PERBANDINGAN PERFORMANSI DYNAMIC INDEXING DAN STATIC INDEXING (STUDI KASUS: PENGAMBILAN DATA DENGAN MENGGUNAKAN MOBILE CRAWLER)

Ammalia Pratiwi[1], Yanuar Firdaus A.w.[2], Arie Ardiyanti Suryani[3]

[1]Teknik Informatika, Fakultas Teknik Informatika, Universitas Telkom

Abstrak

Search Engine merupakan alat yang sangat berguna seiring dengan perkembangan World Wide Web (www). Adanya perubahan terhadap dokumen web yang merupakan sumber informasi bagi search engine, mengharuskan search engine secara rutin memperbaharui index file yang dimilikinya. Oleh karena itu diperlukan suatu metode pengindeksan yang efektif untuk menjawab permasalahan tersebut.

Tugas akhir ini mengkaji perbandingan metode pengindeksan antara static indexing dan dynamic indexing. Static indexing merupakan pengindeksan secara one-pass proses dan dynamic indexing merupakan pengindeksan yang memerlukan penggabungan dari dua buah index file.

Pengambilan data untuk kedua metode indexing yang diimplementasikan dilakukan oleh web crawler yang memanfaatkan peran dari mobile agent.

Hasil pengujian menunjukkan bahwa untuk kasus pengindeksan kembali, dynamic indexing memiliki performansi yang lebih baik jika dilihat dari sisi penggunaan waktu. Sedangkan static indexing memiliki performansi yang lebih baik jika dilihat dari sisi penggunaan memory. Metode ini juga baik digunakan untuk pembangungan awal suatu index file karena memiliki performansi waktu yang lebih baik.

Kata Kunci : indexing, dynamic indexing, website, crawler, mobile agent

Abstract

Search engine is a very useful media along with World Wide Web development. A changing situation of web document which is an information source to a search engine makes it routinely update its index files. Therefore, it is needed an effective indexing method to solve this problem.

This Final project explains about indexing methods comparison between static indexing and dynamic indexing. Static indexing is a one-pass process indexing while dynamic indexing is an indexing which needs combination from two file indexes. Data gainings for both implemented indexing methods done by web crawler which is using the role of mobile agent.

Testing results show that in reindexing update, dynamic indexing has better performance from time using, while static indexing has better performance from memory using. This method is also good for early development to an index file because it yields better time performance.

Keywords : indexing, dynamic indexing, website, crawler, mobile agent

Bab I
Pendahuluan

1.1 Latar Belakang

Mesin pencari atau *Search Engine* merupakan alat yang sangat berguna seiring dengan perkembangan *World Wide Web* (WWW) sebagai salah satu sumber informasi yang mudah diakses. Dokumen akan di telusuri dan di proses oleh komponen yang dimiliki *search engine* dengan tujuan membentuk *database index* yang mempermudah pengguna *search engine* untuk mencari informasi yang diinginkan. Komponen utama yang dimiliki oleh *search engine* untuk mendukung layanan yang dimilikinya, yaitu: *web crawler, indexing system,* dan *searching system*[3].

Web Crawler bertugas untuk menjelajahi dan mengumpulkan semua informasi yang ada di dalam halaman web. *Mobile crawler* adalah *crawler* yang memanfaatkan peran *mobile agent* dalam pelaksanaan tugasnya. *Mobile agent* akan bermigrasi ke sumber informasi, membawa keseluruhan kode program dan melakukan operasi penjelajahan halaman web dan pengunduhan terhadap informasi secara lokal pada *remote server* [4].

Selanjutnya, halaman yang telah didapatkan oleh *crawler* dianalisa oleh *indexing system* dengan cara mengindeks setiap kemungkinan *term* yang terdapat didalamnya dan disimpan ke dalam *database index*. Sejauh ini, kita mengasumsikan bahwa setiap dokumen halaman web adalah dokumen yang bersifat statis, dalam artian bahwa tidak terdapat perubahan terhadap isinya. Tetapi dalam kenyataannya, terkadang suatu halaman web juga terjadi penambahan dokumen, penghapusan dokumen, ataupun pembaharuan. Sehingga akan terjadi juga penambahan, penghapusan ataupun pembaharuan istilah atau *term* yang ada pada daftar list indeks. Cara yang paling mudah untuk menyelesaikan masalah ini adalah dengan merekonstruksi indeks secara berkala.

Dalam tugas akhir ini, akan dikaji metode pengindeksan pada mesin pencari yang memanfaatkan peran dari *mobile crawler* dalam pengambilan datanya. Metode pengindeksan yang akan dikaji adalah metode pengindeksan statis dan metode pengindeksan dinamis :

1. *Static indexing* merupakan metode pengindeksan secara *one-pass* proses, dimana indeks harus dibangun kembali dari awal jika terdapat perubahan.
2. *Dynamic indexing* merupakan metode pengindeksan yang memerlukan penggabungan dari dua buah *index file*[10].

Kedua metode tersebut akan dilihat perbandingan performansinya, dalam kasus ini adalah waktu dan memory yang digunakan, berdasarkan jumlah data yang telah diambil oleh *mobile crawler*. Perbandingan ini dilakukan untuk mendapatkan solusi terbaik, dalam hal performansi, terhadap proses pengindeksan dokumen yang sering terjadi perubahan pada isinya.

1.2 Perumusan Masalah

Perumusan masalah dari tugas akhir ini berdasarkan latar belakang masalah yang dipaparkan di atas adalah:

a. Bagaimana mengimplementasikan *web crawler* dengan menggunakan *mobile agent*
b. Bagaimana mengimplementasikan metode *dynamic indexing* dan *Static Indexing* pada system yang akan dibangun.
c. Bagaimana performansi *dynamic indexing* dan *static indexing* dengan parameter waktu dan penggunaan memory jika dibandingkan berdasarkan jumlah dokumen.

Adapun batasan masalah dalam tugas akhir ini adalah sebagai berikut:

1. Kemampuan *mobile crawler* dalam melakukan pengunduhan dokumen hanya mencapai 500 dokumen.
2. Dokumen yang diunduh merupakan dokumen *web* dengan tipe HTML.
3. Data yang digunakan adalah dokumen berita berbahasa Inggris.
4. *Term Indexing* yang dilakukan hanya terdiri dari *Tokenization*, *Stoplist* dan *Stemming*.

1.3 Tujuan

Hal-hal yang ingin dicapai dalam tugas akhir ini yaitu :

1. Merancang dan membangun *web crawler* yang menggunakan *mobile agent*.
2. Mengimplementasikan metode *static indexing* dan *dynamic indexing* untuk mengolah data yang didapatkan oleh *mobile crawler*
3. Menganalisis dan membandingkan hasil dari pengimplementasian metode pengindeksan dengan mengukur performansi dari masing-masing metode konstruksi pengindeksan, dalam hal ini parameter waktu dan penggunaan memory, berdasarkan jumlah dokumen yang terambil oleh *mobile crawler*.

1.4 Metodologi Penyelesaian Masalah

Metode yang digunakan dalam penyelesaian tugas akhir ini adalah:

1. Studi Literatur dengan mempelajari literatur-literatur yang relevan dengan permasalahan yang meliputi studi pustaka dan referensi tentang:
   a. *Web crawler*
   b. *Mobile agent*
   c. *Static Indexing* dan *Dynamic Indexing*
   d. Dokumen *web*
2. Melakukan perancangan perangkat lunak dengan menggunakan analisis kebutuhan sistem.
3. Melakukan implementasi berdasarkan analisis dan desain sistem yang telah dibuat.
4. Melakukan pengujian perangkat lunak yang telah dibangun
5. Melakukan analisis performansi *mobile crawler* yang mengimplementasikan *static indexing* dan *dynamic indexing*.
6. Penyusunan laporan tugas akhir dan kesimpulan akhir.

1.5 Sistematika Penulisan

Struktur Pembahasan Tugas Akhir ini disusun sebagai berikut :

BAB I PENDAHULUAN

Bab ini berisi latar belakang masalah, perumusan masalah, batasan masalah, tujuan pembahasan, metodologi penyelesaian masalah dan sistematika penulisan.

BAB II LANDASAN TEORI

Membahas dasar teori yang berhubungan dengan pengertian mengenai *World Wide Web (WWW)*, mesin pencari, *web crawler, indexing, dynamic indexing , mobile agent.*

BAB III PERANCANGAN PERANGKAT LUNAK

Bab ini akan membahas proses perancangan *web crawler* dengan memanfaatkan *mobile agent* yang mengimplementasikan *dynamic indexing* dan *static indexing*

BAB IV IMPLEMENTASI DAN ANALISIS HASIL UJI COBA

Membahas tentang analisis dari hasil pengujian ataupun percobaan pada implementasi *web crawler* dengan memanfaatkan *mobile agent* yang mengimplementasikan *dynamic indexing* dan *static indexing*

BAB V KESIMPULAN & SARAN

Pada bab ini akan menjelaskan kesimpulan dan saran sebagai hasil dari analisa dan implementasi Tugas Akhir.

Bab IV
Implementasi dan Pengujian Sistem

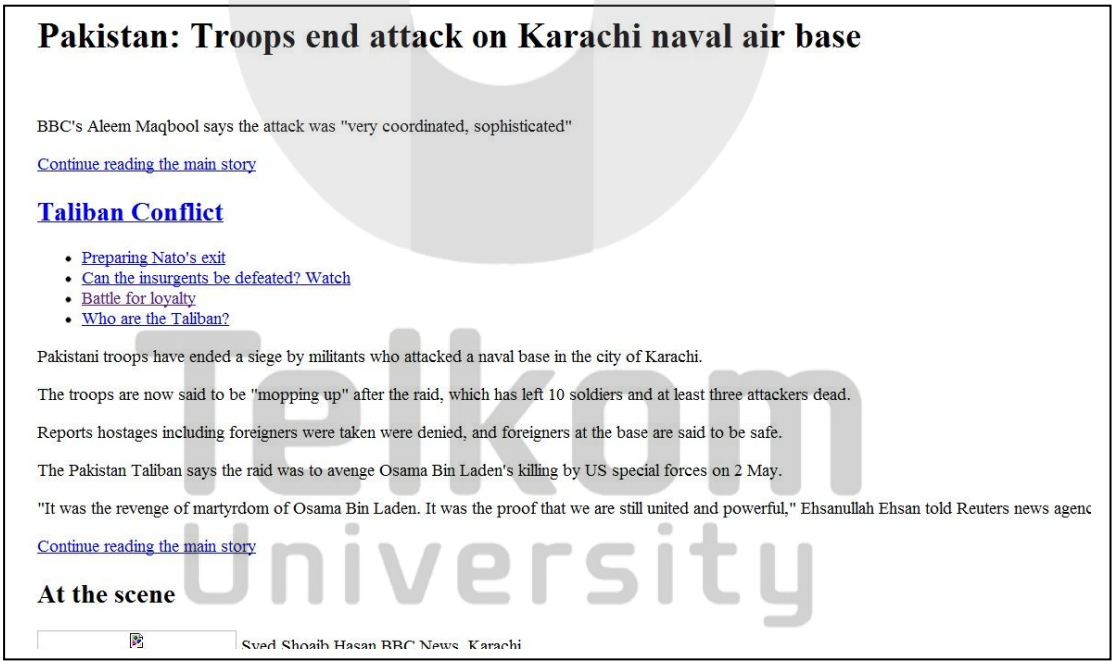
4.1 Implementasi Sistem

Tugas akhir ini membangun perangkat lunak yang memiliki fungsi utama untuk melakukan proses pembangunan index yang menggunakan dua buah metode, yaitu *static indexing* dan *dynamic indexing*. Inputan dari perangkat lunak ini adalah dokumen yang memiliki format *.html. Hasil dari sistem ini berupa index yang tersimpan dalan suatu database.

4.1.1 Inputan Sistem

a. Untuk *Crawler*
Inputan terhadap sistem adalah seed URL, yaitu alamat dari suatu halaman web dimana dalam tugas akhir ini *website* yang akan dikunjungi adalah salinan dari *http://www.bbc.co.uk*. Sebelumnya, dokumen web ini diunduh dengan menggunakan *tools Teleport Pro Website Copier* yang merupakan *tools* untuk *offline browsing* maupun *site mirroring.* Karakteristik dokumen yang menjadi inputan sistem adalah dokumen web berita berbahasa Inggris dengan format *.htm

Contoh dokumen web inputan dari sistem dapat dilihat pada gambar 4-1.



Pakistan: Troops end attack on Karachi naval air base

BBC's Aleem Maqbool says the attack was "very coordinated, sophisticated"

Continue reading the main story

Taliban Conflict

- Preparing Nato's exit
- Can the insurgents be defeated? Watch
- Battle for loyalty
- Who are the Taliban?

Pakistani troops have ended a siege by militants who attacked a naval base in the city of Karachi.

The troops are now said to be "mopping up" after the raid, which has left 10 soldiers and at least three attackers dead.

Reports hostages including foreigners were taken were denied, and foreigners at the base are said to be safe.

The Pakistan Taliban says the raid was to avenge Osama Bin Laden's killing by US special forces on 2 May.

"It was the revenge of martyrdom of Osama Bin Laden. It was the proof that we are still united and powerful," Ehsanullah Ehsan told Reuters news agenc

Continue reading the main story

At the scene

Syed Shoaib Hasan BBC News, Karachi

Gambar 4-1 : Contoh dokumen web inputan crawler

b. Untuk *Indexing*
Inputan terhadap sistem adalah dataset yang berupa dokumen *web* berita Bahasa Inggris yang merupakan hasil dari proses *crawling.* Inputan ini berlaku bagi kedua buah metode pengindeksan.

4.1.2 Outputan Sistem

a. Untuk Sistem *Crawler*
Outputan dari sistem ini adalah sebuah *database* yang berisi halaman yang berhasil diunduh pada proses *crawling*, *database* ini memiliki tiga buah *field*:
   1. id, menunjukkan identitas dari alamat URL.
   2. url, berisi alamat URL yang berhasil diunduh.
   3. html, berisi halaman html dari URL yang berhasil diunduh.

   Sebuah *database* yang berisi ukuran dari halaman tersebut, *database* ini memiliki tiga buah *field*:
   1. id, menunjukkan identitas dari alamat URL.
   2. url, berisi alamat URL yang berhasil diunduh.
   3. size, berisi ukuran dari alamat URL yang berhasil diunduh dalam satuan kilobytes.

b. Untuk Sistem *Indexing* dengan metode *Static Indexing*
Outputan dari sistem ini adalah sebuah *database* yang berisi data term, *database* ini memiliki tiga buah *field*:
   1. id, menunjukkan identitas dari term.
   2. url, menunjukkan alamat URL asal term.
   3. term, berisi term yang terbentuk dari proses *indexing*.

c. Untuk Sistem *Indexing* dengan metode *Dynamic Indexing*
Outputan dari sistem ini adalah sebuah *database* yang berisi data term, *database* ini memiliki tiga buah *field*:
   1. id, menunjukkan identitas dari term.
   2. url, menunjukkan alamat URL asal term.
   3. term, berisi term yang terbentuk dari proses *indexing*.

   Serta sebuah temporary index Z_0 yang berisi term dan alamat URL asal term. Ukuran dari temporary index ini belum memenuhi standar maksimalnya, sehingga belum dapat tersimpan ke dalam *database*

4.2 Skenario Pengujian Sistem

Setelah tahap implementasi, untuk mencapai tujuan dari tugas akhir ini, diperlukan pengujian sistem dan analisis terhadap hasil pengujian. Pengujian sistem menggunakan dataset yang telah dijelaskan sebelumnya lalu diuji dan dianalisis hasilnya.

Analisis performansi dari kedua buah metode pengindeksan dalam tugas akhir ini diukur dari dua buah parameter, yaitu :

a. Berdasaran waktu proses. Analisis dilakukan terhadap lamanya sistem dalam melakukan prose indexing terhadap sekumpulan dokumen web.
b. Berdasarkan memory. Analisis dilakukan terhadap penggunaan memory pada saat sistem sedang melakukan proses pembangunan indeks.

Berikut adalah skenario pengujian terhadap sistem yang dibangun pada tugas akhir ini :

1. **Crawler**
   a. Pengujian dilakukan dengan menggunakan dokumen *web* yang berasal dari *http://www.bbc.co.uk*
   b. Sistem Crawler yang mengimplementasikan Aglet sebagai *mobile agent* akan berjalan dengan input seed URL dari direktori tempat disimpannya dokumen web yang telah diunduh sebelumnya, yaitu pada alamat *http://192.168.1.9/www.bbc.co.uk/*
2. **Indexing**
   Proses *Indexing* akan dilakukan pada *client site* dengan menggunakan data *output* dari *mobile crawler*. Proses pengujian terhadap parameter performansi, dalam hal ini waktu dan *memory*, dilakukan sebanyak 3 kali lalu dihitung hasil performansi rata-rata dari setiap pengujian yang telah dilakukan. Pengujian ini berdasarkan kepada :
   a. Jumlah Dokumen
      Pengujian dilakukan dengan batasan jumlah dokumen yang bervariasi, mulai dari 5, 10, 25, 50, 100, 125, 250 hingga 500 dokumen. Pemilihan variasi jumlah dokumen pada pengujian ini berdasarkan presentase perubahan dokumen. Presentase perubahan dokumen dengan variasi seperti yang disebutkan sudah cukup mewakili jumlah dokumen yang berubah pada suatu web berita dan terbaca oleh pengguna internet. Serta dengan variasi pemilihan dokumen ini sudah dapat terlihat kinerja program secara keseluruhan.
   b. Jumlah Revisit
      Pengujian dilakukan dengan mengunjungi kembali website yang telah berhasil dijelajahi dan diunduh dengan tujuan untuk memperbaharui *index file* dan untuk mengetahui pengaruh perubahan dokumen terhadap sistem yang telah dibangun. Pengunjungan kembali ini dilakukan sebanyak tiga kali,

   Selain itu untuk sistem yang mengimplementasikan *dynamic indexing*, akan dilakukan pengujian tambahan, yaitu berdasarkan kepada:
   c. Ukuran temporary index Z_0
      Setiap pengujian dengan jumlah dokumen yang sama dilakukan sebanyak dua kali dengan mengubah ukuran temporary index Z_0. Batasan ukuran temporary index Z_0 adalah 100 dan 1000 term. Ukuran yang telah ditentukan ini merupakan asumsi yang ditujukan untuk mengetahui pengaruh ukuran temporary index terhadap proses pengindeksan dinamis secara keseluruhan.

   Dari pengujian-pengujian yang telah dilakukan akan didapat waktu proses dan memory yang digunakan pada saat proses pengindeksan baik menggunakan metode static indexing maupun dynamic indexing. Hasil dari tiap proses inilah yang digunakan sebagai materi analisis hasil pengujian.

4.3 Analisis Hasil Pengujian

Pengujian yang dilakukan pada tugas akhir ini berdasarkan atas dua buah parameter, yaitu :

1. Waktu
2. Memori
   Memori yang dianalisis adalah memori yang digunakan pada saat proses *indexing* sedang berlangsung (*memory usage* aplikasi) dan juga memori yang dipakai untuk menyimpan data dari *index* yang telah terbentuk (*memory index file*).

4.3.1 Pengujian Berdasarkan Jumlah Dokumen

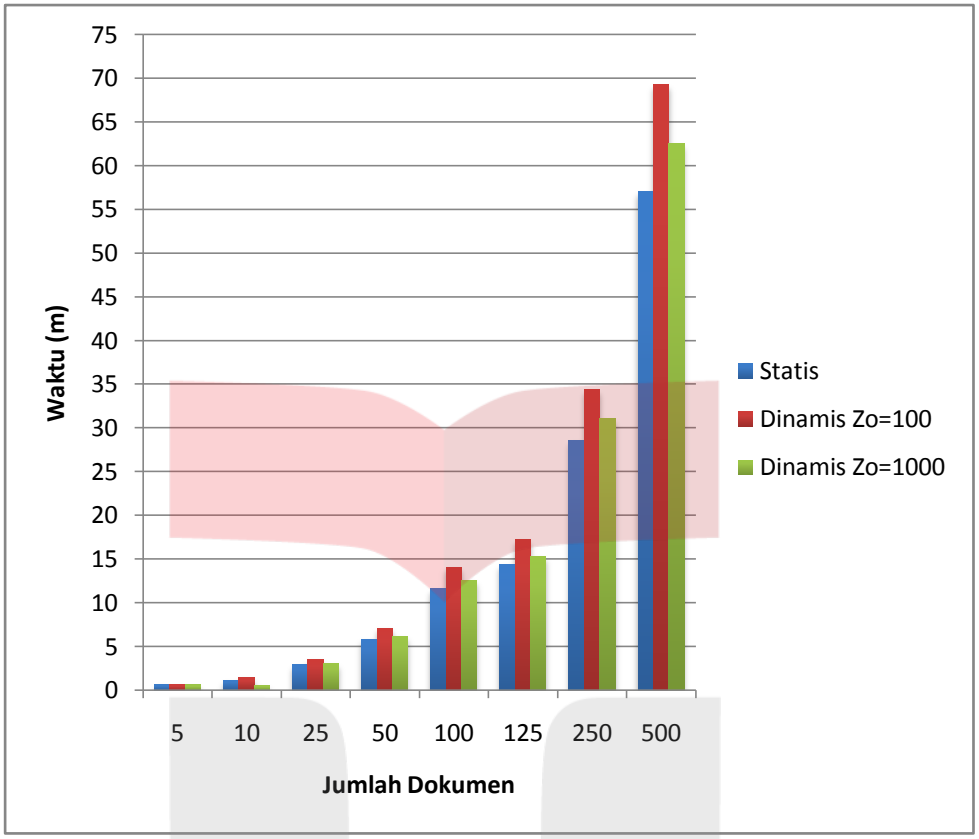
Pengujian dilakukan untuk mengetahui pengaruh dari jumlah dokumen terhadap parameter pengujian. Pada pengujian ini dibandingkan performansi kedua buah metode saat melakukan proses pembangunan index dimana jumlah revisitnya adalah nol, atau pada saat pembangunan awal suatu index file. Nilai perbandingan yang didapatkan dari pengujian berdasarkan jumlah dokumen dipaparkan pada tabel 4-1.

Tabel 4-1 : Nilai Perbandingan Performansi Berdasarkan Jumlah Dokumen

Σ DOK	PENGINDEKSAN								
	STATIS			DINAMIS Z_0 = 100			DINAMIS Z_0 = 1000		
	Waktu (m)	Memory Usage Aplikasi (kB)	Memory Index File (kB)	Waktu (m)	Memory Usage Aplikasi (kB)	Memory Index File (kB)	Waktu (m)	Memory Usage Aplikasi (kB)	Memory Index File (kB)
5	0.66	11477.33	405.56	0.67	12442.33	412.27	0.60	12310.67	374.70
10	1.16	11690.33	1124.20	1.43	11274.67	1116.94	0.56	12807.00	1102.38
25	2.95	11245.67	2614.05	3.48	12313.67	2620.27	3.03	12588.00	2561.18
50	5.82	11626.67	5429.52	7.09	12949.67	5429.44	6.17	11610.67	5393.27
100	11.63	11684.67	9594.27	14.04	10171.00	9595.11	12.58	13016.00	9601.11
125	14.33	11622.00	10879.66	17.25	11908.33	10877.28	15.32	11433.67	10809.48
250	28.55	11288.00	19901.02	34.38	15369.00	19902.76	31.03	11830.33	19873.36
500	57.03	11543.00	41411.59	69.31	27966.33	41414.20	62.54	20228.33	41390.53

1. Waktu

Pengujian dilakukan untuk mengetahui perbandingan performansi, dalam hal ini parameter waktu, yang digunakan pada saat proses pembangunan index antara sistem yang mengimplementasikan metode *static indexing* dan sistem yang mengimplementasikan metode *dynamic indexing*. Perbandingan performansi berdasarkan waktu digambarkan dalam bentuk grafik seperti terlihat pada gambar 4-2

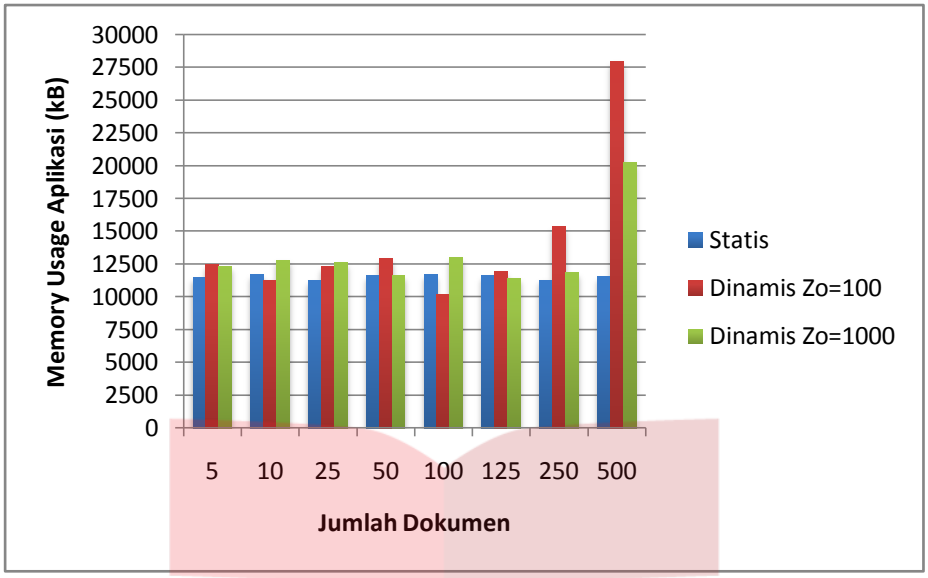


Gambar 4-2 : Grafik Perbandingan Waktu Berdasarkan Jumlah Dokumen

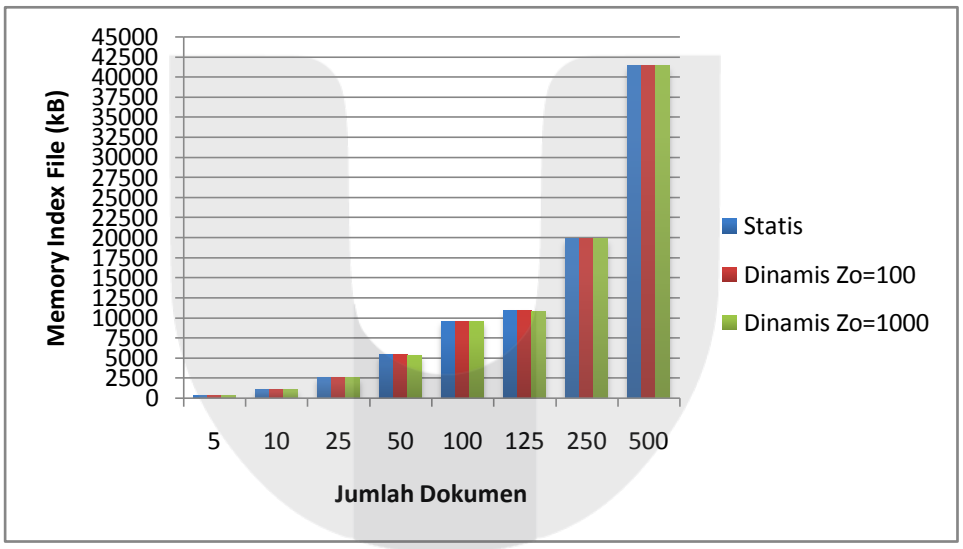
Berdasarkan gambar 4-2 diketahui bahwa ***dynamic indexing*** **memerlukan waktu relatif lebih lama dibandingkan dengan** ***static indexing*** **untuk pembangunan awal suatu** ***index file*****.** Hal ini dapat disebabkan karena *term-term* yang telah diolah pada proses *preprocessing* oleh *dynamic indexing* dan akan disimpan ke dalam *index file* harus dimasukkan ke dalam *temporary index* terlebih dahulu hingga ukuran *temporary* mencapai ukuran maksimum yang telah ditentukan. Sedangkan pada *static indexing*, *term-term* yang telah di olah pada proses preprocessing akan langsung dimasukkan ke dalam *index file*. semakin banyak dokumen yang diproses oleh dynamic indexing, maka semakin lama pula waktu untuk prosesnya.

2. Memori

Pengujian dilakukan untuk mengetahui perbandingan performansi, dalam hal ini parameter memori, yang digunakan pada saat proses pembangunan index. Memori yang dimaksudkan adalah memori yang terpakai saat aplikasi dijalankan dan memori yang dipakai untuk menyimpan *index file* pada *database*. Perbandingan performansi berdasarkan memori akan terbagi menjadi dua bagian dan tergambar pada gambar 4-3 dan gambar 4-4 :



Gambar 4-3 : Grafik Perbandingan Memory Usage Aplikasi berdasarkan Jumlah Dokumen



Gambar 4-4 : Grafik Perbandingan Memory Index File berdasarkan Jumlah Dokumen

Berdasarkan gambar 4-3 dan 4-4 dapat diketahui bahwa *memory index file* yang digunakan oleh *static indexing* dan *dynamic indexing* memiliki nilai yang hampir sama. **Akan tetapi untuk *memory usage* aplikasi, *dynamic indexing* memiliki nilai yang rata-rata relatif lebih besar jika dibandingkan dengan *static indexing*. Hal ini menunjukkan bahwa *dynamic indexing* memerlukan sejumlah memory tambahan sebagai tempat penyimpanan *term* sementara saat nilai *temporary* belum mencapai maksimum**, contohnya adalah pada dynamic indexing dengan jumlah 500 dokumen, pada gambar 4-4 terlihat bahwa kasus ini memiliki ukuran *index file* yang setara dengan *static indexing* yang mengolah data dengan jumlah yang sama besar. Akan tetapi pada gambar 4-3 dapat terlihat bahwa dynamic indexing dengan jumlah 500 dokumen memiliki ukuran *memory usage* aplikasi yang jauh lebih besar dibanding dengan *static indexing* yang mengolah jumlah data yang sama. Hal ini membuktikan bahwa **jumlah dokumen tidak mempengaruhi besarnya ukuran *memory usage* aplikasi yang digunakan.**

4.3.2 Pengujian Berdasarkan Jumlah Revisit

Pengujian dilakukan untuk mengetahui pengaruh proses revisit terhadap proses pembangunan kembali *index file*. Pengujian akan dibagi berdasarkan banyaknya jumlah dokumen yang diproses pada saat proses revisit berlangsung. Performansi dari masing-masing metode diukur lalu dibandingkan dengan melihat persentase perubahan dokumen. Pengukuran parameter waktu pada pengujian ini dihitung dari awal dilakukannya proses revisit hingga proses pengindeksan selesai dijalankan.

Tabel 4-2 : Tabel Persentase Perubahan Dokumen hasil Revisit

∑ Dok	REVISIT											
	1				2				3			
	Dok Ubah	Dok Statis	Dok hapus	Presentase Dok berubah	Dok Ubah	Dok Statis	Dok Hapus	Presentase Dok berubah	Dok Ubah	Dok Statis	Dok Hapus	Presentase Dok berubah
5	3	2	0	60%	3	2	0	60%	3	2	0	60%
10	8	2	0	80%	8	2	0	80%	8	2	0	80%
25	22	3	0	88%	20	3	2	80%	20	3	0	87%
50	44	6	0	88%	42	3	5	84%	42	3	0	93.3%
100	88	12	0	88%	84	6	10	84%	85	5	0	94.4%
125	102	23	0	81.60%	97	17	11	77.60%	96	18	0	84.2%
250	172	78	0	68.80%	161	67	22	64.40%	154	74	0	67.50%
500	390	110	0	78%	335	100	65	67%	330	105	0	75.86%

Berdasarkan tabel 4-2, perubahan jumlah dokumen yang didapat setelah dilakukannya revisit rata-rata memiliki nilai persentase yang lebih besar dari 50%. Perubahan yang terjadi pada dokumen dapat berupa penambahan atau pengurangan jumlah term ditandai dengan bertambah atau berkurangnya ukuran file dokumen. Pada proses revisit ke-2 terdapat penghapusan sejumlah dokumen, hal ini dikarenakan pada keadaan sebenarnya dokumen tersebut telah dihapus dari direktori penyimpanannya. Sehingga pada saat *crawler* bekerja untuk mencari dokumen tersebut, crawler tidak dapat menemukan dokumen yang dicari dan menghapus data URLnya dari *database* yang dimiliki oleh sistem. ***Dynamic indexing*** **hanya akan memproses kembali dokumen yang berubah, sedangkan *static indexing* akan memproses kembali semua jumlah dokumen yang ada.** Nilai perbandingan performansi pada pengujian ini, dijelaskan pada table 4-3:

Tabel 4-3 : Nilai Perbandingan Performansi berdasarkan Jumlah Revisit dan batasan dokumen

KETERANGAN DOKUMEN		Revisit								
		1			2			3		
		waktu (m)	memory usage (kB)	memory disk (kB)	waktu (m)	memory usage (kB)	memory disk (kB)	waktu (m)	memory usage (kB)	memory disk (kB)
5 DOK	Statis	1.34	11325.00	435.01	1.38	11478.67	353.91	1.35	11354.67	326.28
	Dinamis $z_0 = 100$	1.09	13323.33	439.42	1.12	12162.33	352.24	1.04	11778.00	472.16
	Dinamis $z_0 = 1000$	1.03	12521.67	374.47	1.02	11962.67	490.24	1.04	11664.00	266.99

10 DOK	Statis	2.49	11504.33	1123.21	2.45	11286.00	962.40	2.39	11408.67	883.38
	Dinamis $z_0 = 100$	2.40	12076.67	1128.54	2.23	11522.33	2027.11	2.18	11919.67	1943.16
	Dinamis $z_0 = 1000$	2.22	12583.33	1016.03	2.19	12828.67	915.10	2.06	11909.33	1310.93
25 DOK	Statis	7.36	11185.67	2586.28	7.29	11460.67	2266.02	6.76	11189.67	2018.57
	Dinamis $z_0 = 100$	6.24	12109.33	2583.64	6.25	12715.33	4317.91	6.27	12452.00	5983.26
	Dinamis $z_0 = 1000$	6.77	11685.00	2539.17	6.84	11335.33	2206.42	6.24	12378.33	2605.12
50 DOK	Statis	12.57	11609.67	5322.76	12.52	11717.33	4853.76	11.90	11552.00	4342.21
	Dinamis $z_0 = 100$	11.87	12367.67	5327.91	13.43	12647.33	8816.51	12.38	11712.33	4341.06
	Dinamis $z_0 = 1000$	12.32	11930.67	5222.36	12.61	12878.33	4792.55	11.28	13201.33	5123.23
100 DOK	Statis	26.17	11662.67	9215.66	24.92	11587.33	8088.00	24.60	11346.00	7283.52
	Dinamis $z_0 = 100$	27.12	14503.67	9217.75	26.75	12155.33	16939.15	25.21	14246.67	16234.98
	Dinamis $z_0 = 1000$	25.76	11953.67	9121.80	26.79	15150.33	8065.46	24.06	13243.00	11812.01
125 DOK	Statis	32.51	11665.33	10468.88	30.98	11312.00	9281.73	29.93	11369.00	8446.85
	Dinamis $z_0 = 100$	32.19	11123.33	10469.78	30.81	12448.33	17039.07	30.20	32326.67	9514.22
	Dinamis $z_0 = 1000$	30.46	12357.67	10449.78	29.07	11905.00	9267.88	27.24	12412.00	10870.98
250 DOK	Statis	59.17	11198.00	19316.43	56.93	11382.00	16565.20	54.53	11933.00	15422.45
	Dinamis $z_0 = 100$	53.63	105604.33	19317.65	53.46	12237.00	32470.88	49.18	9076.67	31861.18
	Dinamis $z_0 = 1000$	50.56	62771.33	19281.60	51.06	12353.33	15417.53	49.52	11864.67	23014.80
500 DOK	Statis	125.73	12736.00	41072.08	114.07	12312.00	34159.07	111.25	12120.00	31162.61
	Dinamis $z_0 = 100$	123.51	191159.33	41080.91	110.09	12242.67	68732.73	103.53	21707.00	67029.11
	Dinamis $z_0 = 1000$	119.49	13526.00	41078.88	105.99	51146.00	34089.25	97.54	13431.67	46869.66

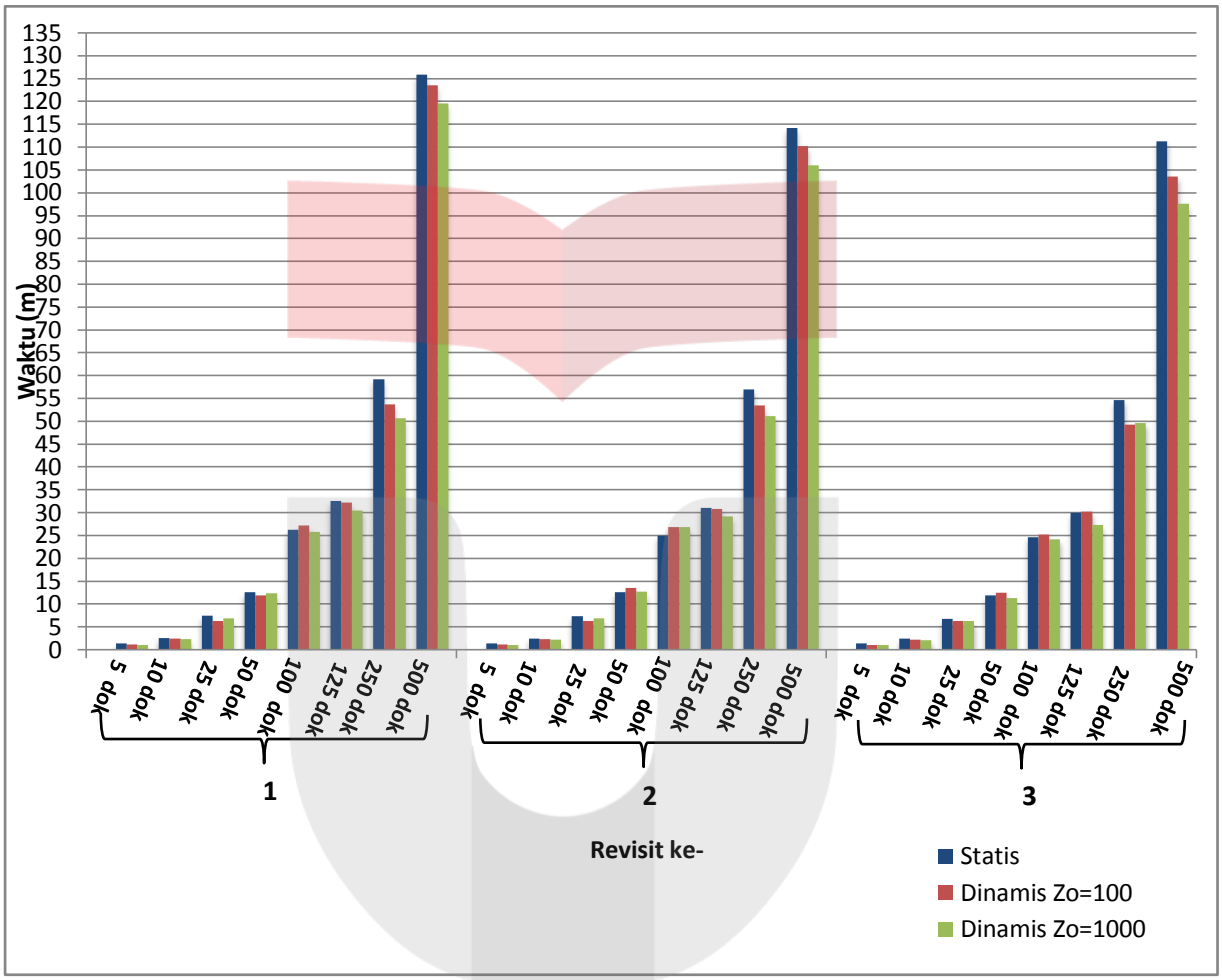
Proses *revisit* atau pengunjungan kembali dilakukan sebanyak tiga kali, dokumen yang yang di-*revisit* adalah dokumen yang sebelumnya telah diambil oleh *mobile crawler*. Pada tugas akhir ini, tidak terdapat penambahan dokumen pada proses *revisit*, yang ada hanyalah penambahan dan pengurangan jumlah term.

Terlihat pada tabel 4-3 bahwa waktu yang digunakan untuk proses pengindeksan dokumen pada saat revisit ke-2 relatif lebih lama jika dibandingkan dengan proses pengindeksan dokumen pada saat revisit ke-3 yang memiliki presentase perubahan dokumen lebih besar. Hal ini disebabkan karena pada saat proses revisit ke-2 terdapat beberapa URL yang dihapus, sehingga jumlah dokumen awal pada proses revisit ke-3 lebih kecil.

Perbandingan performansi dari masing-masing sistem, digambarkan dalam bentuk grafik dimana grafik perbandingannya terbagi atas tiga bagian, yaitu :

1. Waktu

Grafik Perbandingan performansi parameter waktu berdasarkan jumlah revisit dapat dilihat pada gambar 4-5

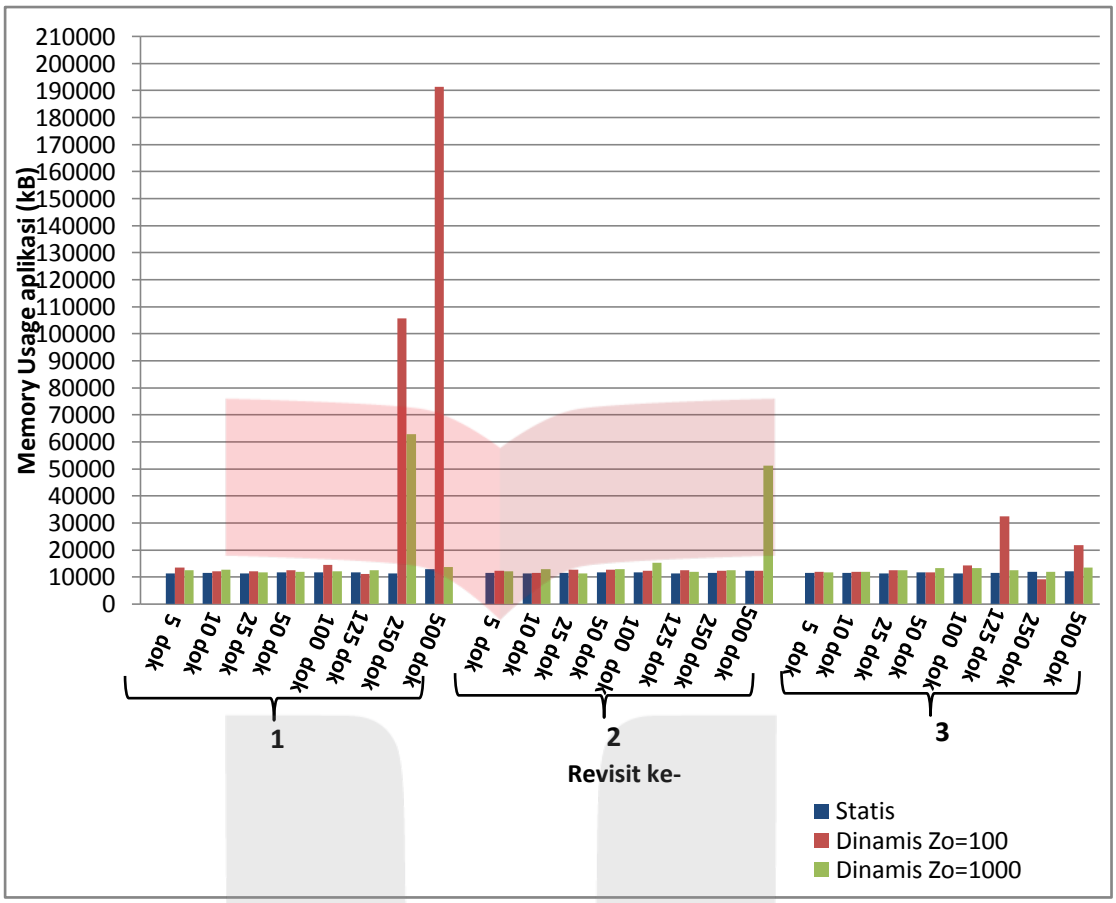


Gambar 4-5 : Grafik Perbandingan Performansi Waktu Berdasarkan jumlah Revisit

Berdasarkan gambar 4-5 diketahui bahwa semakin banyak jumlah dokumen yang harus melewati proses pengindeksan kembali, maka semakin banyak pula waktu yang diperlukan oleh sistem *static indexing* dan *dynamic indexing* untuk melakukan proses pengindeksan.

2. Memory Usage Aplikasi

Grafik Perbandingan performansi parameter *memory usage aplikasi* berdasarkan jumlah revisit dapat dilihat pada gambar 4-6

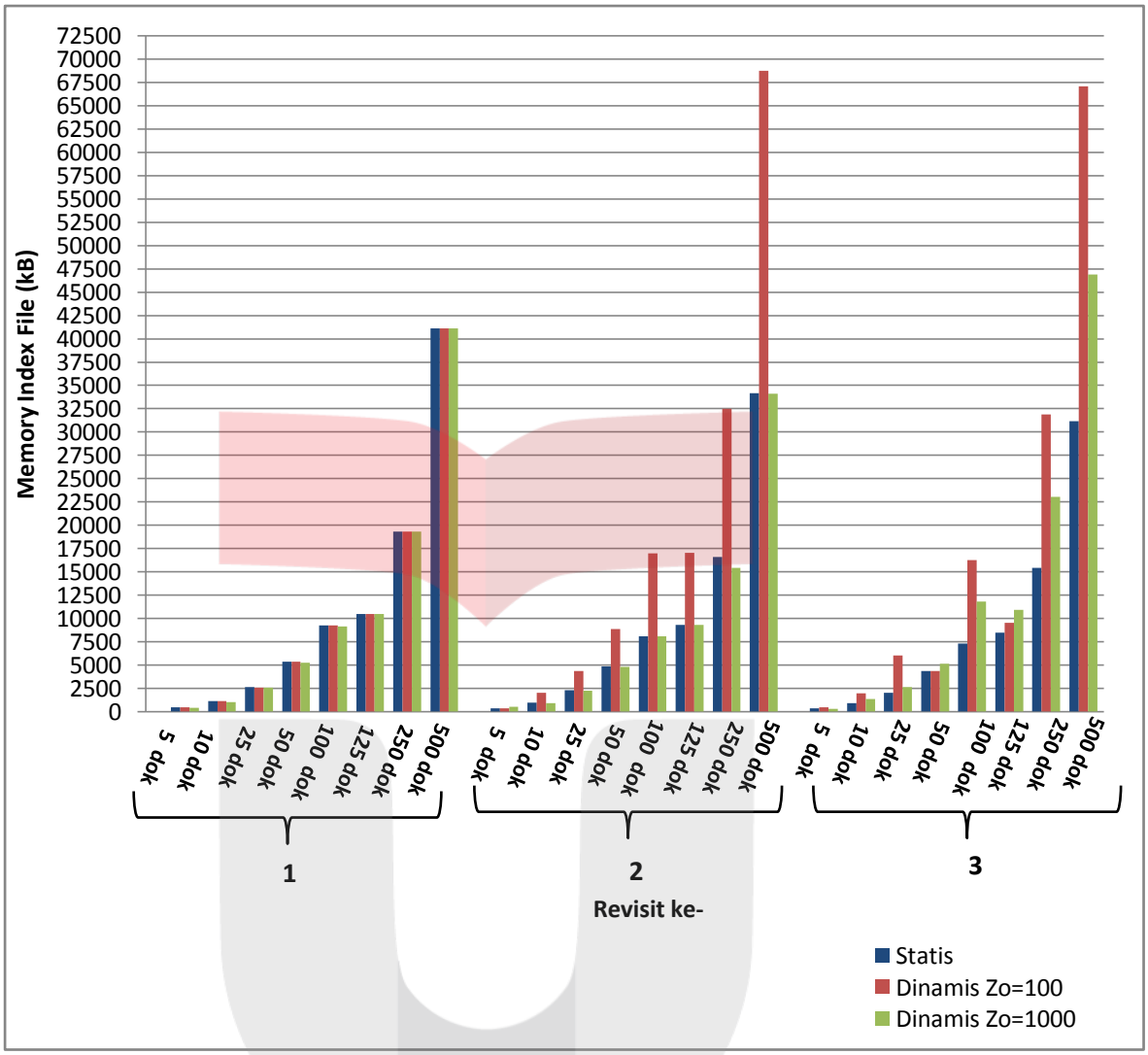


Gambar 4-6 : Grafik Perbandingan Performansi Memory Usage Aplikasi Berdasarkan jumlah Revisit

Berdasarkan gambar 4-6 dapat diketahui bahwa perubahan *memory usage* aplikasi tidak bergantung pada banyaknya *revisit* yang dilakukan ataupun banyaknya jumlah dokumen yang diakses dan perlu pengindeksan kembali, ***memory usage* bergantung pada jumlah *term* dokumen yang tertampung didalamnya. Semakin banyak *term* yang tertampung, maka akan semakin besar memori yang digunakan.** *Memory usage* untuk *static indexing* terlihat lebih stabil dibandingkan dengan *memory usage* yang dipakai oleh *dynamic indexing.*

3. Memory Index File

Grafik Perbandingan performansi parameter memory *index file* berdasarkan jumlah revisit dapat dilihat pada gambar 4-7



Gambar 4-7 : Grafik Perbandingan Performansi Memory Index File Berdasarkan jumlah Revisit

Setelah dilakukan pengujian berdasarkan jumlah revisit, diperoleh hasil bahwa jumlah perubahan dokumen setiap proses revisit dapat mempengaruhi performansi sistem jika dilihat dari parameter waktu. **Semakin banyak jumlah dokumen yang berubah dalam suatu proses revisit, maka akan semakin lama waktu yang digunakan untuk proses pengindeksan kembali.** Untuk kasus dalam tugas akhir ini, waktu yang digunakan untuk proses revisit mempengaruhi waktu pengukuran proses pengindeksan. **Waktu yang diperlukan sistem yang mengimplementasikan *static indexing* relatif lebih lama jika dibandingkan dengan sisem yang mengimplementasikan dynamic indexing, hal ini disebabkan karena sistem yang mengimplementasikan *static indexing* harus kembali membangun index file dari awal.**

Memory usage aplikasi yang digunakan untuk pengindeksan dengan metode *static indexing* lebih stabil jika dibandingkan dengan metode *dynamic indexing*. Static indexing tidak memerlukan sejumlah memory tambahan untuk menjalankan sistemnya, berbeda dengan *dynamic indexing* yang memerlukan *memory* tambahan yang digunakan untuk menyimpan *term* sementara selama sistem masih

berjalan. Ukuran dari *memory temporary* pada *dynamic indexing* bergantung pada jumlah dan ukuran term yang tersimpan didalamnya.

Memory index file pada *dynamic indexing* dipengaruhi oleh *memory usage* aplikasi, semakin besar ukuran *memory usage* aplikasi maka semakin banyak jumlah *term* yang tersimpan di dalamnya. Sehingga ukuran *index file* yang dihasilkan akan relatif lebih kecil dibandingkan dengan ukuran *index file* yang seharusnya. *Memory Index file* yang dimiliki oleh dynamic indexing memiliki ukuran memori yang tidak stabil dibandingkan dengan *static indexing*, disebabkan oleh jumlah *term* yang tersimpan di dalam *temporary* tidak menentu

4.3.3 Pengujian Berdasarkan Ukuran Z_0

Pengujian dilakukan dengan mengubah ukuran dari Z_0 yang merupakan temporary index. Z_0 digunakan untuk menampung *term* sementara pada sistem *dynamic indexing*. Batasan ukuran yang diujikan pada tugas akhir ini adalah sebesar 100 dan 1000 term.

Nilai perbandingan performansi antara dynamic indexing yang menggunakan Z_0 = 100 dan Z_0 = 1000 dijelaskan pada tabel 4-4:

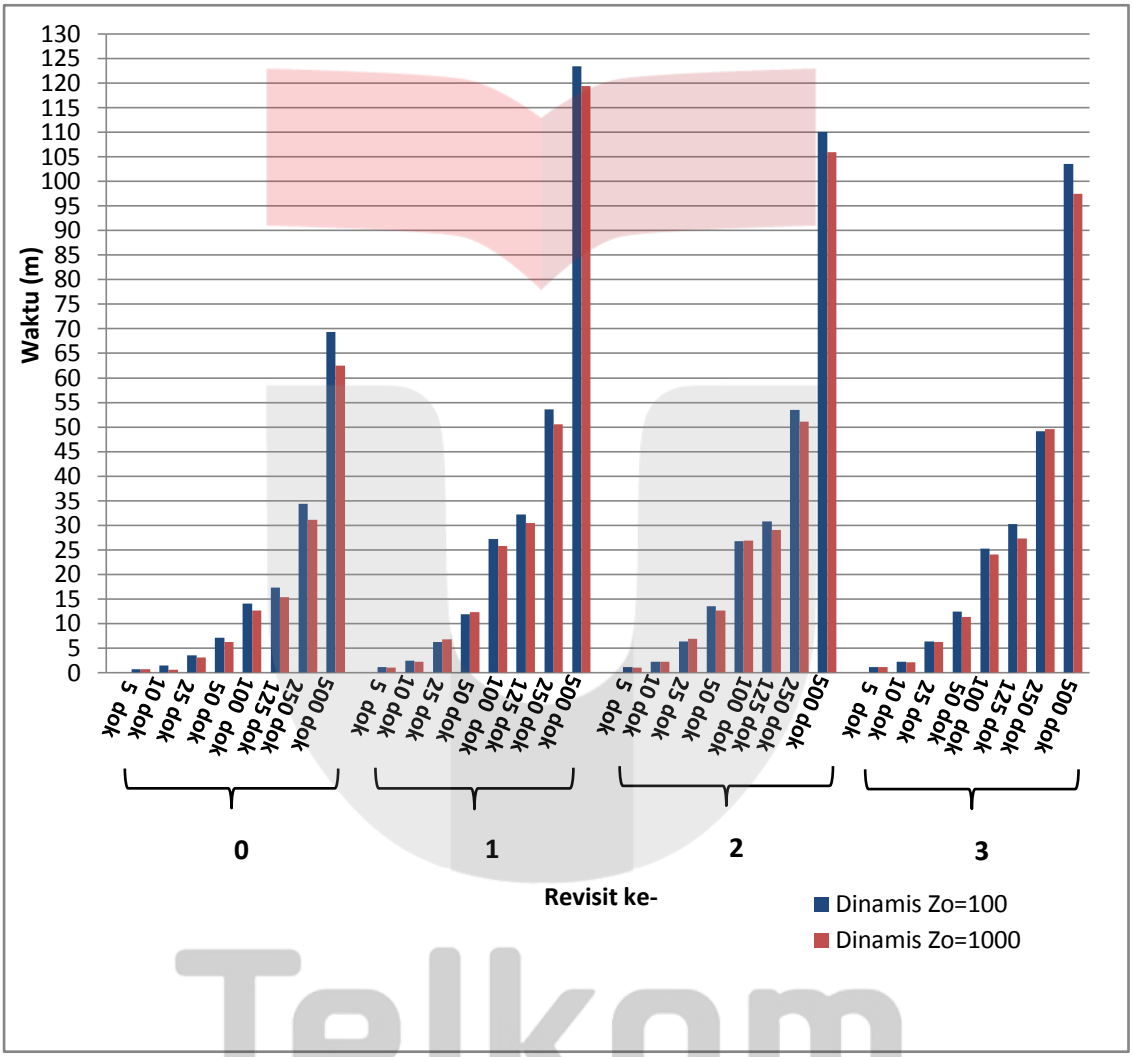
Tabel 4-4 : Nilai Perbandingan Performansi berdasarkan Ukuran Z_0 pada Dynamic I

∑ DOK	Size Z_0	REVISIT									
		0			1			2			
		Waktu (m)	Memory Usage Aplikasi (kB)	Memory Index File (kB)	Waktu (m)	Memory Usage Aplikasi (kB)	Memory Index File (kB)	Waktu (m)	Memory Usage Aplikasi (kB)	Memory Index File (kB)	Waktu (m)
5	100	0.67	12442.33	412.27	1.09	13323.33	439.42	1.12	12162.33	352.24	1.04
	1000	0.60	12310.67	374.70	1.03	12521.67	374.47	1.02	11962.67	490.24	1.04
10	100	1.43	11274.67	1116.94	2.40	12076.67	1128.54	2.23	11522.33	2027.11	2.18
	1000	0.56	12807.00	1102.38	2.22	12583.33	1016.03	2.19	12828.67	915.10	2.06
25	100	3.48	12313.67	2620.27	6.24	12109.33	2583.64	6.25	12715.33	4317.91	6.27
	1000	3.03	12588.00	2561.18	6.77	11685.00	2539.17	6.84	11335.33	2206.42	6.24
50	100	7.09	12949.67	5429.44	11.87	12367.67	5327.91	13.43	12647.33	8816.51	12.38
	1000	6.17	11610.67	5393.27	12.32	11930.67	5222.36	12.61	12878.33	4792.55	11.28
100	100	14.04	10171.00	9595.11	27.12	14503.67	9217.75	26.75	12155.33	16939.15	25.21
	1000	12.58	13016.00	9601.11	25.76	11953.67	9121.80	26.79	15150.33	8065.46	24.06
125	100	17.25	11908.33	10877.28	32.19	11123.33	10469.78	30.81	12448.33	17039.07	30.20
	1000	15.32	11433.67	10809.48	30.46	12357.67	10449.78	29.07	11905.00	9267.88	27.24
250	100	34.38	15369.00	19902.76	53.63	105604.33	19317.65	53.46	12237.00	32470.88	49.18
	1000	31.03	11830.33	19873.36	50.56	62771.33	19281.60	51.06	12353.33	15417.53	49.52
500	100	69.31	27966.33	41414.20	123.51	191159.33	41080.91	110.09	12242.67	68732.73	103.53
	1000	62.54	20228.33	41390.53	119.49	13526.00	41078.88	105.99	51146.00	34089.25	97.54

Berdasarkan tabel 4-4 akan didapatkan grafik perbandingan performansi dynamic indexing berdasarkan parameter pengujian :

1. Waktu

Berikut ini adalah grafik pengujian yang didapatkan berdasarkan tabel 4-4 jika dilihat dari parameter waktu.

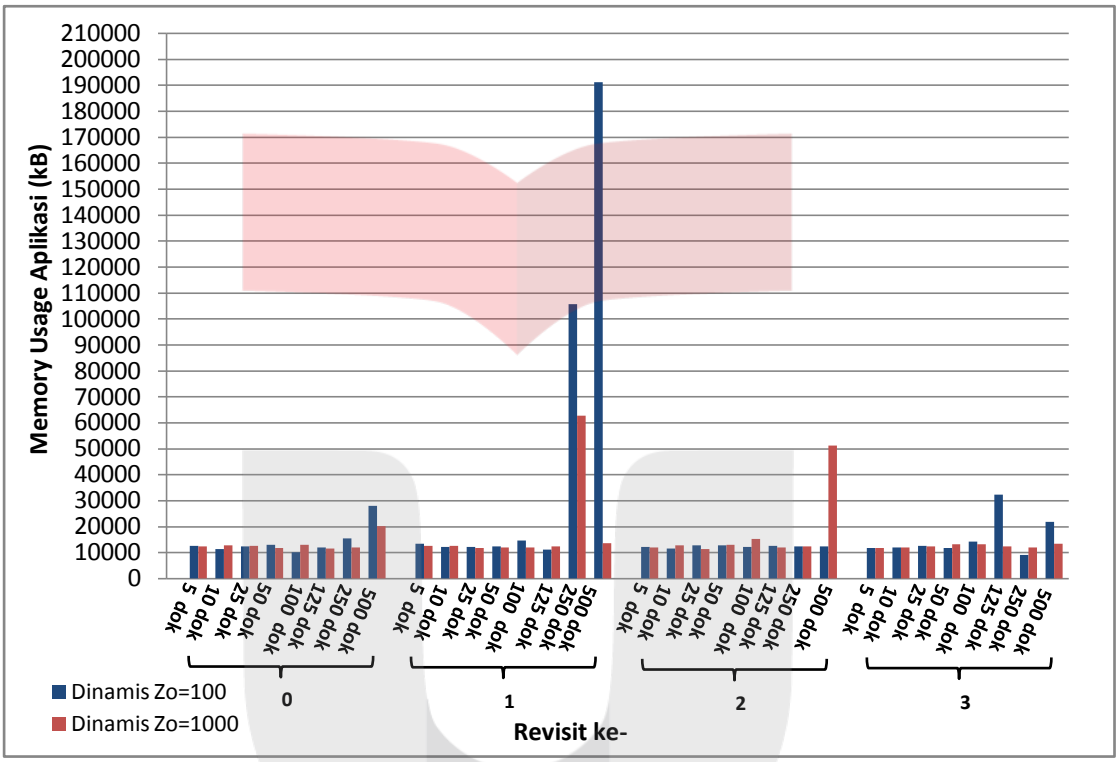


Gambar 4-8 : Grafik Perbandingan Performansi Waktu antara Dynamic Indexing Z_0=100 dan Z_0 =1000

Pada gambar 4-8 dapat diketahui bahwa *dynamic indexing* dengan nilai *temporary index* Z_0 = 1000 memiliki performansi waktu yang lebih baik jika dibandingkan dengan dynamic indexing dengan nilai temporary Z_0 = 100. **Waktu pembentukan konstruksi indeks dari *dynamic indexing*, Z_0=1000 lebih singkat. Hal ini dikarenakan nilai maksimum *temporary index* Z_0=100 akan lebih cepat terpenuhi sehingga *system* lebih banyak mengakses *database* untuk menyimpan data *term* yang sebelumnya telah tertampung di dalamnya**. Pengaksesan database inilah yang membuat proses penyimpanan *index file* pada *dynamic indexing*, Z_0=100 menjadi lebih lambat.

2. Memory Usage Aplikasi

Berikut ini adalah grafik pengujian yang didapatkan berdasarkan tabel 4-4 jika dilihat dari parameter *memory usage* aplikasi.

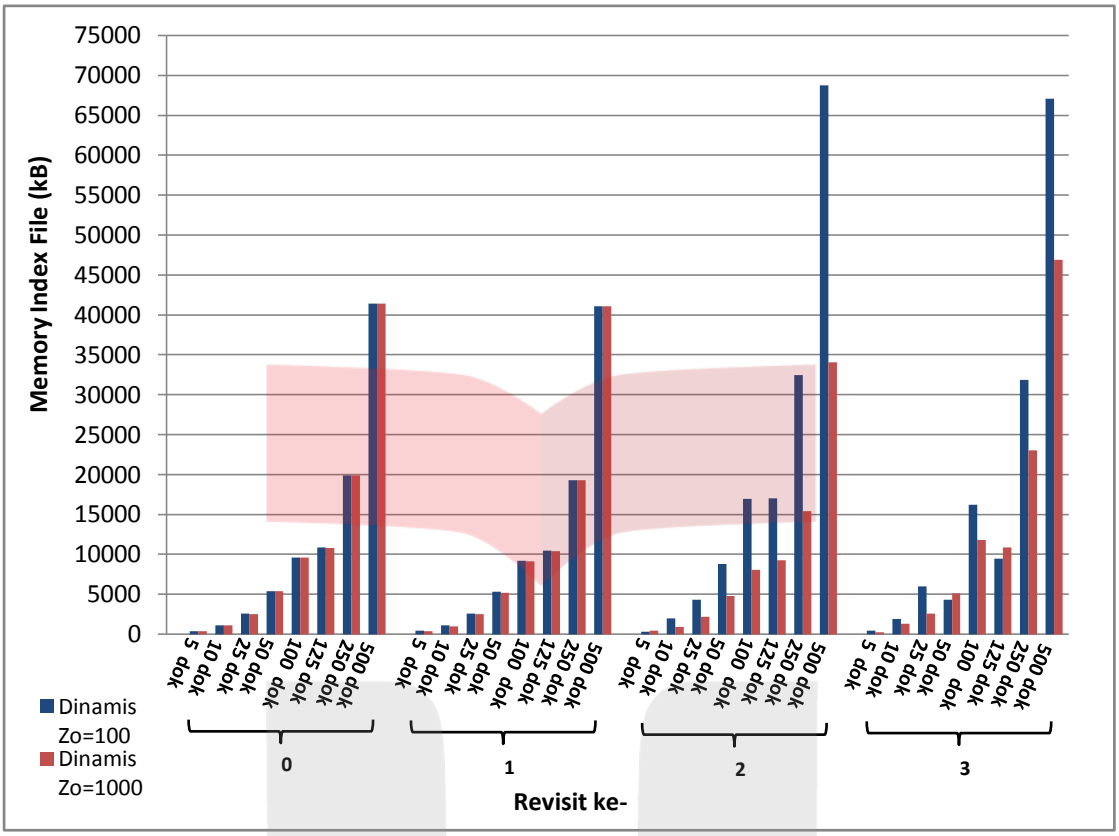


Gambar 4-9 : Grafik Perbandingan Performansi Memory Usage Aplikasi antara Dynamic Indexing Z_0=100 dan Z_0 =1000

Berdasarkan gambar 4-9 dapat diketahui bahwa *dynamic indexing* dengan Z_0=100 lebih banyak menggunakan memori tambahan untuk menyimpan term pada *temporary index*, sehingga memakan banyak *memory usage* aplikasi. Contohnya terlihat pada kasus *dynamic indexing* dengan batasan 500 dokumen dimana memory usage aplikasinya terlihat lebih menjulang dibandingkan dengan kasus yang lainnya.

3. Memory Index File

Berikut ini adalah grafik pengujian yang didapatkan berdasarkan tabel 4-4 jika dilihat dari parameter *memory usage* aplikasi.



Gambar 4-10 : Grafik Perbandingan Performansi Memory Index File antara Dynamic Indexing Z_0=100 dan Z_0 =1000

Berdasarkan gambar 4-10 dapat diketahui bahwa *dynamic indexing* dengan Z_0=100 lebih banyak menggunakan *memory index file* dibandingkan *dynamic indexing* dengan Z_0=1000.

Berdasarkan pengujian ini, **dynamic indexing dengan nilai Z_0=1000 memiliki performansi yang lebih baik jika dilihat dari parameter waktu dan memory yang digunakan dibandingkan dynamic indexing dengan nilai Z_0=100.**

Daftar Pustaka

[1] Bal, S and Nath, R,*Filtering the web pages that are not modified at remote site without downloading using mobile crawler*. Information Technology journal 9(2)2010 ISSN 1812- 5638, Asian Network for Sciencetific information. (pp: 376-380)
available at: http://docsdrive.com/pdfs/ansinet/itj/2010/376-380.pdf, diakses tanggal : 15 Januari 2011

[2] Boon, Peter. 2005. *Mobile Agent Systems Implemented with Java*.
available at: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.122.5207 &rep=rep1&type=pdf, diakses tanggal : 19 Desember 2010.

[3] Firdaus, Yanuar. 2008 : *Text Processing Method*. Bandung : IT Telkom.
available at: http://www.ittelkom.ac.id/staf/yanuar, diakses tanggal: 10 oktober 2009.

[4] Firdaus, Yanuar. 2008 : *Web Search*. Bandung : IT Telkom. available at: http://www.ittelkom.ac.id/staf/yanuar, diakses tanggal: 10 oktober 2009.

[5] Joachim, Hammer and Jan Fiedler. 2000. *Using Mobile Crawler to Search the Web Efficiently*. International Journal of Computer and Information Science. 1 : 36-58.

[6] Green, S. et al., *Software Agents: A review*, Technical Report, Department of Computer Science, Trinity College, Dublin, Ireland.

[7] Lange, D. B., Chang, Daniel T. *IBM Aglets Workbench White Paper*
available at: http://www.trl.ibm.co.jp/aglets/whitepaper.html,diakses tanggal : 12 Desember 2010

[8] Lange, D.B. and M. Oshima. 1998. *Mobile Agents with Java : The Aglet API*. Available at: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.96.1463&rep=rep1&type=pdf , diakses tanggal 12 Desember 2010.

[9] Manning, Christopher D., Prabhakar Raghavan, Hinrich Schütze. 2009. *An Introduction to Information Retrieval*. Cambridge : Cambridge University Press.

[10] Shah, Shalin. *Implementing Effective Web Crawler*.
http://www.devbistro.com/articles/Misc/Implementing-Effective-Web-Crawler, diakses tanggal: 8 oktober 2009.

[11] The Web Robots Pages. About the Robots <META> tag.
available at: http://www.robotstxt.org/meta.html, diakses tanggal 12 Desember 2010

[12] Wikipedia. Robot.txt.
available at : http://en.wikipedia.org/wiki/Robots_exclusion_standard, diakses tanggal 12 Desember 2010.