Jurnal Ilmu Komputer Agri-Informatika

Tersedia secara online di:
http://journal.ipb.ac.id/index.php.jika

Volume 1 Nomor 2 halaman 45-51
ISSN: 2089-6026

Peningkatan Kinerja Server Aplikasi Web GIS Berbasis PostgreSQL dan MapServer

Performance Improvement of Web GIS Application Server Based on PostgreSQL and MapServer

AURIZA RAHMAD AKBAR*, HARI AGUNG ADRIANTO

Abstrak

Aplikasi web GIS berbasis PHP MapScript dengan data geografis yang sangat besar sangatlah lama waktu *loading*-nya. Pada penelitian ini, kami melakukan *scale-up* server supaya bekerja secara optimal. Kami juga menggunakan teknik *caching* untuk mempercepat skrip PHP dan pembuatan gambar peta, masing-masing menggunakan Alternative PHP Cache (APC) dan TileCache. APC dapat digunakan sebagai *opcode cache* dan *data cache* untuk mempercepat aplikasi web berbasis PHP secara umum. Solusi TileCache menawarkan pembuatan gambar peta yang lebih cepat dan konsumsi sumber daya yang lebih sedikit.

Kata kunci: Mapserver, PostgreSQL, server, TileCache, web GIS

Abstract

The PHP MapScript-based web GIS application with a huge amount of geographic data is very slow to load. In this research, we scale-up the server to bring the real potential out of the hardware. We also use caching technique to speed-up both the PHP scripts and the map generation, using APC (Alternative PHP Cache) and TileCache respectively. APC can be used as opcode cache and data cache to speed-up PHP-based web application in general. TileCache solution offers faster map generation and lower resource consumption.

Keywords: mapserver, postgresql, server, tilecache, web GIS

PENDAHULUAN

Aplikasi web *Geographic Information System* (GIS) 'Scibun' yang dikembangkan oleh PT. Sucofindo untuk Kementerian Pertanian memiliki data geografis yang besar, yaitu hampir 600 MB dalam bentuk teks SQL. Data geografis tersebut merupakan data perkebunan kelapa sawit, karet, dan kakao di hampir seluruh provinsi di Indonesia. Dengan data sebesar itu, waktu yang diperlukan oleh Scibun untuk menggambar peta sangat lama, bisa lebih dari 10 detik.Oleh karena itu, perlu dilakukan konfigurasi untuk meningkatkan kinerja server. Scibun menggunakan PostgreSQL dengan ekstensi PostGIS untuk menyimpan data geografis dan MapServer (PHP MapScript) untuk menggambar peta.

Penelitian ini bertujuan meningkatkan kinerja server (*scale-up*) sehingga aplikasi web GIS Scibun dapat berjalan lebih cepat. Selain itu, akan dicoba alternatif lain yang tidak terlalu membebani prosesor, yaitu TileCache.*Scale-up* adalah teknik untuk meningkatkan kinerja server dengan menambah sumber daya. Hal ini mencakup konfigurasi server supaya sumber daya tersebut dapat dimanfaatkan secara optimal (Michael *et al.* 2007).

TileCache adalah implementasi WMS-C (*Web Map Service Caching*) *open source* yang dikembangkan oleh MetaCarta. TileCache merupakan skrip CGI Python yang dapat membuat

Departemen Ilmu Komputer, Fakultas Matematika dan Ilmu Pengetahuan Alam, Institut Pertanian Bogor, Bogor 16680
*Penulis korespondensi: Tel/Faks: 0251-8625584; Surel: auriza.akbar@gmail.com

cache dari server WMS apapun. Hasilnya kemudian dapat ditampilkan pada klien yang mendukung WMS-C, seperti OpenLayers. OpenLayers adalah pustaka JavaScript *open source* untuk menampilkan peta pada *web browser*. OpenLayers menyediakan antarmuka untuk membuat aplikasi web GIS yang interaktif seperti Google Maps.

METODE

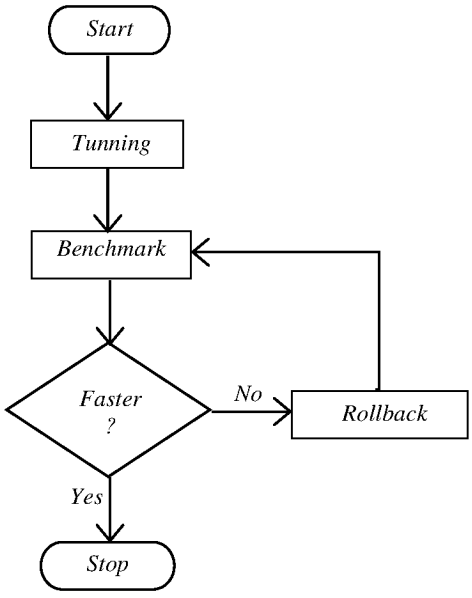
Aplikasi Web dan Data

Aplikasi web GIS Scibun digunakan untuk menguji kinerja server. Scibun ditulis dalam bahasa PHP dan menggunakan ekstensi PHP MapScript untuk menggambar peta.Data geografis yang digunakan adalah hasil survey perkebunan oleh PT. Sucofindo. Namun, halaman peta yang diuji hanya peta awal, yaitu peta administrasi yang terdiri atas tiga *layer* data geografis, yaitu provinsi, kabupaten, dan jalan. Hal ini dilakukan karena pada saat penelitian, data pada *layer* perkebunan belum lengkap seluruhnya.

Selain halaman peta, halaman awal (indeks) Scibun juga digunakan untuk menguji kinerja server. Hal ini dilakukan supaya tidak hanya halaman peta saja yang menjadi lebih cepat, tetapi juga untuk keseluruhan halaman aplikasi web.

Prosedur

Benchmark dilakukan secara sistematis, yaitu dengan mengkonfigurasi satu opsi dalam satu waktu, kemudian dibandingkan hasilnya. Hanya beberapa opsi yang dianggap mempengaruhi kinerja server saja yang diuji. Jika ternyata hasil konfigurasi untuk opsi tersebut meningkatkan kinerja server, konfigurasi tersebut akan tetap dipertahankan. Tetapi jika tidak, konfigurasi pada opsi tersebut dikembalikan seperti semula. Diagram alur untuk prosedur *benchmark* untuk satu iterasi dapat dilihat pada Gambar 1.



Gambar 1 Prosedur *benchmark*

Penjelasan dari langkah-langkah diagram alur pada Gambar 1 di atas adalah sebagai berikut:

- ***Tuning***: mengkonfigurasi salah satu opsi pada aplikasi server.
- ***Benchmark***: menjalankan program *benchmark* untuk mengukur kinerja server.
- ***Faster?***: membandingkan kinerja server setelah konfigurasi dengan yang sebelumnya. Kinerja server diukur dengan metrik *transaction per second*.

- ***Rollback***: jika ternyata hasil konfigurasi tidak meningkatkan kinerja server, maka konfigurasi opsi tersebut dikembalikan seperti semula dan kembali melakukan *tuning*.

Meskipun sering dikritik, *benchmark* adalah cara yang efektif dan terjangkau dalam menjalankan eksperimen. Intinya, *benchmark* adalah sebuah sampel dari suatu domain tugas; sampel ini kemudian dieksekusi oleh komputer. Pengukuran kinerja dilakukan selama proses eksekusi. Sebuah *benchmark* menyediakan medan yang setara untuk ide-ide yang saling berlawanan, dan jika benchmark tersebut cukup representatif, *benchmark* dapat menghasilkan perbandingan yang dapat diulang dan objektif. Paling tidak, *benchmark* dapat menyingkirkan pendekatan yang keliru dan klaim yang berlebihan (Tichy 1997).

Benchmark dilakukan pada setiap komponen aplikasi server. Urutan komponen server yang akan dikonfigurasi adalah PostgreSQL, Apache, PHP, dan MapServer. TileCache akan dicoba jika semua komponen di atas telah dikonfigurasi secara optimal.

Secara umum, program 'siege' digunakan untuk mensimulasikan pengguna yang mengakses server dalam waktu bersamaan (konkuren). *Benchmark* dilakukan pada tingkat konkurensi pengguna 4, 8, 16, 32, 64, 128, 256, dan 512. Jika *benchmark* pada tingkat konkurensi tertentu terjadi kegagalan akses lebih dari 1%, maka *benchmark*tidak dilanjutkan ke tingkat konkurensi yang lebih tinggi. Artinya, server tidak dapat melayani permintaan pada tingkat konkurensi tersebut dengan baik. Metrik yang dibandingkan dari hasil *benchmark* ini adalah nilai *transaction per second*.

Opsi *benchmark* dibedakan antara halaman peta dengan halaman indeks. *Benchmark* pada halaman peta hanya dilakukan sebanyak empat kali permintaan karena *loading* halaman ini cukup lama, sedangkan *benchmark* pada halaman indeks dilakukan selama 60 detik.

Benchmark dilakukan sebanyak tiga kali ulangan pada tiap tingkat konkurensi, kemudian diambil rataannya. Di antara tiap ulangan, diberikan jeda 30 detik agar jumlah proses anak server PostgreSQL dan Apache kembali normal. Server tidak di-*restart* supaya *cache* aplikasi web tetap ada. Jika *cache* hilang, kinerja server akan menurun dan mempengaruhi hasil *benchmark*.

HASIL DAN PEMBAHASAN

Lingkungan Pengujian

Spesifikasi server perlu disebutkan di sini sebagai pembanding bsgi yang ingin menguji ulang hasil penelitian ini. Server untuk aplikasi web GIS memiliki prosesor dengan arsitektur 64-bit (x86-64). Berikut adalah spesifikasi server ini.

- Nama: Extron NetSystem E400
- CPU : Intel Xeon E5606 Quad Core 2.13 GHz
- RAM : UDIMM DDR3 2 GB – 1333 ECC
- *Harddisk* : SATA 500 GB, 7200 rpm
- NIC : 2× Gigabit Ethernet

Kecepatan tulis dan baca *harddisk* pada server ini adalah 88 MB/s dan 107 MB/s.

Sistem operasi yang dipilih untuk server ini adalah Debian GNU/Linux 6.0. Debian merupakan distribusi GNU/Linux yang banyak digunakan sebagai server web (W3Techs 2011). Perangkat lunak server memakai distribusi 64-bit. Kinerja server Apache meningkat 39.2% dengan memakai perangkat lunak 64-bit (Tonkikh 2006). Server hanya diinstal perangkat lunak yang dibutuhkan. Aplikasi server yang dipakai pada server ini antara lain: PostgreSQL 8.4.7, PostGIS 1.5.1, Apache 2.2.16, PHP 5.3.3, MapServer 5.6.5, dan TileCache 2.03.

Nilai waktu eksekusi maksimum skrip PHP default adalah 30 detik. Karena permintaan data spasial yang berukuran besar, maka waktu 30 detik ini terkadang tidak cukup. Untuk mengatasinya, opsi waktu eksekusi maksimum ditambah menjadi 60 detik.

Konsumsi RAM server saat *idle* adalah 54 MB, sedangkan konsumsi RAM untuk satu proses saat melayani permintaan pada tingkat konkurensi 16 dapat dilihat pada Tabel 1.Sebelum dikonfigurasi, server hanya mampu melayani permintaan halaman peta dengan baik sampai tingkat konkurensi 16. Kinerja server menurun drastis pada tingkat konkurensi 32.

Tabel 1 Konsumsi RAM tiap proses

Akses halaman	PostgreSQL	Apache
Peta	50 MB	90 MB
Indeks	6 MB	16 MB

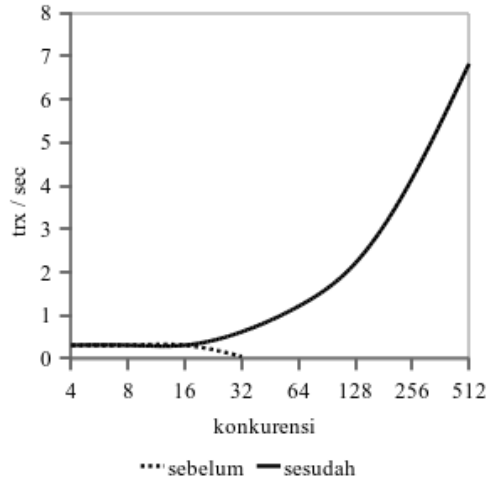
PostgreSQL

Konfigurasi server PostgreSQL terletak pada*/etc/postgresql/8.4/main/postgresql.conf*. Berikut adalah konfigurasi PostgreSQL yang perlu disesuaikan.

- *shared_buffers = 256MB*
- *effective_cache_size = 768MB*
- *max_connections = 32*

Konfigurasi PostgreSQL perlu dilakukan *scale-up* untuk memanfaatkan RAM pada server secara optimal. Jumlah RAM yang dialokasikan untuk PostgreSQL dan sistem operasi adalah 1024 MB. Nilai *shared_buffers* adalah 25%-nya, yaitu 256 MB. Nilai *effective_cache_size* adalah 75%-nya, yaitu 768 MB (PostgreSQL 2009). Nilai *max_connections* didapat dari jumlah alokasi RAM untuk PostgreSQL dibagi dengan rataan konsumsi RAM tiap proses PostgreSQL (Temme 2007). Jika tidak terjadi *swapping*, nilai ini dapat dinaikkan lagi.

Hasil konfigurasi dapat dilihat pada Gambar 2. Kinerja server secara umum sama seperti kondisi awal. Akan tetapi, server sekarang mampu melayani permintaan halaman peta dengan baik pada tingkat konkurensi yang lebih tinggi.



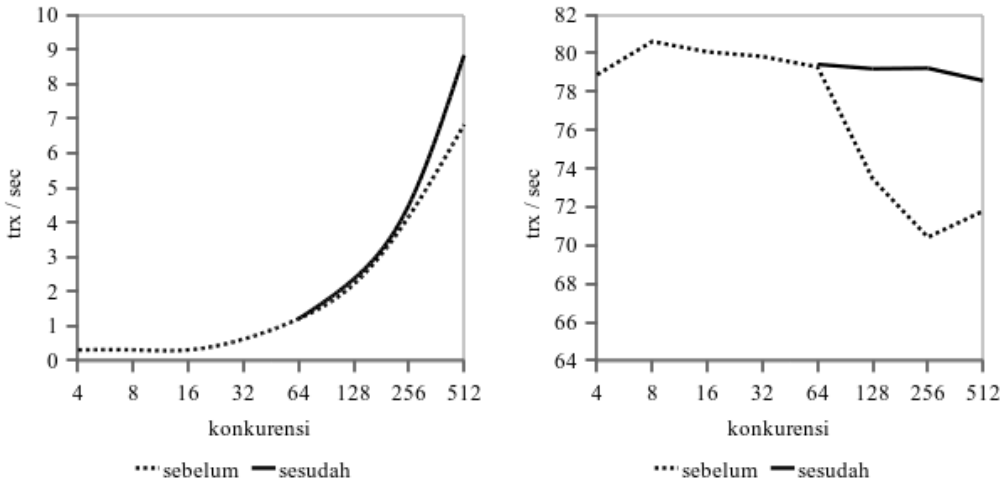
Gambar 2 Kinerja server sebelum dan sesudah *scale-up* PostgreSQL untuk halaman peta

Apache

Konfigurasi utama server web Apache terletak pada */etc/apache2/apache2.conf*. Berikut adalah konfigurasi Apache yang perlu disesuaikan.

- *MaxClients 64*

Jumlah klien maksimum ini didapat dari alokasi RAM untuk Apache dibagi dengan dengan konsumsi RAM tiap proses Apache (Temme 2007). Hasil perhitungan untuk jumlah klien maksimum adalah sebagai berikut: 1024 MB / 16 MB = 64.Kinerja server Apache secara keseluruhan meningkat setelah dilakukan konfigurasi di atas. Perbandingan kinerja server Apache untuk halaman peta dan indeks dapat dilihat masing-masing pada Gambar 3.



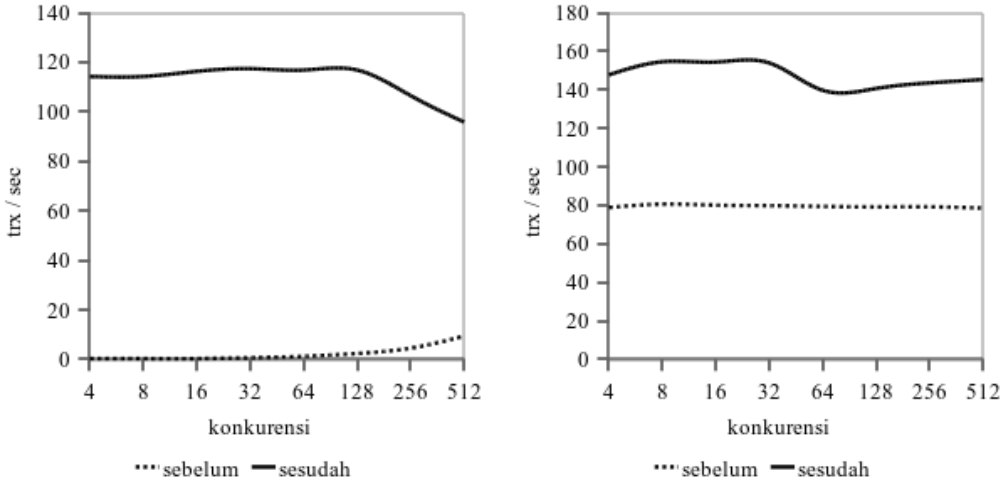
Gambar 3 Kinerja server setelah konfigurasi Apache untuk halaman peta (kiri) dan halaman indeks (kanan)

PHP

Salah satu implementasi *opcode cache* untuk PHP adalah APC (*Alternative PHP Cache*). APC akan menyimpan *opcode* hasil kompilasi ke dalam *shared memory* secara otomatis. Kinerja aplikasi web PHP semakin meningkat karena *opcode* disimpan di dalam RAM.APC juga dapat digunakan sebagai *data cache*. Tujuannya adalah untuk mengurangi koneksi ke *database*. Data langsung diambil dari *cache*sehingga kinerja server meningkat.

Query untuk mengambil data dari *database* kebanyakan memakai parameter untuk memilih data yang diinginkan. Parameter tersebut dipakai dalam klausa WHERE pada *query*. Hasil dari *query* ini dapat disimpan dalam *cache* dengan teknik *data cache* parametrik. Nama parameter ditambahkan pada nama *cache* APC untuk membedakannya dari *cache* yang berjenis sama tetapi dengan parameter yang berbeda.

Pemakaian APC sebagai *opcode* dan *data cache* dapat meningkatkan kinerja aplikasi web PHP sampai 100 kali lipat lebih. Perbandingan kinerja server sebelum dan sesudah konfigurasi PHP untuk halaman peta dan indeks dapat dilihat masing-masing pada Gambar 4. Akan tetapi peningkatan kinerja ini hanya berlaku untuk halaman peta awal saja, yaitu peta administrasi. Teknik *caching* peta lainnya perlu dicari untuk mengatasi kekurangan tersebut.



Gambar 4 Kinerja server setelah konfigurasi PHP untuk halaman peta (kiri) dan halaman indeks (kanan)

MapServer

Prosedur *benchmark* MapServer berbeda dari prosedur sebelumnya. *Benchmark* MapServer menggunakan program 'shp2img' untuk menguji kinerja konfigurasi *mapfile*. Program ini dijalankan dalam mode *debug* sebanyak lima kali ulangan.

Opsi *MAXSCALEDENOM* digunakan untuk mengimplementasikan LOD (*level of detail*) pada *mapfile*. Selain meningkatkan kinerja, LOD juga berfungsi untuk mengurangi kerumitan pada tampilan peta. Opsi ini dapat digunakan untuk objek *layer* maupun *class*.

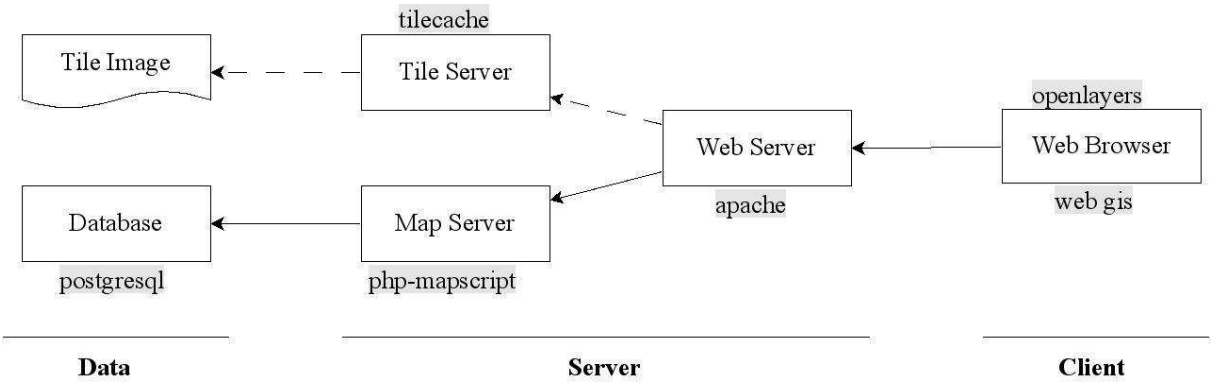
GiST (*Generalized Search Tree*) menyediakan indeks spasial untuk kolom bertipe geometris pada PostGIS. Indeks GiST bermanfaat untuk mempercepat pencarian data spasial (PostGIS 2010). *Benchmark* pada *mapfile* dilakukan pada *extent* peta tertentu. Hal ini dilakukan agar pengaruh indeks GiST terlihat. Jika seluruh peta yang digambar, maka indeks tidak akan dipakai.

Implementasi LOD mampu mengurangi waktu yang dibutuhkan MapServer untuk menggambar peta lebih dari 50%. MapServer yang sebelumnya membutuhkan waktu 10.5 detik untuk menggambar peta, sekarang hanya membutuhkan waktu 4.9 detik. Penggunaan indeks GiST juga mempercepat *loading* halaman peta saat di-*zoom* pada *extent* tertentu.

TileCache

Pada pembahasan sebelumnya, metode *caching* peta dengan APC masih memiliki kekurangan yang mendasar, yaitu tidak semua peta dapat dibuat *cache*-nya. TileCache adalah alternatif *caching* peta yang akan digunakan untuk mengatasi kekurangan tersebut.

Implementasi TileCache memerlukan arsitektur web GIS yang baru. Arsitektur lama dengan PHP MapScript sudah sulit untuk ditingkatkan lagi kinerjanya. Arsitektur web GIS ditunjukkan pada Gambar 5. Garis putus-putus menunjukkan arsitektur TileCache yang baru.

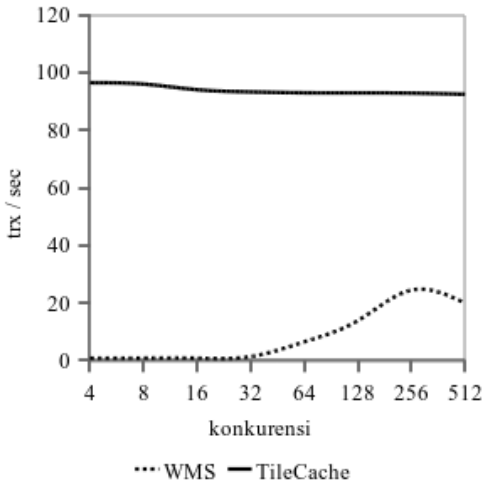


Gambar 5 Perbandingan arsitektur web GIS

Jika TileCache sudah berjalan, maka semua *tile* peta yang pernah diminta oleh klien akan dibuat *cache*-nya. Permintaan dari klien selanjutnya akan diambil dari *cache tile* peta. Jika *tile* peta yang dimaksud tidak ada dalam *cache*, maka TileCache akan meminta *tile* peta ke server WMS, dan kemudian hasilnya disimpan ke dalam *cache*.

Benchmark TileCache dilakukan dengan dua cara. Cara pertama: dengan memakai ekstensi 'Firebug' untuk Firefox. Metrik yang dibandingkan adalah waktu *loading* halaman peta pada OpenLayers. Program 'siege' tidak dapat dipakai karena OpenLayers menggunakan Ajax untuk meminta *tile* peta dari serversehingga *output* program 'siege' menjadi tidak valid.Cara kedua: program 'siege' akan dipakai dalam *benchmark* untuk satu permintaan *tile* peta langsung ke server WMS MapServer dan TileCache. Metrik yang dibandingkan adalah nilai *transaction per second*.

Waktu yang diperlukan OpenLayers untuk menampilkan peta dari server WMS MapServer dan TileCache masing-masing adalah 5.04 detik dan 0.66 detik. TileCache mengurangi waktu *loading* peta sebesar 87%. Perbandingan kinerja TileCache dengan server WMS MapServer dalam melayani permintaan satu *tile* peta ditunjukkan pada Gambar 6. TileCache unggul 100 kali lipat dari server WMS pada tingkat konkurensi di bawah 32.



Gambar 6 Perbandingan kinerja server WMS MapServer dengan server WMS-C TileCache

SIMPULAN

Kinerja server aplikasi web GIS berbasis PostgreSQL dan PHP MapScript dapat ditingkatkan dengan langkah-langkah berikut:

- *scale-up* server PostgreSQL dan Apache sesuai dengan jumlah RAM server,
- instalasi APC sebagai *opcode* dan *data cache* bagi PHP,
- implementasi LOD (*level of detail*) pada *mapfile*, dan
- penambahan indeks GiST pada kolom geometri PostGIS.

Jika setelah mengikuti langkah-langkah di atas kinerja server masih kurang memuaskan, maka TileCache dapat digunakan sebagai alternatif. TileCache mampu meningkatkan kinerja server WMS hingga 100 kali lipat. Akan tetapi, implementasi TileCache memerlukan arsitektur yang berbedasehingga tidak bisa langsung ditambahkan ke aplikasi web GIS lama.

DAFTAR PUSTAKA

Michael M, Moreira JE, Shiloach D, Wisniewski RW. 2007. Scale-up x scale-out: a case study using Nutch/Lucene. Di dalam: *21st International Parallel and Distributed Processing Symposium, IPDPS 2007*; Long Beach(US), 2007 Mar 26–30. California (US): IEEE International. hlm 1–8.

[PostgreSQL] PostgreSQL Global Development Group. 2009. *PostgreSQL 8.4.6 Documentation* [Internet]. [diunduh 2011 Jun 29]. Tersedia pada: http://www.postgresql.org/files/documentation/pdf/8.4/postgresql-8.4.6-A4.pdf.

Temme S. 2007. Apache performance tuning part one: scaling up. Di dalam: *ApacheCon US 2007*; Atlanta(US), 27 Nov 12–17. Georgia (US): Apache Software Foundation. hlm 1-20.

Tichy WF. 1997. Should computer scientists experiment more?. *IEEE Computer* 31(5):32–40. doi:10.1.1.36.5786.

Tonkikh A. 2006. Benchmarks: AMD64 in 32bit mode vs 64bit mode [Internet]. [diunduh 2011 Jun 29]. Tersedia pada: http://art-blog.no-ip.info/files/amd64vsi386.odt.

[W3Techs] World Wide Web Technology Surveys. 2011. Usage statistics and market share of Debian for websites [Internet]. [diunduh 2011 Jun 29]. Tersedia pada: http://w3techs.com/technologies/details/os-debian/all/all.