IMPLEMENTASI DAN ANALISIS ALGORITMA PARALEL *FUZZY C-MEANS CLUSTERING* DENGAN PENDEKATAN *GRAPHICS PROCESSING UNITS* (GPU)

IMPLEMENTATION AND ANALYSIS PARALLEL FUZZY C-MEANS CLUSTERING ALGORITHM USING GRAPHICS PROCESSING UNITS (GPU)

Dimas Andika[1], Fhira Nhita, S.T, M.T.[2], Izzatul Ummah, S.T, M.T.[3]
Ilmu Komputasi, Fakultas Informatika, Universitas Telkom
[1]dimasandikaw@gmail.com , [2]fhiranhita@telkomuniversity.ac.id , [3]izzatulummah@telkomuniversity.ac.id

Abstrak

***Data mining* adalah metode untuk mengambil informasi pada *dataset*. Pada *data mining* teknik *clustering* mempunyai peran yang penting karena dapat digunakan untuk mengelompokkan penyakit berdasarkan karakteristik yang diderita. Permasalahan pada data medis adalah kesulitan mendapatkan kecepatan waktu eksekusi dan performansi yang baik dalam mengolah data berdimensi tinggi. *Fuzzy C-Means Clustering* (FCM) adalah algoritma *clustering* yang pengerjaannya berdasarkan pada logika *fuzzy*, dimana pengelompokan data berdasarkan pada nilai derajat keanggotaan dan mengizinkan data menjadi anggota lebih dari satu kelompok. Untuk mengatasi permasalahan tersebut maka FCM diimplementasikan dengan pendekatan *Graphics Processing Units* (GPU) sehingga dapat meningkatkan kecepatan waktu eksekusi dan performansi dibandingkan penerapannya secara sekuensial. Penerapan GPU mengoptimalkan kinerja komputasi karena dapat bekerja secara paralel. Salah satu metode penerapannya dengan menjadikan fungsi perhitungan dalam mencari nilai derajat keanggotaan dijalankan secara paralel di GPU. Hasilnya mampu mempersingkat waktu eksekusi sebesar 10,70287 detik untuk data berdimensi 15.154 dan 4,13423 detik untuk data berdimensi 12.600.**

Kata kunci: *Data Mining, Fuzzy C-Means*, paralel, *clustering, GPU*, Waktu Eksekusi.

Abstract

Data mining is a method for retrieving information in the dataset. In the data mining, clustering techniques have an important role because it can be used to classify diseases based on the characteristics suffered. Problems on the medical data is difficult to get the speed of execution time and good performance in high-dimensional data processing. Fuzzy C-Means Clustering (FCM) is a clustering algorithm based on fuzzy logic process, in which the grouping data based on the value of the degree of membership and allow the data become a member of more than one group. To overcome these problems, the FCM implemented with the approach of Graphics Processing Units (GPU) that can increase the speed of application execution and performance compared sequentially. Optimize the application of GPU computing performance because it can work in parallel. One method of application by making the calculation function in finding the value of the degree of membership is run in parallel on the GPU. The result is able to shorten the time of execution of 10.70287 seconds for the data dimension of 15 154 and 4.13423 seconds for the data dimension 12,600.

***Keywords:* Data Mining, Fuzzy C-Means, parallel, clustering, GPU, execution time.**

1. Pendahuluan

Penerapan teknologi dalam berbagai pengaplikasian ilmu pengetahuan, bisnis dan sosial akan menciptakan data yang jumlahnya sangat banyak. Efek penyimpanan data menimbulkan sulitnya pengolahan data agar mendapatkan suatu analisis, kesimpulan penting dan informasi yang berguna. Dari sejumlah data yang sangat besar ini harus dapat dikelompokan dan di teliti lebih lanjut untuk penerapan kembali dalam menggali informasi yang tersembunyi, contohnya data medis yang memiliki dimensi tinggi sangatlah membutuhkan sumber daya yang besar untuk bisa mengolah data tersebut, misalnya seperti data gen *expresion*. Dalam analisis klasifikasi medis banyak permasalahan yang tidak pasti. Dengan menggunakan algoritma *fuzzy clustering* permasalahan ini dapat diselesaikan. Hasil dari *clustering* nantinya akan digunakan sebagai acuan untuk para pakar dalam proses mendiagnosa penyakit yang dikelompokan berdasarkan gejala-gejala yang ditimbulkan dari peyakit tersebut.

Fuzzy C-Means (FCM) adalah algoritma yang penentuan kelompoknya berdasarkan nilai derajat keanggotan dari data tersebut. FCM merupakan adaptasi dari algoritma *K-Means* dengan keanggotaan halus atau bisa di artikan sebagai suatu data yang fleksibel dalam pengelompokannya dimana suatu data dapat masuk lebih dari satu kelompok. FCM adalah salah satu algoritma yang biasa digunakan untuk *clustering* (pengelompokan) suatu diagnosis gejala penyakit menurut referensi [5,16,18]. FCM adalah salah satu algoritma yang populer untuk *clustering* dan dapat diaplikasikan pada berbagai permasalahan teknik, geologi, *image analysis*, diagnosa medis dan lainnya. Algoritma FCM membutuhkan waktu eksekusi yang tidak sedikit dalam pengerjaannya menurut referensi [6, 10].

Graphics Processing Units (GPU) bekerja secara paralel yang dapat meningkatkan kecepatan dalam pengolahan data untuk menghasilkan waktu eksekusi yang lebih baik[2]. Dengan kelebihan GPU untuk mengolah data dan dilatarbelakangi oleh keperluan *resource* yang tinggi untuk mengaplikasikan algoritma FCM untuk mengolah data berdimensi tinggi, maka pada penelitian tugas akhir ini, akan dibangun sebuah sistem dari sebuah implementasi algoritma FCM paralel dikombinasikan dengan kinerja GPU sebagai pembanding kinerjanya dengan penggunaan FCM sekuensial.

Rumusan masalah penelitian ini adalah :

a. Bagaimana mengimplementasikan algoritma *Fuzzy C-Means* (FCM) dengan menggunakan pendekatan *Graphics Processing Unit* (GPU) pada data yang mempunyai banyak *record* dan *attribute* atau dimensi?
b. Bagaimana analisis hasil *clustering*, waktu eksekusi dan performansi dari FCM paralel dengan GPU dengan FCM sekuensial tanpa GPU, agar dapat diketahui nilai SSE, *speedup*, *performance improvement* dan efisiensi yang dihasilkan ?
c. Bagaimana pengaruh parameter *fuzziness*, *epsilon* dan jumlah *cluster* yang digunakan pada FCM sekuensial dan FCM paralel terhadap waktu eksekusi?

Tujuan dari penelitian ini adalah :

a. Mengimplementasikan algoritma FCM paralel dengan menggunakan pendekatan GPU.
b. Mengetahui perbedaan hasil waktu eksekusi dan performansi yang didapat antara algoritma FCM sekuensial dengan FCM paralel dari hasil SSE, *speedup*, *performance improvement* dan efisiensi.
c. Menjelaskan pengaruh parameter-parameter seperti *fuzziness*, *epsilon* dan jumlah *cluster* terhadap waktu eksekusi.

Batasan masalah penelitian ini adalah :

a. Data yang digunakan hanya bersifat *numerical* (tidak kategorikal).
b. Tidak membahas lebih jauh tentang *pre-processing* data.
c. Faktor *hardware* dan *software* tidak dibahas lebih lanjut.

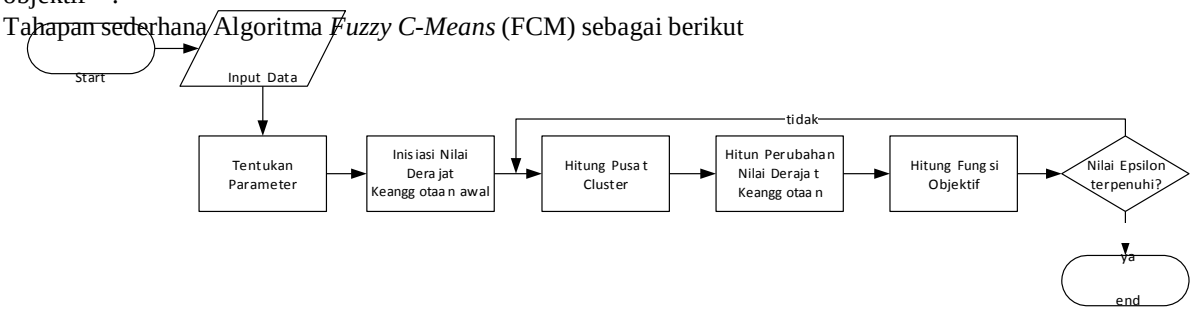
Penelitian ini diharapkan dapat memberikan tambahan pembelajaran mengenai pengelompokan data menggunakan algoritma *Fuzzy C-Means* (FCM) yang diimplementasikan menggunakan pendekatan *Graphic Processing Unit* (GPU).

2. Landasan Teori

2.1 *Fuzzy C-Means* (FCM)

Fuzzy C-Means adalah teknik *clustering* dimana keberadaan data dalam suatu *cluster* ditentukan oleh nilai derajat keanggotaan. FCM merupakan suatu algoritma *supervised clustering* karena jumlah *cluster* yang akan dibuat sudah di tentukan terlebih dahulu[8]. Algoritma ini pertama kali diperkenalkan oleh Jim Bezdek pada tahun 1981. FCM memungkinkan satu bagian dari data untuk memiliki dua atau lebih kelompok. Tujuan dari algoritma *Fuzzy C-Means* adalah untuk menemukan pusat *cluster* (*centroid*) dengan meminimumkan fungsi objektif [3].

Tahapan sederhana Algoritma *Fuzzy C-Means* (FCM) sebagai berikut



Gambar 2.1 Tahapan Sederhana Algoritma *Fuzzy C-Means* (FCM)

1. Input data yang akan di-*cluster* x, berupa matriks berukuran i x j (i = jumlah *record* data, j = *attribute* setiap data)
2. Tentukan parameter
   - Jumlah *cluster* = ◆
   - Pangkat pembobot = ◆
   - *Error* terkecil yang diharapkan = ◆
3. Inisiasi nilai derajat keanggotaan secara acak ◆ dimana i = 1,2,..., n; j = 1,2,...,c; sebagai elemen-elemen matrik partisi awal ◆
4. Hitung pusat *cluster* (*centroid*) dari masing-masing kelompok

$$◆ = \frac{\sum ◆ * ◆}{\sum ◆} \quad (1)$$

Untuk:

- c_i = Pusat *cluster* (*centroid*)
- μ_{ij} = Derajat keanggotaan titik ke-j di *cluster* ke-i
- m = Pangkat pembobot
- x = Data masukan ke-k

5. Hitung perubahan nilai derajat keanggotaan baru dari masing-masing data ke masing-masing kelompok

$$\mu_{ij} = \frac{1}{\sum_{k=1}^{c} \left[\sum_{p=1}^{n} (x_{ip} - c_k)/(x_{ip} - c_k)\right]^{\frac{2}{m-1}}} \quad (2)$$

6. Selanjutnya lakukan normalisasi fungsi objektif pada $P_t = \sum_{i=1}^{n} \sum_{j=1}^{c} \mu_{ij} \|x_i - c_j\|^2$ (3)
7. Cek kondisi berhenti

$$\max_{ij} \{|\mu_{ij}^{k+1} - \mu_{ij}^{k}|\} < \varepsilon \quad (4)$$

8. Jika tidak terpenuhi ulangi langkah ke-4

2.2 *Graphic Processing Unit* (GPU)

Graphics Processing Unit (GPU) adalah *chip* khusus pada sebuah *Video Graphics Array* (VGA) yang dirancang untuk memanipulasi dan mengubah memori sehingga mempercepat pembangunan grafis, namun pada beberapa tahun terakhir GPU juga digunakan untuk melakukan komputasi paralel[6]. GPU mempunyai *core* yang lebih banyak daripada *processor* CPU sehingga kini dimanfaatkan untuk pekerjaan komputasi yang berat, selain untuk game dan simulasi.

Komputasi paralel dengan GPU merupakan suatu metode komputasi yang masih baru. Pada awalnya komputasi paralel dapat dilakukan dengan menggunakan *grid computing*, di mana suatu pekerjaan didistribusikan ke banyak komputer yang saling terhubung melalui jaringan. GPU memiliki sejumlah *multi-core processor* dan setiap *processor* memiliki SIMD- *processor* (*Single Instruction Multiple Data*)[4]. SIMD artinya dengan satu instruksi dapat mengeksekusi sejumlah data paralel dalam waktu yang bersamaan. GPU kini sangat efisien dalam memanipulasi grafis komputer dan struktur kinerja paralel yang membuatnya lebih efektif dibandingkan pengerjaan menggunakan CPU untuk penerapan suatu algoritma, karena pengolahan blok besar data dilakukan secara paralel atau terpartisi.

Keunggulan GPU di antaranya :

1. Memiliki ribuan *core* (*multi-core processor*)
2. Dapat diprogram secara paralel
3. *Hardware* mendukung *multi threading* skala besar

2.3 CUDA

Compute Unified Device Architecture merupakan sebuah bahasa pemrograman berbasis C dan Fotran yang dikembangkan oleh NVIDIA untuk meningkatkan dan mempermudah optimasi sebuah GPU untuk keperluan non-grafis. Pada tahun 2007, Nvidia mempublikasikan teknik komputasi paralel baru yaitu CUDA yang menggunakan GPU sebagai unit pemrosesan. Selama ini GPU hanya digunakan untuk melakukan kalkulasi grafis, tetapi dengan CUDA, prosesor-porsesor grafis yang ada pada GPU dapat digunakan untuk menyelesaikan masalah-masalah yang lebih umum yang biasanya dikerjakan oleh CPU. GPU dirancang untuk melakukan banyak operasi secara paralel. Kemampuan eksekusi paralel secara masif inilah yang kemudian dapat dimanfaatkan oleh CUDA. CUDA kini memiliki kompabilitas dengan *platform* lain seperti MATLAB, Microsoft Visual Studio yang menambahkan *library*-nya untuk bisa menulis *source code* CUDA pada *platform* tersebut.

2.5 Evaluasi Cluster

Evaluasi *cluster* yang digunakan untuk menilai ketepatan sebuah teknik *clustering* dalam mengelompokkan data-data menjadi sebuah *cluster* salah satunya adalah menggunakan *Sum Squre Error* (SSE). Perhitungan yang di hasilkan SSE baik untuk membandingkan dua *clusterings* atau dua *clusters*[9]. SSE termasuk ke dalam *internal measure*. SSE dapat dijelaskan pada rumus berikut[10].

$$SSE = \sum_{k=1}^{K} \sum_{x \in C_k} dist^2(m_k, x) \quad (5)$$

Dimana x adalah titik data pada *cluster* C_k dan m titik representatif data untuk *clusters* C_k. Jika diberikan dua *clusterings*, maka kita dapat memilih mana yang memilik *error* lebih kecil[10].

2.6 Evaluasi Performansi Model

Pada penelitian ini, evaluasi dari performansi model menggunakan *speedup*, *performance improvement*, dan *efficiency*[1].

Speedup mengukur seberapa cepat waktu komputasi algoritma paralel dibandingkan dengan algoritma sekuensial.

$$Speedup = \frac{Wa[illegible]}{W[illegible]} \quad (6)$$

Performance improvement menggambarkan peningkatan performa yang menyatakan bahwa algoritma paralel lebih baik dari algoritma sekuensial.

$$Performance\ improvement = \frac{Wa[illegible] - Wa[illegible]}{W[illegible]} \quad (7)$$

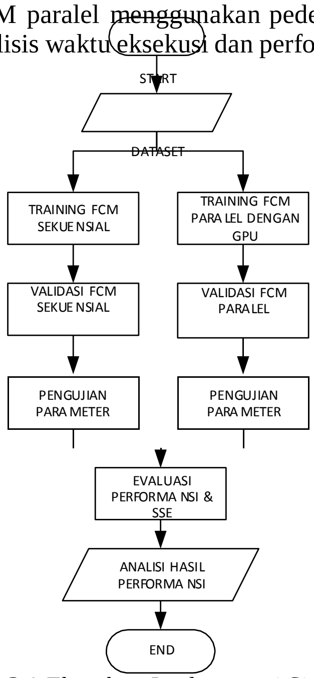
Efficiency digunakan untuk mengestimasi seberapa baik proses pemanfaatan *thread* dalam menyelesaikan masalah dibandingkan dengan usaha yang digunakan dalam berkomunikasi dan sinkronisasi.

$$Efficiency = \frac{Wa[illegible]}{[illegible] \times W[illegible]} \quad (8)$$

3. Perancangan Sistem

3.1 Deskripsi Sistem

Pada tugas akhir ini, akan di bangun sebuah sistem yang dapat mengimplementasikan algoritma FCM sekuensial dan FCM paralel dengan pendekatan *Graphics Processing Unit*. Proses yang akan dilakukan yaitu penerapan FCM sekuensial, penerapan FCM paralel menggunakan pedekatan GPU sebagai pembanding, tahap pelatihan dan pengujian *clustering* data, analisis waktu eksekusi dan performansi yang dihasilkan.



Gambar 3.1 *Flowchart* Implementasi Sistem

Dalam sistem yang dibangun, akan dilakukan analisis mengenai perbandingan akurasi dan waktu komputasi. Untuk akurasi dan waktu komputasi akan dibandingkan FCM sekuensial dengan FCM paralel.

3.2 Tahapan Pelatihan

Tahap pelatihan digunakan untuk membuat *cluster* dan nilai derajat keanggotan dari *dataset* yang ada. Dalam tahap pelatihan, algoritma yang digunakan adalah algoritma FCM sekuensial dan FCM Paralel dengan GPU.

a. Data latih disesuaikan dengan skenario pada implementasinya nanti.
b. Membuat nilai derajat keanggotaan awal secara *random* sebagai bobot awal tiap titik data.
c. Jumlah *cluster* ditentukan sesuai keingingan atau kerateristik data.
d. *Centroid* awal dibentuk berdasarkan nilai derajat keanggotaan awal kemudian berubah-ubah nilainya sejalan dengan perbaikan nilai derajat keanggotaan.
e. Parameter kondisi berhenti hanya *epsilon*, sedangkan *fuzziness* adalah nilai pembobot untuk menentukan penentuan nilai derajat keanggotaan, mempengaruhi jarak bobot tiap titik data antara 0 - 1.

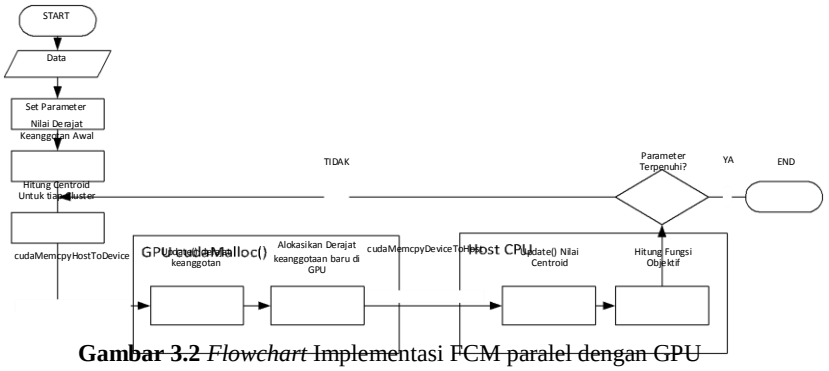
3.3 Tahapan Validasi

Setelah menghasilkan data *clustering* pada tahap pelatihan kemudian nilai derajat keanggotaan dan *centroid* tersebut digunakan untuk tahap validasi hasil *clustering*.

a. Data *clustering* berupa nilai derajat keanggotan dan *centroid* yang sudah melewati ketentuan parameter pada iterasi.
b. Pemilihan *cluster* ditentukan oleh bobot nilai derajat keanggotaan masing-masing tiap titik data.
c. Dari hasil *clustering* dan *running* program di dapat waktu eksekusi dan nilai SSE.

3.4 Implementasi FCM paralel dengan pendekatan GPU

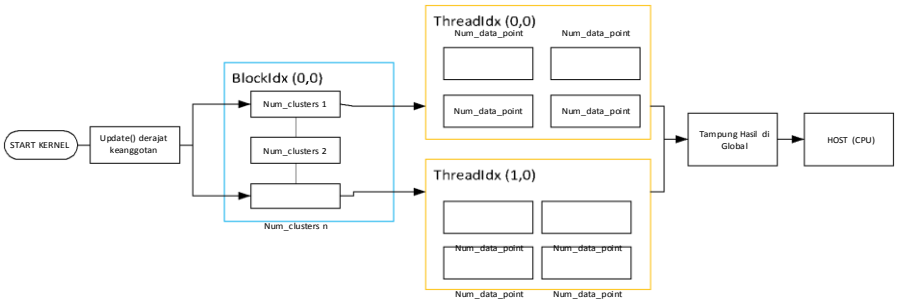
Tahapan dalam mengimplementasikan FCM paralel dengan pendekatan GPU adalah dengan penambahan instruksi pada algoritma FCM sekuensial di suatu fungsi atau prosedur menggunakan CUDA. Pada *Fuzzy C-Means* sekuensial terdapat fungsi untuk memperbarui nilai derajat keanggotaan terus-menerus hingga parameter terpenuhi (2). Dimana fungsi atau prosedur memperbarui nilai derajat keanggotaan ini membutuhkan iterasi yang tidak sedikit dan mempengaruhi proses lainnya seperti penetapan hasil *centroid* dan *clustering* tiap titik data.



Gambar 3.2 *Flowchart* Implementasi FCM paralel dengan GPU

3.5 Model Alokasi Memori GPU

Model alokasi pada sistem menggunakan *thread* dan *block* sebagai alokasi penyimpanan nilai yang diperlukan untuk fungsi memperbarui derajat keanggotaan. Pada fungsi ini jumlah *cluster* (di tentukan di awal sebagai parameter) akan di alokasikan pada bagian *block* di GPU dan tiap titik data di alokasikan pada *thread*. Dimana di dalam satu *block* terdapat ratusan *thread*.



Gambar 3.3 *Flowchart* Model Alokasi di GPU

4. Pembahasan

4.1 Skenario Pengujian

Dataset yang digunakan diambil dari *Kent Ridge Bio-medical Dataset*[12] dan *UCI Machine Learning Repository*[11].

Tabel 4.1 Keterangan *Dataset* Pengujian

No	*Dataset*	*Attribute*	*Record*	Kelas	Keterangan	Size
1	Prostate Cancer	12.600	102	2	Normal, Tumor	3,54 MB
2	Ovarian Cancer	15.154	253	2	Normal, Cancer	32,1 MB

a. *Dataset* Prostate Cancer terdiri dari 12.600 *attribute* dan 102 *record*, dengan karakteristik data berupa *integer* yang menjelaskan nilai dari tiap 12.600 *genes* dalam mengelompokkan *cluster* normal dan tumor.
b. *Dataset* Ovarian Cancer terdiri dari 15.154 *attribute*, dengan karakteristik data berupa angka *range* dari 0 sampai 1 dan menyatakan nilai identitas M/Z spectral, yaitu identitas yang membedakan kelompok kanker ovarium dengan non-kanker.

Skenario pengujian pada penelitian ini adalah sebagai berikut:

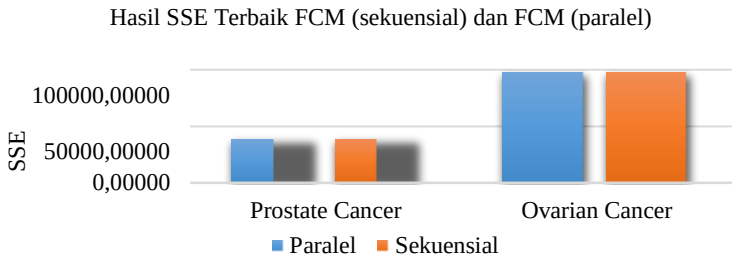
a. Parameter *fuzziness* ($m > 1$): **1.1**; **1.5**; **2**
   Parameter *fuzziness* berfungsi untuk memberikan jarak bobot nilai derajat keanggotaan pada *cluster*. *Fuzzines* menentukan bobot yang dihasilkan akan menghasilkan nilai bobot yang *crisp* atau *fuzzy*.
b. Parameter *epsilon* (): **10^{-1}**; **10^{-3}**; **10^{-5}**
   Epsilon sebagai kondisi iterasi berhenti, dengan penentuan selisih nilai fungsi objektif awal dengan fungsi objektif t+1[7]. *Epsilon* menentukan lama iterasi yang akan dilakukan dan menentukan akurasi yang akan dihasilkan.
c. Jumlah *cluster* (): **2**; **4**

Jumlah *Cluster* yang digunakan diluar dari *dataset*, karena ingin membuktikan dalam membuat *cluster* baru maka perhitungan matriks antara data dengan derajat keanggotaan akan semakin bertambah.

4.2 Analisis dan Hasil Pengujian

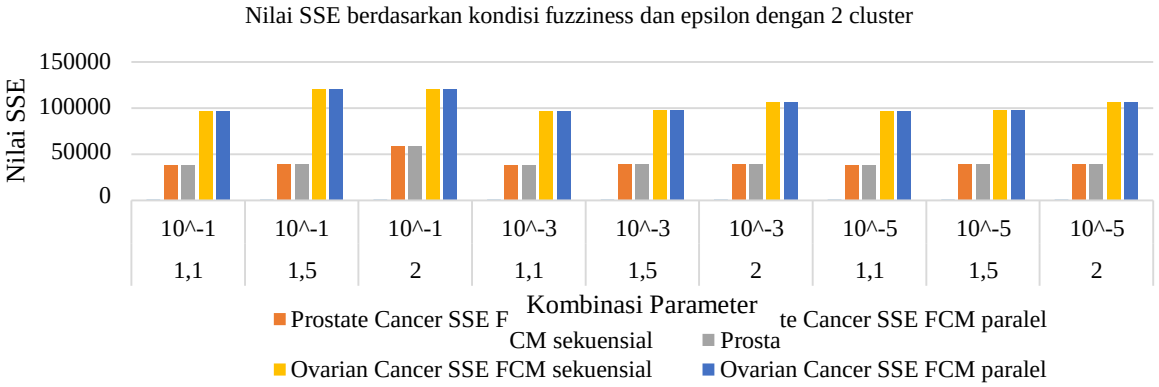
4.2.1 Analisis Evaluasi *Sum Squared of Error* (SSE)

SSE merupakan salah satu metode pengukuran performansi pada *clustering*. Pada pengujian ini SSE yang di hitung merupakan SSE terbaik dari hasil uji tiap *dataset* dengan semua parameter yang ada di skenario pengujian. Berikut adalah SSE terbaik dari setiap dataset yang diuji terlihat pada Gambar 4.1



Gambar 4.2 Hasil SSE Terbaik FCM (sekuensial) dan FCM (paralel)

Dari hasil Gambar 4.1, didapatkan hasil SSE terbaik untuk *dataset* Prostate Cancer adalah 38489,41568 dengan *fuzziness* 1,1 dan *epsilon* 10^{-5} dan pada *dataset* Ovarian Cancer adalah 97271,36719 dengan *fuzziness* 1,1 dan *epsilon* 10^{-3}. Perubahan nilai *fuzziness* dan *epsilon* sama-sama mempengaruhi nilai akhir SSE. *Centroid* dari hasil *clustering* sekuensial dan paralel tidak berbeda. Sehingga menyebabkan SSE dari FCM sekuensial dan FCM paralel juga sama. Hal ini terjadi karena tahapan pada fungsi pengerjaan tidak ada yang berbeda hanya penerapan pada paralelisasi yang mengubah waktu eksekusi.

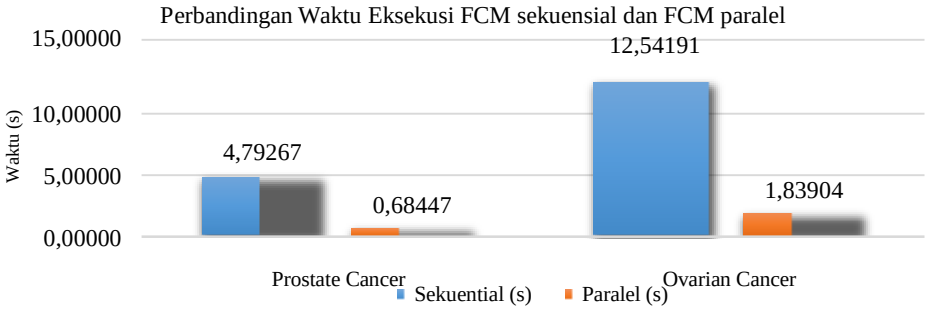


Gambar 4.3 Nilai SSE berdasarkan kondisi *fuzziness* dan *epsilon* dengan 2 *cluster*

Nilai SSE terburuk sama pada kedua *dataset* saat kondisi parameter *fuzziness* 2, *epsilon* 10^{-1}. Ini didapatkan karena iterasi yang berhenti terlalu cepat sebelum optimal, dilihat dari hasil SSE terbaik saat kondisi berhenti antara 10^{-3} dan 10^{-5} yang tidak terlalu jauh berbeda.

4.2.2 Analisis Waktu

Waktu eksekusi merupakan indikator penting yang di analisis pada penelitian tugas akhir ini. Dalam penilitian ini, akan dilakukan pengujian perbandingan antara waktu sekuensial dan paralel. Hasil perbandingan eksekusi waktu diambil berdasarkan nilai SSE terbaik dari setiap parameter menyatakan klasifikasi pada *dataset* yaitu dengan jumlah *cluster* 2.



Gambar 4.4 Perbandingan Waktu Eksekusi FCM sekuensial dan FCM paralel

Dari Gambar 4.3 dilihat bahwa pada data Ovarian Cancer memiliki waktu eksekusi paling lama dibanding Prostate Cancer. Ini disebabkan karena jumlah *attribute* dan *record* pada Ovarian Cancer merupakan data dengan jumlah *attribute* terbanyak dari *dataset*. Pada percobaan berikutnya karena setiap percobaan nilai waktu eksekusinya tidak fluktuatif atau berubah secara segnifikan antara proses pertama dan pengulangan proses kedua, maka analisis perbandingan waktu dapat digunakan.

Table 4.2 Perbandingan Waktu Eksekusi FCM sekuensial dan FCM paralel

Dataset	FCM Sekuensial (s)	Keterangan	FCM Paralel (s)	Keterangan	Selisih Waktu (s)	*speedup*
Prostate Cancer	4,79267	m = 1,1 ; e = 10^{-5}	0,68447	m = 1,1 ; e = 10^{-5}	4,10820	7,00205
Ovarian Cancer	12,54191	m = 2 ; e = 10^{-5}	1,83904	m = 2 ; e = 10^{-5}	10,70287	6,81982

Kemudian dapat dilihat juga pada Tabel 4.2 bahwa waktu eksekusi FCM sekuensial memiliki waktu lebih lama dibandingkan dengan waktu eksekusi pada FCM paralel. Hal tersebut dikarenakan penggunaan GPU pada FCM paralel. Perbandingan waktu tersebut diambil berdasarkan nilai SSE terbaik.

4.2.3 Analisis Performansi

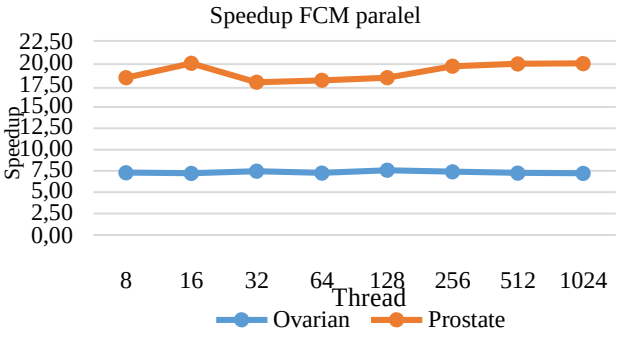
Menggunakan pendekatan GPU memang mengurangi waktu eksekusi yang dibutuhkan pada proses *clustering* data. Di sisi lain penggunaan GPU membutuhkan biaya komputasi yang besar baik penyediaan *hardware* yang tidak murah untuk GPU *high-end* maupun ketersedian *resource* lain yang diperlukan. Evaluasi performansi pada penelitian ini meliputi *Speedup*, *Performance Improvement* dan Efisiensi. *Speedup* berfungsi untuk menghitung seberapa cepat waktu yang dihasilkan oleh algoritma FCM paralel dibandingkan algoritma FCM sekuensial. *Performance Improvement* berfungsi untuk menggambarkan hubungan antara algoritma FCM sekuensial dengan algoritma FCM paralel, apakah memiliki kelebihan ataupun sebaliknya. Efisiensi digunakan untuk mengukur berapa jumlah *thread* yang efisien agar waktu eksekusi yang dibutuhkan lebih sedikit dengan biaya komputasi juga leboh sedikit.

Penjelasan mengenai *speedup*, *performance improvement* dan efisiensi dijelaskan lebih lanjut pada Gambar 4.4, 4.5 dan 4.6.

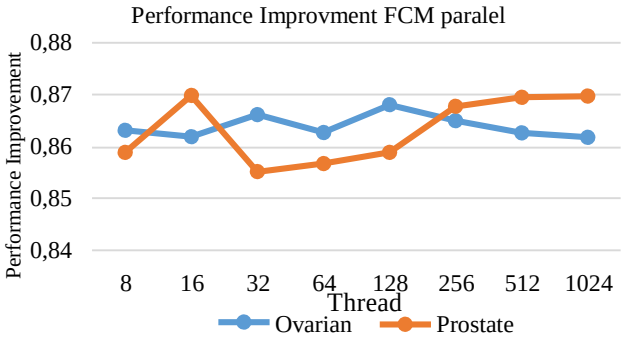
Dari beberapa percobaan dengan menggunakan data dan jumlah *thread* yang berbeda, dihasilkan bahwa *speedup* tertinggi pada *dataset* Ovarian Cancer didapat dengan menggunakan 128 *thread*. Sedangkan untuk *dataset* Prostate Cancer didapat dengan menggunakan 16 *thread*.

Performance Improvement pada *dataset* Ovarian Cancer dengan *thread* dari 8 sampai 128 memiliki perbedaan yang fluktuatif. Tetapi nilai *Performance Improvement* tertinggi pada Ovarian Cancer terdapat saat menggunakan 128 *thread*. Jika lebih dari itu maka tingkat *Performance Improvement* akan menurun atau tidak signifikan lagi pengaruh jumlah *thread*-nya. Pada Prostate Cancer *Performance Improvement* terbaik di dapat pada jumlah 16 *thread*, tetapi dilihat dari *trend* penggunaan *thread* lebih baik menggunakan 128 *thread* untuk mendapatkan pengaruh yang signifikan karena nilainya yang selalu naik hingga 1024 *thread*.

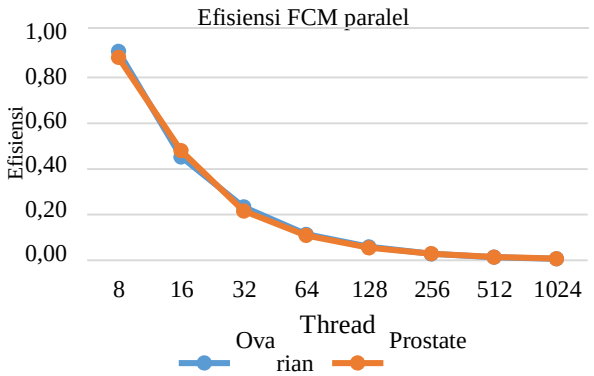
Efisiensi pada *dataset* Ovarian Cancer dan Prostate Cancer adalah jumlah *thread* sama dengan 8 memiliki efisiensi paling tinggi. Tetapi tidak menjadi kriteria terbaik untuk menggunakan jumlah *thread*.



Gambar 4.4 Speedup FCM paralel



Gambar 4.5 Performance Improvment FCM paralel



Gambar 4.7 Efisiensi FCM paralel

5. Kesimpulan

Kesimpulan yang didapatkan dari hasil analisis pada penelitian ini adalah :

a. Cara mengimplementasikan algoritma *Fuzzy C-Means* dengan pendekatan GPU adalah dengan mengubah alokasi memori GPU dengan perintah-perintah CUDA untuk menampung hasil perhitungan sementara secara paralel dan merubah iterasi perhitungan di dalam fungsi memperbarui nilai derajat keanggotaan dengan CUDA.

b. Setelah dilakukan analisis evaluasi, waktu dan performansi, terlihat bahwa untuk data Ovarian Cancer didapatkan hasil SSE terkecil 97271,36719 dengan *fuzziness* 1.1, *epsilon* 10^{-3} dan *cluster* 2. Untuk perbandingan waktu eksekusi didapat waktu 12,54191 detik untuk FCM sekuensial dan 1,83904 detik untuk FCM paralel dan selisih waktu 10,70287 detik dengan kondisi parameter yang sama.
Untuk *speedup* , *performance improvement* dan efisiensi pada dataset Ovarian Cancer, *speedup* tertinggi didapat pada jumlah 128 *thread* dengan nilai 7,58002, *performance improvement* sebesar 0,86807 dan efisiensi dengan 8 *thread* dengan nilai 0,91435. Untuk data Prostate Cancer didapatkan hasil SSE terkecil 38489,41568 dengan *fuzziness* 1.1, *epsilon* 10^{-5} dan *cluster* 2. Untuk perbandingan waktu eksekusi didapat waktu 4,79267 detik untuk FCM sekuensial dan 0,68447 detik untuk FCM paralel dan selisih waktu 4,10820 detik dengan kondisi parameter yang sama.
Untuk *speedup* , *performance improvement* dan efisiensi pada dataset Prostate Cancer, *speedup* tertinggi didapat pada jumlah 16 *thread* dengan nilai 20,09944, *performance improvement* sebesar 0,86980 dan efisiensi dengan 8 *thread* dengan nilai 0,88727.

c. Untuk pengaruh parameter *fuzziness*(m), semakin tinggi nilainya maka semakin tinggi pula nilai SSE dan waktu eksekusinya, namun itu juga dipengaruhi parameter kondisi berhenti (e). Untuk *epsilon*(e) semakin kecil nilainya, maka semakin lama iterasi karena proses terus mencari selisih nilai fungsi objektif hingga mencapai nilai yang di tentukan. Untuk *cluster*(c) membuktikan bahwa semakin banyak permintaan *cluster*, maka waktu eksekusi akan semakin lama dan nilai SSE yang didapat semakin baik berdasarkan hasil pengujian.

d. Dari hasil SSE dan waktu eksekusi dari setiap *dataset*, maka Ovarian Cancer memiliki karateristik data yang menyebar dimana akan berpengaruh pada nilai SSE dan waktu eksekusi yang dihasilkan akan lebih besar dan lebih lama dibandingkan dengan nilai SSE dan waktu eksekusi pada Prostate Cancer yang lebih rendah dan lebih cepat nilainya, dikarenakan sifatnya yang menyatu dimana titik datanya yang lebih dekat satu sama lainya.

Daftar Pustaka :

[1] Ahmad Firdaus Ahmad Fadzil, Noor Elaiza Abdul Khalid, dan Mazani Manaf. (2011), *Scaling Perormance of Task-Intensive Applications via Mapreduce Parallel Processing*, Faculty of Computer anda Mathematical Science, UiTM Shah Alam, Selangor, Malaysia.

[2] Arul, S., Dash, M., and Tue, M., 2008. *Graphics Hardware based Efficient and Scalable Fuzzy C-Means Clustering*, Singapore, Nanyang Technological University.

[3] Bezdek, J. C., Cannon, R. L., Dave, J. V., 1986. *Efficient Implementation of the Fuzzy c-Means Clustering Algorithms*, IEEE.

[4] Böhm, Christian, Noll, Robert, Plant, Claudia, Wackersreuther, B 2009, *Density-based Clustering using Graphics Processors*, University of Munich, Munich, Germany.

[6] Harvey, Jesse Patrick, "*GPU acceleration of object classification algorithms using NVIDIA CUDA*" (2009). Thesis. Rochester Institute of Technology.

[7] Kusumadewi, S., Purnomo, H., 2010. *Aplikasi Fuzzy Untuk Pendukung Keputusan*, Graha Ilmu Jakarta.

[8] Nugraheni, Yohana, 2011. *Data Mining Dengan Metode Fuzzy Untuk Customer Relationship Management (CRM) Pada Perusahaan Retail*.

[9] Sayekti, E.R., Hidayat, N., Soebroto, A.A. *Implementasi Algoritma Fuzzy C-Mean Untuk Pembangkitan Aturan Fuzzy Pada Pengelompokan Tingkat Risiko Penyakit Kangker Payudara*. Program Studi Informatika/ Ilmu Komputer.

[10] Tan, P., Steinbach, Kumar. 2006. *Introduction to Data Mining*. Boston: Pearson Education.

[11] Dataset yang digunakan untuk penelitian ini, diambil dari situs (*https://archive.ics.uci.edu/ml/index.html*, diakses tanggal 24 Juli 2015)

[12] Dataset yang digunakan untuk penelitian ini, diambil dari situs (*http://datam.i2r.a-star.edu.sg/datasets/krbd/index.html*, diakses tanggal 30 Juli 2015)