

# Optimasi Alur Kerja Ekstraksi dan Transformasi dalam proses ETL dengan Representasi State-Space dan Algoritma Heuristic-greedy

## Extraction and Transformation Workflow Optimization in ETL process with State-Space Representation and Heuristic-greedy Algorithm

Muhammad Fardan

Universitas Telkom  
muhammad.fardan55@gmail.com

---

**Abstrak** - Extraction, Transformation, dan Loading, yang lebih dikenal dengan singkatan ETL, merupakan salah satu proses yang sangat penting dalam penerapan konsep data warehouse. Secara garis besar tugas dari proses ETL adalah untuk mengumpulkan data yang berasal dari sumber data dengan sistem operasi dan environment berbeda-beda menjadi sebuah kumpulan besar data yang terintegrasi. Dengan demikian semakin besarnya jumlah data yang akan diintegrasikan maka akan semakin berat dan semakin lama pula kinerja dari proses ETL. Hal tersebut akan mempengaruhi performansi data warehouse keseluruhan. Dalam buku tugas akhir ini penulis melakukan optimasi terhadap ETL dengan representasi state-space dan algoritma heuristic-greedy yang dapat mengurangi jumlah baris data yang diproses dalam setiap transformasi. Kemudian dilakukan analisis pada hasil optimasi ETL dengan membandingkan antara cost value dan waktu eksekusi yang dihasilkan algoritma heuristic-greedy dan algoritma exhaustive. Hasil akhir dari penelitian tugas akhir ini menunjukkan waktu eksekusi algoritma heuristic-greedy jauh lebih sedikit dibandingkan dengan algoritma exhaustive, dan algoritma heuristic-greedy menghasilkan nilai cost minimal yang sedikit lebih besar atau sama dengan nilai cost minimal yang dihasilkan algoritma exhaustive. Maka dari itu algoritma heuristic-greedy memberikan solusi yang sangat baik, karena dapat menghasilkan nilai cost minimal yang cukup memuaskan jika dibandingkan dengan hasil dari algoritma exhaustive dengan waktu yang jauh lebih cepat.

**Kata kunci** : etl, optimasi etl, data warehouse, heuristic-greedy, state-space, cost model

---

**Abstract** - Extraction, Transformation, and Loading, also known as ETL, is an important process in implementation of data warehouse concept. In general, ETL's task is to collect data from various sources with different environments and operation systems to become an integrated data collection. Thus, the greater size of data, the greater workload and execution time of ETL process, and that will influence all over datawarehouse performance. In this book, the author performs an ETL optimization with state-space representation and heuristic-greedy algorithm that can reduce the number of rows of data that are processed in each transformation. Afterward, the results of optimization are analyzed by comparing the cost value and execution time between heuristic-greedy and exhaustive algorithms. The end result of this research shows the execution time of heuristic-greedy algorithm is much less than the exhaustive algorithm, and heuristic-greedy algorithm generates minimal cost value slightly greater than or equal to the value of minimal cost that generated by exhaustive algorithm. Thus heuristic-greedy algorithm provides an excellent solution, because it can generates a minimum cost value that quite satisfactory when compared with the results of the exhaustive algorithm with much faster time.

**Keywords** : etl, optimasi etl, data warehouse, heuristic-greedy, state-space, cost model

---

## **1. Pendahuluan**

### **1.1 Latar Belakang**

Sistem informasi berperan penting dalam perkembangan bisnis saat ini. Sistem informasi membantu perusahaan dalam meningkatkan tujuan dan strategi bisnis serta membantu perusahaan dalam mengidentifikasi serta mengatasi masalah dan juga kelemahan yang dimiliki perusahaan [Lip13]. Informasi – informasi tersebut dapat diperoleh dengan mengolah data yang dimiliki perusahaan, maka kesuksesan pada setiap kegiatan bisnis yang ada dalam perusahaan bergantung terhadap manajemen pengolahan data penting yang dimiliki perusahaan [Rau07]. Dan salah satu konsep pengolahan data yang baik adalah menggunakan data warehousing. Menurut beberapa pakar industri, sekitar 60% - 80% upaya dalam proyek data warehouse dihabiskan dalam proses ETL [Gou10]. Proses ETL menangani volume data yang besar serta kerja yang rumit, sehingga memerlukan waktu operasional yang relatif panjang dan biaya yang tidak sedikit dalam pengadaan system resource-nya [Siv12]. Dalam penelitian ini akan ditawarkan solusi untuk mengurangi jumlah baris data yang diproses dengan mengatur ulang susunan transformasi pada ETL dengan representasi state-space dan penerapan algoritma heuristic-greedy.

### **1.2 Pokok Masalah**

Jurnal ini akan membahas bagaimana mengurangi beban ETL dengan optimasi dalam representasi state-space. Kemudian menganalisis seberapa baik penerapan algoritma heuristic-greedy dalam mengurangi beban ETL.

### **1.3 Tujuan**

Penelitian ini bertujuan untuk mendesain alur kerja ETL menjadi bentuk graf asiklik, kemudian menerapkan algoritma heuristic-greedy serta lima jenis transisi pada graf : swap, merge, split, factorize, dan distribute untuk optimasi. Selanjutnya melakukan analisis terhadap hasil optimasi ETL dengan parameter cost value, yang dipengaruhi oleh jumlah baris yang diproses dalam transformasi, dan waktu eksekusi algoritma

## **2. Data Warehouse**

### **2.1 Data Warehouse**

Data warehouse adalah sebuah sistem yang mengekstrak, menyaring, menyesuaikan (integrasi), dan menyalurkan sumber data menuju penyimpanan data dimensional dan kemudian mendukung serta mengimplementasikan query dan menganalisisnya untuk tujuan pengambilan keputusan (decision-making) [Kim04].

### **2.2 ETL**

ETL merupakan singkatan dari Extraction, Transformation, dan Load. Proses ETL adalah proses awal yang dilewati data sebelum masuk ke dalam data warehouse. Extraction dilakukan untuk membaca data dari sebuah sumber basis data tertentu dan meng-ekstrak subset data yang dibutuhkan. Ekstraksi dilakukan secara berkala dan terjadwal, bergantung pada kebutuhan sistem pengolahan data. Berikut adalah daftar dari beberapa topik pokok pada ekstraksi data [Pon01]. Tahap transformasi melakukan perubahan terhadap data hasil ekstraksi sebelum nantinya di-load ke tabel data warehouse. Proses transformasi dapat dilakukan dengan beberapa cara tergantung kondisi dari tiap data : memilih detail data yang dibutuhkan saja sebagai informasi yang sesuai dengan kebutuhan data warehouse, memberi tambahan informasi pada data agar sesuai dengan kualitas data yang diinginkan, atau merubah standar dan format data agar menjadi satu format yang sama, sesuai dengan standar yang diijinkan pada data warehouse. Pada tahap load, data hasil transformasi dimasukkan ke dalam tabel data warehouse. Pemrosesan load dapat dilakukan dengan dua cara : 1) dengan refresh data, yaitu menghapus seluruh data di tabel data warehouse lalu memasukkan data baru mulai dari periode awal hingga akhir, atau 2) dengan update data, yaitu memilih data yang belum dimasukkan berdasarkan periode sebelum dimasukkan ke tabel data warehouse.

## **3. State-Space**

State-space dapat diartikan sebagai kumpulan konfigurasi atau kondisi yang mungkin dicapai oleh suatu masalah dan keadaan sekelilingnya [Sut10]. Alur kerja ETL yang ada akan dirubah ke dalam bentuk graf melalui tahap pemodelan konseptual dan logikal. Graf tersebut akan diterapkan kombinasi satu atau lebih transisi graf dan akan menghasilkan graf-graf baru, kemudian graf-graf tersebut akan membentuk sebuah kumpulan state. Lalu dengan algoritma pencarian, akan dicari semua state yang mungkin terbentuk, dan kumpulan state ini membentuk sebuah state-space.

Graf  $G(V, E)$  terdiri dari kumpulan vertex  $V$  dan edge  $E$ , dimana setiap vertex pada mewakili semua aktivitas dan recordset dan edge menunjukkan sebuah hubungan antara dua buah vertex; antara penyedia dan pemakai data. Aktivitas merupakan transformasi yang diterapkan dalam alur kerja, dan recordset merupakan tabel sumber ataupun tabel target pada alur kerja ETL. Sedangkan edge menunjukkan sebuah hubungan antara recordset dan aktivitas ataupun aktivitas dengan aktivitas. Ada dua jenis aktivitas, yaitu unary dan binary. Unary mendapatkan masukan dari satu provider, dan binary mendapatkan masukan dari dua provider. Secara umum setiap aktivitas memiliki lima elemen utama : 1) input schemata, kumpulan field masukan, 2) functional schemata, kumpulan field yang diproses, 3) generate schemata, kumpulan field yang dihasilkan dari proses, 4) projected-out schemata, kumpulan field yang tidak akan diproses lagi pada aktivitas selanjutnya, dan 5) output schemata, seluruh field yang dikeluarkan aktivitas.

#### 4. Heuristic-Greedy

Heuristic berasal dari bahasa Yunani kuno *heuriskein* dan bahasa latin *heuristicus*, yang berarti mencari atau menemukan. Algoritma heuristic dapat diartikan sebagai kumpulan aturan, metode, atau trik yang digunakan untuk meningkatkan efisiensi penyelesaian masalah yang rumit. Pada kebanyakan kasus algoritma heuristic memberikan solusi yang mendekati solusi optimal, dan tak jarang pula menghasilkan solusi optimal.

Algoritma greedy merupakan metode yang digunakan untuk memperkecil ruang pencarian pada solusi masalah. Algoritma greedy mengambil solusi terbaik pada setiap sub-ruang pencarian, lalu meneruskan pencarian solusi dari sub-solusi tersebut.

#### 5. Cost Model

Cost model merupakan bentuk perencanaan perhitungan beban untuk mengestimasi beban suatu alur kerja yang dibentuk. Dalam buku ini digunakan cost model yang pernah dikemukakan oleh Wentao Wu dan tim pada International Conference Data Engineering (ICDE), sebagai berikut :

$$CO = ns.cs + nr.cr + nt.ct + ni.ci + no.co$$

cs : seq page cost, cost I/O untuk mengakses tuple secara terurut. cr : random page cost, cost I/O untuk mengakses tuple secara acak. ct : cpu tuple cost, cost CPU untuk memproses sebuah tuple. ci : cpu index tuple cost, cost CPU untuk memproses sebuah tuple menggunakan akses indeks. co : cpu operator cost, cost CPU untuk melakukan operasi seperti hash atau agregasi. Sedangkan (ns, nr, nt, ni, no) : jumlah tuple yang diproses pada masing-masing cost.



#### 6. Deskripsi Umum Sistem





Borang pada Naskah Akademik Akreditasi merupakan salah satu perangkat instrumen akreditasi yang dibuat BAN-PT sebagai upaya peningkatan mutu program studi di Indonesia. Skema data warehouse terdiri dari sepuluh tabel dimensi dan lima tabel fact. Skema lengkap datawarehouse dapat dilihat pada lampiran 1. Data yang akan dijadikan data sumber merupakan data akademik dan non-akademik yang dikelola oleh Direktorat Sistem Informasi Universitas Telkom.

#### 7. Model Konseptual

Model konseptual didesain dalam empat langkah. Langkah pertama adalah identifikasi sumber data dilakukan dengan memeriksa langsung ke sumber data. Pada tahap ini identifikasi hanya dilakukan pada lapisan skema basis data dan tabel-tabel pada setiap skema. Langkah berikutnya adalah mengidentifikasi lebih lanjut sumber data hingga ke bagian field atau kolom dan nilai dari field tersebut, kemudian dipetakan antara field pada tabel target dengan field pada tabel sumber yang memiliki kemungkinan untuk menyediakan data. Langkah ketiga adalah mendesain model konseptual untuk setiap alur kerja. Pada langkah ini, alur kerja mulai dibuat dengan cara menggambarkan hasil pemetaan pada langkah sebelumnya dan dilengkapi dengan proses ekstraksi ataupun transformasi yang mungkin terjadi pada setiap field dari tabel sumber sebelum dimasukkan ke tabel target. Untuk setiap tabel target, dibuat satu buah alur kerja.

Tabel 1 : Simbol Model Konseptual

Gambar Simbol	Nama	Deskripsi
	concept	mewakili sebuah entitas pada sumber data atau <i>data warehouse</i> .
	attribute	merupakan sebuah sifat atau keterangan yang menempel pada sebuah entitas








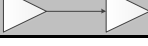



	transformation	menggambarkan proses ekstraksi dan transformasi yang memproses <i>attribute</i>
	note	label tambahan dari desainer untuk memberikan penjelasan terhadap transformation
	serial composition	kombinasi transformation yang harus dilalui <i>attribute</i> sebelum masuk ke tabel target, dihubungkan dengan sebuah garis dan biasanya dilengkapi dengan nama <i>attribute</i> yang diproses
	part of relationship	garis penghubung antara concept dan <i>attribute</i> yang dimilikinya

Langkah berikutnya adalah menguji kebenaran model konseptual pada tool ETL, dalam proyek ini digunakan Oracle Warehouse Builder (OWB) sebagai tool ETL. Seluruh elemen pada model konseptual diterapkan dalam tool sesuai dengan fungsi yang disediakan

## 8. Model Logikal

Tahap selanjutnya adalah merubah alur kerja model konseptual ke dalam bentuk graf asiklik. Pada tahap ini dilakukan penjelasan lebih detail terhadap transformasi yang terdapat dalam diagram yang dihasilkan model konseptual. Langkah pertama yang dilakukan adalah mengganti semua simbol menggunakan simbol model logikal untuk membedakan alur yang masih dalam bentuk model konseptual dengan model logikal.

Tabel 2 : Simbol Model Logikal

Simbol Konseptual	Simbol Logikal	Nama
		concept => recordset
		<i>attribute</i>
		transformation => aktivitas
		notes
		serial composition
		part of relationship
		<i>attribute</i> indicator
	 <p>SHEMATA FUN : {} GEN : {} PRD : {}</p>	<i>schematas</i> annotation

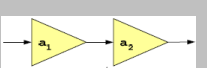
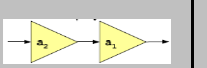
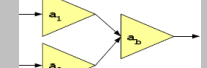
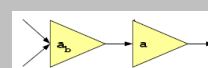
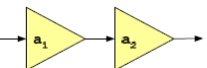
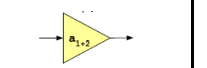
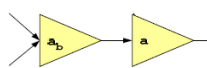
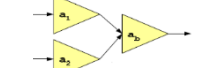
Langkah berikutnya adalah menentukan urutan prioritas aktivitas. Pada model konseptual, sebuah transformasi dapat dihubungkan ke dua atau lebih transformasi setelahnya, sedangkan pada model logikal sebuah aktivitas hanya dapat dihubungkan ke satu aktivitas, dengan hubungan sebagai penyedia dan pemakai data. Maka dari itu aktivitas perlu diberi prioritas untuk mengetahui urutan penyedia dan pemakai data.

## 9. Optimasi

### 9.1 Transisi Graf

Ada 5 jenis transisi : swap, merge, split, factorize, dan distribute, seperti yang sudah dijelaskan pada bab sebelumnya. Jenis masukan prosedur ini adalah satu atau lebih aktivitas (vertex) bergantung pada jenis transisi yang diterapkan, dan keluarannya berupa aktivitas-aktivitas dengan susunan yang berbeda. Setiap transisi memiliki aturan untuk menentukan vertex mana yang dapat ditransisi dan tidak.

Tabel 3 : Jenis Transisi Graf

Jenis Transisi	Contoh Masukan	Contoh Keluaran	Jenis Transisi	Contoh Masukan	Contoh Keluaran
<i>Swap</i>			<i>Factorize</i>		
<i>Merge</i>			<i>Distribute</i>		



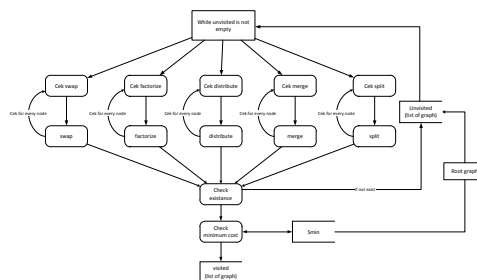
### 9.2 Penghitungan Cost

Pada bagian cost model telah dijelaskan model cost yang digunakan. Pada bagian ini dibahas cara perhitungan cost dengan lebih rinci. Nilai konstanta  $c$  untuk setiap elemen mengambil nilai default yang sudah ada,  $c_s = 1.0$ ,  $c_r = 4.0$ ,  $c_t = 0.01$ ,  $c_o = 0.0025$ . Sedangkan nilai  $n$  untuk setiap elemen ditentukan tergantung ekspresi aljabar setiap aktivitas. Asumsikan bahwa jumlah tuple masukan pada aktivitas adalah  $T$ , maka  $nt = T$  untuk semua ekspresi aljabar. Nilai  $nr$  untuk selection, notnull, join, dan distinct  $nr = T$ , sedangkan untuk source, aggregation, function, sk assignment, dan union  $nr = 0$ . Nilai  $ns$  untuk aggregation, function, sk assignment  $ns = 0$ , dan yang lainnya  $ns = T$ . Nilai  $no$  untuk selection, notnull, dan distinct  $no = T \cdot \log T$ , untuk join, aggregation, function, dan sk assignment  $no = T$ , sedangkan untuk source dan union  $no = 0$  [WuW13].

Estimasi nilai tuple masukan didapat dari nilai tuple keluaran aktivitas sebelumnya (penyedia data). Asumsikan bahwa nilai tuple masukan  $T_{in}$ , nilai tuple keluaran  $T_{out}$ , dan nilai tuple keluaran aktivitas sebelumnya  $T_{prev-out}$ , maka nilai  $T_{in}$  untuk join  $T_{in} = T_{prev1-out} \times T_{prev2-out}$ , untuk union  $T_{in} = T_{prev1-out} + T_{prev2-out}$ , sedangkan untuk yang lainnya  $T_{in} = T_{prev-out}$ . Kemudian nilai estimasi  $T_{out}$  didapat dari hasil query agregasi count pada database sumber.

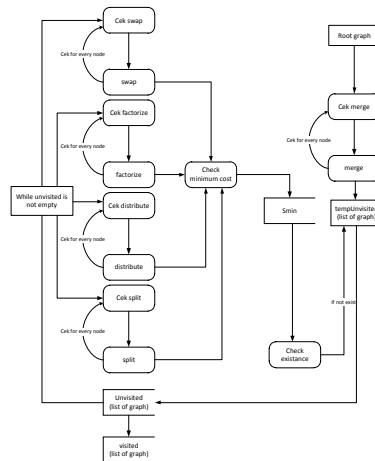
### 9.3 Algoritma Pencarian

Algoritma exhaustive akan mencari seluruh kemungkinan bentuk graf yang dapat dibentuk dari hasil setiap transisi. Asumsikan bahwa alur kerja ETL hasil model logikal merupakan graf akar. Algoritma exhaustive dimulai dengan menelusuri setiap aktivitas pada graf akar, kemudian diperiksa apakah dapat diterapkan setiap transisi graf. Jika graf dapat ditransisi, maka hasil tansisi disimpan dalam list unvisited. Setelah semua aktivitas selesai ditelusuri, maka graf akar disimpan dalam list visited. Kemudian pencarian diteruskan ke semua graf yang ada dalam list unvisited, kemudian graf dipindahkan dari unvisited ke visited jika telah selesai diperiksa semua aktivitasnya, begitu seterusnya hingga graf di list unvisited habis.



Gambar 1 : Alur Proses Algoritma Exhaustive

Algoritma heuristic-greedy merupakan gabungan dari heuristic dan greedy, dimana heuristic membatasi proses transisi yang terjadi dan greedy mengurangi ruang pencarian graf. Alur yang terjadi adalah pada awal pencarian seluruh aktivitas pada graf akar diperiksa untuk diterapkan transisi merge dan menghasilkan satu graf, graf merge. Selanjutnya setiap aktivitas pada graf merge ditelusuri untuk diterapkan transisi swap. Setelah di-swap hasil transisi dibandingkan cost value nya antara sebelum dan sesudah di-swap. Jika cost value hasil swap lebih minimal, maka graf hasil swap disimpan di list tempunvisited, jika sebaliknya maka tidak disimpan. Penuluran diteruskan ke transisi factorize hanya menggunakan graf pada tempunvisited dengan proses yang sama. Penuluran graf dilakukan hingga diterapkan transisi distribute dan split.



**Gambar 2 : Alur Proses Algoritma HS-Greedy**

## 10. Pembahasan

### 10.1 Ragam Graf

Berdasarkan teori pada literatur sumber, terdapat dua jenis graf dalam proyek tugas akhir ini, yaitu graf skala kecil dan menengah. Graf skala kecil terdiri kurang dari dua puluh vertex dan skala menengah memiliki dua puluh sampai empat puluh vertex [Sim05]. Tabel berikut menampilkan informasi dari sebelas graf yang dihasilkan.

**Tabel 4 : Karakteristik Graf**

Skema alur ETL	Jml Vertex	Jml Aktivitas	Jumlah Baris Masukan
Dim_dosen	5	3	[7204]
Dim_fakultas	3	1	[7]
Dim_jenjang_pend	14	11	[7204] [3462]
Dim_jns_tenaga_kepend	4	2	[672]
Dim_periode	9	7	[43928]
Dim_prodi	9	6	[50] [7]
Fact_aktivitas_dosen	24	17	[43928] [10519] [50] [7204] [672] [7]
Fact_ipk_lulusan	15	11	[54484] [50] [43509]
Fact_mahasiswa	14	10	[1771898] [54484]
Fact_tenaga_kepend	15	11	[7204] [672] [7]
Fact_dosen	18	14	[7204] [672] [7]

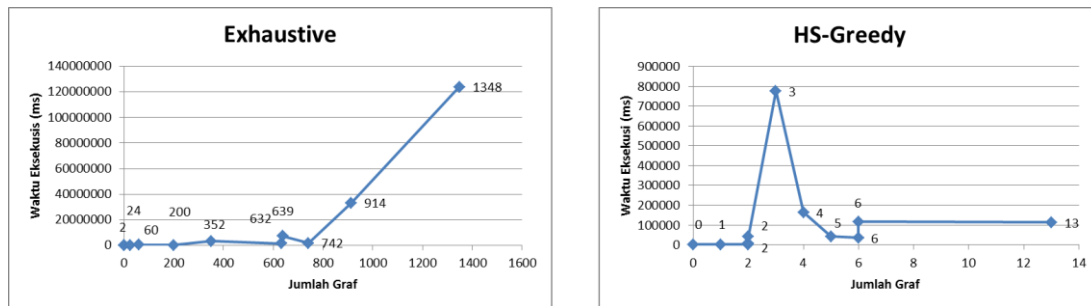
### 10.2 Waktu Eksekusi

**Tabel 5 : Hasil Pengujian Waktu Eksekusi**

Nama Skema	Waktu Rataan Exhaustive (ms)	Waktu Rataan HS-Greedy (ms)
Dim_dosen	36448	1689
Dim_fakultas	3061	1437
Dim_jenjang_pend*(2)	106473	41592
Dim_jns_tenaga_kepend	2353	1523
Dim_periode*(358)	7217711	35319
Dim_prodi	219551	6941
Fact_aktivitas_dosen*(6)	1465388	163314
Fact_ipk_lulusan* (335)	33153061	116852
Fact_mahasiswa*(322)	3197550	776751
Fact_tenaga_kepend*(44)	1536356	42215
Fact_dosen*(2830)	123825233	113075

(\*) waktu eksekusi diambil pada pengulangan pertama saat menghasilkan cost minimal untuk exhaustive

Tabel di atas menunjukkan waktu eksekusi dari lima percobaan yang dilakukan terhadap masing-masing skema dan algoritma. Kedua tabel di atas menunjukkan bahwa waktu eksekusi algoritma heuristic-greedy memang lebih cepat dibandingkan exhaustive pada semua skema.



Gambar 3 : Grafik Keterhubungan antara Jumlah Graf dan waktu Eksekusi

Grafik pada gambar di atas menunjukkan keterhubungan antara jumlah graf terbentuk (lihat Tabel 4-2) dan waktu eksekusi algoritma (lihat Tabel 4-3, 4-4). Pada algoritma exhaustive grafik menunjukkan bahwa semakin besar jumlah graf yang terbentuk maka semakin lama pula waktu eksekusinya, namun tidak demikian pada algoritma heuristic-greedy. Tidak terlihat adanya pola tertentu antara jumlah graf dengan waktu eksekusi pada algoritma heuristic-greedy. Grafik tersebut menunjukkan bahwa jumlah graf tidak sepenuhnya berpengaruh terhadap waktu eksekusi. Hanya pada algoritma exhaustive, jumlah graf memiliki pengaruh terhadap waktu eksekusi. Hal ini terjadi karena karakter algoritma yang berbeda, dimana algoritma heuristic-greedy tidak melakukan pengecekan keseluruhan terhadap seluruh graf yang mungkin terbentuk. Pembentukan graf baru pada algoritma heuristic-greedy terbentuk dari graf dengan nilai cost minimal dari setiap transisi, sehingga memungkinkan untuk membentuk graf lebih banyak dalam waktu yang sama ataupun lebih kecil.

### 10.3 Cost Value

Tabel 6 : Hasil Pengujian Cost Value

Nama Skema	Jumlah Aktivitas	Cost Value Minimal		Persentase Rasio
		Exhaustive	Heuristic-greedy	
Dim_dosen	3	43528,36	43528,36	100%
Dim_fakultas	1	7,23	7,23	100%
Dim_jenjang_pend*(250)	11	64865,27	82200,13	78,91%
Dim_jns_tenaga_kepend	2	4062,29	4062,29	100%
Dim_periode*(358)	7	267497,34	267497,34	100%
Dim_prodi	6	2030,18	2030,18	100%
Fact_aktivitas_dosen*(92)	17	2763095959,11	2763330017,96	99,99%
Fact_ipk_lulusan*(858)	11	7651935,55	5901067628,65	0,13%
Fact_mahasiswa*(322)	10	55269600053,40	55269600053,40	100%
Fact_tenaga_kepend*(250)	11	700275,65	700276,85	99,99%
Fact_dosen*	14	246488,50	246486,74	100%

Merujuk ke tabel 4-6 dapat dilihat bahwa algoritma exhaustive akan menghasilkan nilai cost minimal yang lebih baik pada kasus dimana skema memiliki 11 aktivitas atau lebih seperti pada skema dim\_jenjang\_pend dan fact\_tenaga\_kepend dengan perbedaan yang berkisar 70%-99%. Hal ini dapat disebabkan karena karakteristik ruang pencarian algoritma exhaustive lebih luas dibandingkan algoritma heuristic-greedy. Sedangkan untuk kasus dimana skema memiliki jumlah aktivitas kurang dari 10, algoritma heuristic-greedy sudah dapat memenuhi nilai cost minimal yang sama dengan algoritma exhaustive. Dari hal tersebut penulis kategorikan untuk skema dengan jumlah aktivitas kurang dari 11 sebagai skala kecil dan skema dengan jumlah aktivitas 11 ke atas sebagai skala menengah untuk studi kasus ini.

Jika dilihat kembali, nilai cost (Tabel 4-6) yang dihasilkan dipengaruhi oleh jumlah aktivitas (Tabel 4-1) pada graf skema. Semakin besar jumlah aktivitas, maka semakin besar pula nilai cost minimal yang dihasilkan. Namun pada skema fact\_aktivitas\_dosen terdapat perbedaan pola, yang disebabkan jumlah aktivitas binary yang dimilikinya cukup besar, yaitu lima buah. Aktivitas binary memungkinkan untuk 'memangkas' data yang diproses di aktivitas berikutnya.

## 11. Kesimpulan

Kesimpulan yang dapat penulis peroleh dari hasil pengujian adalah sebagai berikut :

- 1) Algoritma exhaustive menghasilkan nilai cost minimal yang lebih baik dibandingkan nilai cost hasil algoritma heuristic-greedy pada skala menengah karena pencarian solusi algoritma exhaustive yang lebih menyeluruh
- 2) Pada skala kecil algoritma exhaustive dan heuristic-greedy menghasilkan nilai cost minimal yang sama karena ruang pencarian keduanya yang lebih kecil.
- 3) Seluruh hasil pengujian menunjukkan bahwa waktu eksekusi algoritma heuristic-greedy jauh lebih singkat dibandingkan eksekusi algoritma exhaustive pada skema berskala kecil maupun menengah karena ruang pencarian heuristic-greedy lebih kecil daripada exhaustive.
- 4) Dengan dua faktor : nilai cost minimal dan waktu eksekusi, yang ditunjukkan dalam hasil pengujian, maka dapat diambil kesimpulan bahwa algoritma heuristic-greedy menghasilkan performansi yang lebih baik dibandingkan algoritma exhaustive pada kasus skala kecil maupun menengah. Dan optimasi ETL dengan penerapan algoritma heuristic-greedy dianjurkan bagi organisasi yang memiliki basis data sumber dengan jumlah baris data yang besar dan relasi tabel yang banyak.

## 12. Daftar Pustaka

- [1] [Cor09] Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). Introduction to Algorithms. London: MIT Press.
- [2] [Gou10] Gour, V., Sarangdevot, D. S., Tanwar, G. S., & Sharma, A. (2010). Improve Performance of Extract, Transform and Load (ETL) in Data Warehouse.
- [3] [Kim04] Kimball, R., & Caserta, J. (2004). The Data Warehouse ETL Toolkit : Practical Techniques for Extracting, Cleaning, Conforming, and Delivering Data. Indianapolis: Wiley Publishing, Inc.
- [4] [Lip13] Lipaj, D., & Davidaviciene, V. (2013). Influence of Information Systems on Business Performance.
- [5] [SNa06] Nagabushana, S. (2006). Data Warehousing : OLAP and Data Mining. New Delhi: New Age International Ltd. Publisher.
- [6] [Pon01] Ponniah, P. (2001). Data Warehousing Fundamentals : A Comprehensive Guide for IT Professionals. New York: John Wiley & Sons, Inc.
- [7] [Rau07] Rausch, N. A., & Wills, N. J. (2007). Super Size It!!! Maximize the Performance of Your ETL Processes. Data Warehousing, Management and Quality.
- [8] [Tri14] Rizki, T. (2014). Executive Information System of Informatics Faculty in Self Evaluation with Data Warehousing and Online Analytical Processing (OLAP) Approach. Bandung: Telkom Engineering School, Informatics Department.
- [9] [Rom85] Romanycia, M. H., & Pelletier, F. J. (1985, January). What is a heuristic? Computational Intelligence, I(1), 47-58.
- [10] [Sim04] Simitsis, A. (2004). Modeling and Optimization of Extraction-Transformation Loading (ETL) Processes in Data Warehouse Environments. Athena: National Technical University of Athens, School of Electrical and Computer Engineering.
- [11] [Sim05] Simitsis, A., Vassiliadis, P., & Sellis, T. (2005, October). State-Space Optimaization of ETL Workflows. IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, 17, 1404-1419.
- [12] [Siv12] Sivaganesh, K., Srinivasu, P., & Satapathy, D. C. (2012). Optimization of ETL Work Flow in Data Warehouse.
- [13] [Sut10] Sutcliffe, D. (2010). Dr. Geoff Sutcliffe : Course Information : COMP6210 : Artificial Intelligence : content : State Spaces. Retrieved April 1, 2014, from Department of Computer Science, University of Miami Web site: <http://www.cs.miami.edu/~geoff/Courses/COMP6210-10M/Content/StateSpaceSearch.shtml>
- [14] [WuW13] Wu, W., Chi, Y., Zhu, S., Tatemura, J., Hacıgumus, H., & Naughton, J. F. (2013). Predicting Query Execution Time : Are Optimizer Cost Models Really Unusable? 2013 IEEE 29th International Conference Data Engineering (ICDE) (pp. 1081-1092). Brisbane: IEEE.
- [15] [Adz03] Adzic, J., & Fiore, V. (2003). Data Warehouse Population Platform. Torino: Telecom Italia Lab.