentos foram conduzidos em um computador com processador Pentium D de 2.8 GHz, 2 GB de memória primária, disco rígido SATA de 320 GB e 7200 RPM, sistema operacional Linux CentOS 5.2, PostgreSQL 8.2.5 e PostGIS 1.3.3. Foram executadas 5 operações de roll-up e calculadas as médias dos tempos decorridos para cada nível de granularidade. Comparou-se o desempenho do SB-index ao do sistema gerenciador de banco de dados (SGBD), o qual processa a consulta realizando a junção estrela e computando o predicado espacial. Para favorecer a computação deste último, foram criados índices espaciais (R-tree e GiST) sobre os atributos espaciais.

A **Figura 4a-c** exibe os resultados obtidos para os DWG híbridos com diferentes volumes de dados, mostrando que o SB-index superou notavelmente o desempenho do SGBD em todos os casos. A redução de tempo provida pelo SB-index esteve acima de 89%, independentemente do volume de dados. Isto foi válido inclusive no maior DWG, que possui 60 milhões de fatos e 100 mil fornecedores. Adicionalmente, a parcela da computação do predicado espacial, usando o SB-index, atingiu um máximo de 0,07% do custo total da consulta. Logo, o predicado convencional, computado por FastBit, consome mais tempo de processamento que o predicado espacial, no DWG híbrido. Ainda, a variação do volume de dados não degenerou o desempenho do SB-index. Por fim, o SB-index mais volumoso construído para os testes requereu 3,5 MB de armazenamento, contra 3,4 GB do índice Bitmap de junção (i.e., uma ínfima fração inferior a 1%).

6. Investigando o impacto da redundância de dados espaciais

Nenhum dos trabalhos pesquisados na literatura investigou a seguinte questão sob uma abordagem experimental: como a redundância de dados espaciais afeta o DWG? Esta questão é tratada nas duas seções seguintes.

6.1. Avaliação preliminar de desempenho

Os experimentos preliminares foram semelhantes aos descritos na Seção 5, exceto pela construção e uso do DWG redundante (Figura 1b) com fator de escala 10. O nível de granularidade Endereço não foi avaliado porque não é redundante. A redundância de dados espaciais afetou os requisitos de armazenamento: enquanto o DWG híbrido gerado com fator de escala 10 ocupou 15 GB, o exemplar redundante requereu 150 GB. Nota-se que o processamento de consultas usando o SGBD foi prejudicado. No nível Região, o processamento no DWG híbrido levou 2790,29s (**Figura 4c**), enquanto que no redundante foram gastos 6200,44s (**Figura 4d**), conduzindo a um aumento de 122%. Ressalta-se que neste nível há um alto grau de redundância, pois existem 5 regiões distintas para 1.000.000 registros na tabela *Supplier*.

O SB-index proveu reduções de tempo expressivas, mesmo sob redundância de dados espaciais: 90,20% no nível Cidade, 65,84% em Nação e 25,45% em Região. Embora o SB-index tenha proporcionado um bom desempenho, observa-se que a fatia de participação do mesmo, no tempo total da consulta, cresceu conforme a redundância aumentou: 12% no nível Cidade, 56% no nível Nação e 70% no nível Região. Logo, o SB-index passou a ser a etapa mais custosa no processamento da consulta nos níveis Nação e Região, superando o tempo do processamento do índice Bitmap de junção (realizado por FastBit). Este fato chamou-nos a atenção para aperfeiçoar o SB-index.

6.2. SB-index NR

Os resultados da Seção 6.1 indicaram que a redundância de dados espaciais degenerou o desempenho do SB-index. Isto ocorreu porque a busca seqüencial sobre o índice avaliava MBR repetidos, bem como a custosa etapa de refinamento testava objetos espaciais repetidos. Este fato motivou a proposição de uma melhoria sobre o SB-index. A idéia principal deste melhoramento é avaliar apenas MBR distintos, tal como faz o SB-index no DWG híbrido. Assim, criou-se o SB-index NR (*non-redundant* SB-index).

No SB-index NR, atribui-se uma lista a cada MBR distinto. Cada lista é armazenada em disco e mantém todos os valores de chaves associados a um MBR específico. No processamento de uma consulta SOLAP, testa-se o relacionamento espacial entre o MBR e a janela de consulta. Se o relacionamento existir, então apenas um valor de chave é conduzido ao refinamento. Então, se o objeto espacial identificado por esta chave for uma resposta da consulta, adicionam-se ao predicado convencional todas as chaves da lista correspondente. Por fim, a consulta reescrita com o predicado convencional é resolvida acessando-se o índice Bitmap de junção. O SB-index NR foi submetido aos mesmos testes da Seção 6.1. Conforme mostra **Figura 4d**, o SB-index NR proveu excelentes reduções de tempo comparadas ao SGBD: 91,73% no nível Cidade, 85,29% em Nação, 80,40% em Região. A manipulação de geometrias distintas, apenas, trouxe grandes benefícios ao SB-index NR.

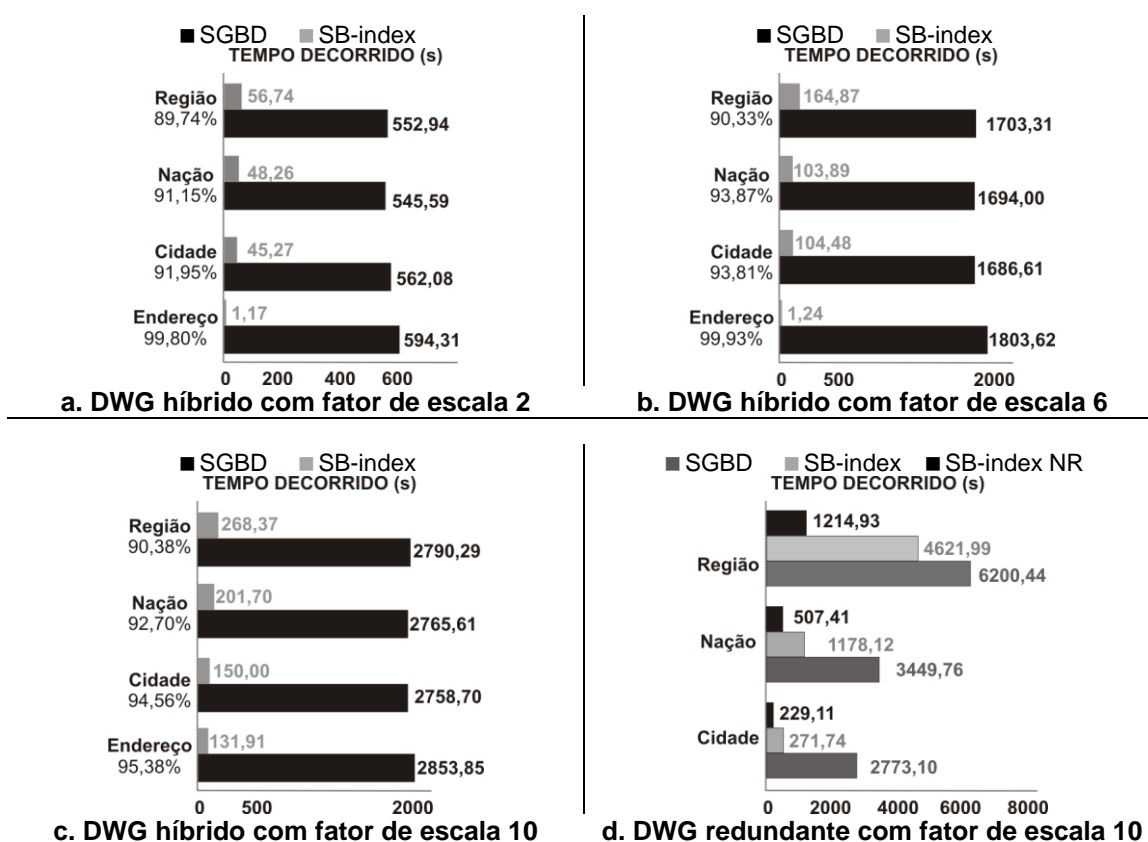


Figura 4. Resultados dos testes de desempenho.

7. Conclusões

Este artigo descreveu um eficiente índice para data warehouse geográfico (DWG) denominado SB-index, cujos diferenciais são: (i) permite consultas multidimensionais com predicados espaciais; (ii) provê suporte a hierarquias espaciais predefinidas; (iii) introduz o uso do índice Bitmap em DWG, herdando as suas vantagens (como as técnicas de codificação, *binning* e compressão); (iv) computa o predicado espacial e o transforma em um predicado convencional; e (v) requer uma quantidade reduzida de espaço para armazenamento. O SB-index corrobora a idéia de que um filtro espacial eficiente, seguido do acesso ao índice Bitmap de junção, aumenta significativamente o desempenho de consultas SOLAP. Outras contribuições do artigo foram: (vi) a investigação dos

efeitos da redundância de dados espaciais sobre o DWG; e (vii) a aplicação e a adaptação de estruturas de indexação para aperfeiçoar o desempenho de consultas em DWG redundantes. A redução de tempo provida pelo SB-index variou de 25% a 99%, em DWG com e sem redundância de dados espaciais. Este trabalho, em nível de Mestrado, produziu as publicações descritas em [Siqueira et al. 2008a, 2008b, 2009].

Agradecimentos

Os autores agradecem o apoio financeiro de: CAPES, FAPESP, CNPq e FINEP. O trabalho está inserido no Projeto Web-PIDE do Observatório da Educação (CAPES/INEP). O aluno agradece também a Profa. Dra. Cristina Dutra de Aguiar Ciferri por suas contribuições.

Referências

- Fidalgo, R. N. et al. (2004) GeoDWFrame: “A Framework for Guiding the Design of Geographical Dimensional Schemas”. In: 6th DaWak. p. 26-37.
- Guttman, A. (1984) “R-trees: a dynamic index structure for spatial searching”. SIGMOD'84, New York, NY, USA: ACM, pp. 47-57.
- Gaede, V.; Günther, O. (1998) “Multidimensional Access Methods”. In: *ACM Computing Surveys*, v.30, n.2, p.170-231.
- Harinarayan, V.; Rajaraman, A.; Ullman, J. D. (1996) “Implementing data cubes efficiently”. ACM SIGMOD Record, v.25, n.2, p.205-216.
- Kimball, R. and Ross, M. (2002) *The Data Warehouse Toolkit*. Wiley, 2a edição.
- Malinowski, E. and Zimányi, E. (2008) *Advanced Data Warehouse Design: from Conventional to Spatial and Temporal Applications*. Springer, 1ª Edição.
- Papadias, D. et al. (2001) “Efficient OLAP Operations in Spatial Data Warehouses”. In: 7th Symposium on Spatial and Temporal Databases. p. 443-459.
- O’Neil, P., Graefe, G. (1995) “Multi-Table Joins Through Bitmapped Join Indices”. In *ACM SIGMOD Record*, v.24, n.3, p.8-11.
- O’Neil, P., O’Neil, E., Chen, X, Revilak, S. (2009) “The Star Schema Benchmark and Augmented Fact Table Indexing”. TPCTC 2009, LNCS 5895 pp.237-252.
- Siqueira, T. L. L. (2009) “SB-index: um Índice Espacial Baseado em Bitmap para Data Warehouse Geográfico”. Dissertação. <http://gbd.dc.ufscar.br/download/files/Thiago.Dissertacao.pdf>
- Siqueira, T. L. L., Ciferri, C. D. A., Times, V. C., Oliveira, A.G., Ciferri, R. R. (2009) “The impact of spatial data redundancy on SOLAP query performance”. In: *Journal of the Brazilian Computer Society*. v.15, p.19-34.
- Siqueira, T. L. L., Ciferri, R. R., Times, V. C., Ciferri, C. D. A (2008a) “Investigating the Effects of Spatial Data Redundancy in Query Performance over Geographical Data Warehouses”. In: 10th GEOINFO.
- Siqueira, T. L. L., Ciferri, R. R., Times, V. C., Ciferri, C. D. A (2009b) “A Spatial Bitmap-Based Index for Geographical Data Warehouses”. In: 24th ACM SAC.
- Stefanovic, N.; Han, J.; Koperski, K. (2000) “Object-Based Selective Materialization for Efficient Implementation of Spatial Data Cubes”. IEEE TKDE v.12, n.6, p.938-958.
- Stockinger, K. and Wu, K. (2007) “Bitmap Indices for Data Warehouses”. In: *Data Warehouses and OLAP: Concepts, Architectures and Solutions*. IRM, p.157-178.