Analisis dan Implementasi Algoritma CLICK dalam Melakukan *Cluster*ing Data Gen Hewan dan Tumbuhan

CLICK Algorithm Analysis and Implementation in Performing Animal and Plant Genes Data Clustering

Dewi Manshuroh[1]	Sri Widowati, MT.[2]	Ade Romadhony, ST. [3]
(113070368)	(93670092-1)	(99750187-1)

Fakultas Informatika Institut Teknologi Telkom Bandung

fyr_fourthelement@yahoo.com[1]	swd@ittelkom.ac.id[2]	ade.romadhony@gmail.com[3]

Abstrak

Teknologi yang semakin maju memungkinkan para peneliti dalam melakukan ekstraksi level ekspresi gen dalam berbagai jenis percobaan. Hal ini memudahkan untuk mempelajari lebih lanjut mengenai fungsi gen, jaringan gen, proses biologi dan efek dari suatu perlakuan medis. Langkah pertama untuk melakukannya adalah dengan melakukan *clustering* gen. Saat ini sudah banyak sekali algoritma *clustering* yang dibuat namun baru sedikit algoritma yang dikhususkan pada *clustering data* gen. CLICK atau *Cluster Identification via Connectivity Kernel* merupakan salah satu algoritma yang bertujuan untuk melakukan *clustering data* gen.

Algoritma ini berbasis graf berbobot dan menggunakan penghitungan probabilitas sehingga diharapkan dapat memberikan hasil yang lebih akurat. Dalam tugas akhir ini dilakukan implementasi dan analisis performansi algoritma CLICK dalam melakukan *clustering data* gen yang bersifat homogen ataupun heterogen yang kemudian akan diuji melalui metode *Figure Of Merit* (FOM), serta untuk mencari tahu apakah faktor sifat homogen atau heterogen suatu data akan mempengaruhi nilai FOM dari data tersebut. Selain itu dalam penelitian ini dilakukan perbandingan nilai FOM yang dihasilkan CLICK dengan nilai FOM yang dihasilkan algoritma K-Means, GenClust dan Cast dalam melakukan *clustering data* yang sama. Sehingga memudahkan para peneliti dalam memilih algoritma yang akan digunakan untuk melakukan *clustering* data gen yang memiliki sifat-sifat tersebut.

Dari hasil penelitian yang dilakukan dihasilkan kesimpulan bahwa untuk jumlah eksperimen yang sama, algoritma CLICK memiliki performansi yang lebih baik dalam melakukan *clustering* data gen yang bersifat lebih heterogen daripada data homogen setelah dibandingkan dengan ketiga algoritma lainnya. Sedangkan nilai FOM tidak dipengaruhi oleh sifat heterogen atau homogen suatu data melainkan oleh rata-rata dari cluster yang dihasilkan serta jumlah eksperimen yang diujikan.

Kata Kunci : *CLICK, clustering, gen*

Abstract

The more advanced technology that allows researchers in conducting the extraction level of gene expression in different types of experiments. This makes it easy to learn more about gene function, gene networks, biological processes and effects of a medical treatment. The first step to do this is to perform gene clustering. We have had a lot of clustering algorithms that are made but very few clustering algorithm that is specific to the gene data clustering. CLICK or Cluster Identification via Connectivity Kernels is one of algorithms that aims to perform data clustering of genes.

The algorithm is based on a weighted graph and use the calculation of probability that is expected to provide more accurate results. In this thesis performed the implementation and performance analysis of algorithms CLICK in performing data clustering of genes that are homogeneous or heterogeneous which will then be tested through the method of Figure Of Merit (FOM), as well as to find out whether the factor of a homogeneous or heterogeneous nature of the data will affect the value of FOM from these data. Also in this study the FOM value of the CLICK algorithm will be compared with FOM value generated by K-Means, GenClust and Cast algorithm in doing the same data clustering. Making it easier for the researchers in selecting the algorithm that will be used to perform data clustering of genes that have these character.

From the results of research conducted a conclusion that for the same amount of experiment, CLICK algorithm has a better performance in doing data clustering of genes that are more heterogeneous than homogeneous data when compared with the three other algorithms. While the FOM value is not affected by the characters of a heterogeneous or homogeneous data but by the average of the resulting clusters and the number of experimentally tested.

Keywords: *CLICK, clustering, genes*

1. Pendahuluan

1.1 Latar Belakang

Teknologi *Microarraay* DNA telah memudahkan para peneliti dalam melihat *level* ekspresi dari gen dalam berbagai kondisi ataupun proses. Teknologi baru ini memungkinkan perolehan *data* yang terbukti mendasar, mulai dari pemahaman tentang sistem biologis kompleks untuk diagnosis klinis hingga fungsi dari gen itu sendiri. Langkah pertama untuk melakukan analisis dari data ekspresi gen tersebut adalah *clustering*. Dari proses *clustering* dapat diperoleh berbagai informasi seperti pemahaman tentang fungsi dan jaringan gen, membantu diagnosa dari suatu kondisi penyakit serta mengetahui efek dari suatu perlakuan medis terhadap suatu gen.

Clustering adalah metode penganalisaan *data* yang bertujuan untuk mengelompokkan *data* dengan karakteristik yang sama ke dalam suatu kelas dan *data* dengan karakteristik yang berbeda ke kelas yang lain. Dalam *clustering data* gen dilakukan proses penghitungan tingkat kesamaan dari tiap gen berdasarkan *level* ekspresi gen kemudian dilakukan pengelompokan menurut kesamaan tersebut. Kesulitan yang sering ditemui dalam *clustering data* gen ini adalah tingkat akurasi yang rendah dalam melakukan pengelompokan gen sehingga sering terjadi pengelompokan dimana terdapat gen yang tidak seharusnya masuk ke dalam suatu kelas tertentu. Sudah banyak algoritma *clustering* yang dibuat guna mencari solusi dari masalah tersebut, seperti algoritma K-*Mean*s dan GenClust. Namun masih sering terdapat kesalahan dalam melakukan *clustering data* gen dan tingkat akurasi yang rendah pada saat menghadapi kasus tertentu terutama saat menghadapi *data* yang bersifat heterogen ataupun homogen.

CLICK, atau *Cluster Identification via Connectivity Kernel* merupakan algoritma yang juga dibuat untuk mengatasi masalah *clustering data* gen. Algoritma ini dikatakan memiliki kinerja yang baik dalam melakukan *cluster*ing data gen dalam berbagai kondisi. Dalam tugas akhir ini dilakukan analisis mengenai kinerja algoritma CLICK dalam menghadapi data yang memiliki sifat heterogen dan homogen dengan menggunakan pengukuran kekuatan prediksi algoritma menggunakan *Figure Of Merit* (FOM).

1.2 Perumusan Masalah

Permasalahan yang menjadi objek dari penelitian tugas akhir ini, terdiri atas :

1. Bagaimana implementasi algoritma CLICK dalam melakukan *clustering data* gen hewan dan tumbuhan?
2. Bagaimana performansi algoritma CLICK dalam melakukan *clustering data* gen hewan dan tumbuhan yang memiliki sifat heterogen dan homogen jika diukur dari kekuatan prediksinya?
3. Bagaimana hubungan antara sifat heterogen dan homogen suatu data dengan nilai FOM dari data tersebut setelah dilakukan *clusterisasi*?

Batasan masalah dalam tugas akhir ini adalah:

1. Aplikasi yang digunakan untuk mengimplementasikan algoritma CLICK adalah aplikasi Matlab.
2. *Data* yang dipakai untuk menjadi pengujian dalam sistem ini adalah *data* gen hewan dan tumbuhan yang telah disediakan dalam *website* penelitian bioinformatics [1].
3. Data yang dipakai berjumlah minimal 16 buah gen.

1.3 Tujuan

Tujuan yang ingin dicapai dalam tugas akhir ini yaitu:

1. Mengimplementasikan algoritma CLICK menggunakan MATLAB dan melakukan analisis performansinya.
2. Melakukan pengukuran kekuatan prediksi algoritma CLICK dalam melakukan *clustering data* gen yang bersifat homogen dan heterogen dengan menggunakan FOM.
3. Mengetahui hubungan antara sifat heterogen dan homogen suatu data dengan nilai FOM data tersebut setelah dilakukan *clusterisasi*.

2. Dasar Teori

1.1 2.1 *Clustering*

Clustering adalah metode penganalisaan *data* yang tujuannya adalah untuk mengelompokkan *data* dengan karakteristik yang sama ke suatu *cluster* yang sama dan *data* dengan karakteristik yang berbeda ke *cluster* yang lain. Tujuan dari *clustering* ini adalah untuk memenuhi dua kriteria berikut, yaitu *Homogeneity* dimana tiap elemen yang berada dalam satu kelas memiliki tingkat kesamaan yang tinggi satu sama lain dan *Separation* dimana elemen-elemen dari kelas yang berbeda memilik tingkat kesamaan yang rendah satu sama lain [7]. Ada beberapa pendekatan yang digunakan dalam mengembangkan metode *clustering* yaitu *clustering* dengan pendekatan partisi dan *clustering* dengan pendekatan hirarki. *Clustering* dengan pendekatan partisi atau sering disebut dengan *partition-based clustering* mengelompokkan *data* dengan memilah-milah *data* yang dianalisa ke dalam *cluster-cluster* yang ada. *Clustering* dengan pendekatan hirarki

atau sering disebut dengan *hierarchical clustering* mengelompokkan *data* dengan membuat suatu hirarki berupa dendogram dimana *data* yang mirip akan ditempatkan pada hirarki yang berdekatan dan yang tidak pada hirarki yang berjauhan. Selain kedua pendekatan di atas, ada juga *clustering* dengan pendekatan *automatic mapping* (*Self-Organising Map*/SOM). *Self-Organising Map* (SOM) merupakan suatu tipe *Artificial Neural Networks* yang di-*training* secara *unsupervised*.

Clustering data gen merupakan suatu metode untuk mengelompokan gen-gen yang memiliki *level* ekspresi mRNA yang memilki kesamaan tinggi ke dalam *cluster-cluster*. *Level* ekspresi mRNA ini diperoleh dengan mengukur kadar mRNA dengan berbagai macam teknik, misalnya dengan *microarray* atau pun *Serial Analysis of Gene Expression* atau Analisis Serial Ekspresi Gen (SAGE). Teknik-teknik tersebut umumnya diterapkan pada analisis ekspresi gen skala besar yang mengukur ekspresi banyak gen (bahkan genom) dan menghasilkan *data* skala besar. Setelah dilakukan perhitungan *level* ekspresi mRNA tersebut, kemudian dilakukan pengujian dengan berbagai kondisi pada ekspresi gen tersebut sehingga diperoleh nilai *vector* yang nantinya akan diproses untuk menentukan kelas manakah gen tersebut berasal.

2.2 Algoritma CLICK

Algoritma CLICK merupakan algoritma yang berbasis graf terhubung dimana setiap *data* gen merupakan elemen atau *vertex* pada graf dan nilai *similarity* antar *data* gen sebagai *edge* nya [6]. Dua buah elemen yang masuk ke dalam satu *cluster* yang sama disebut *mates*, sedangkan dua buah elemen yang masuk ke dalam *cluster* yang berbeda disebut non-*mates*. Algoritma ini menerima *input* dalam dua bentuk, yang pertama adalah *Fingerprint Data*, yaitu dimana tiap elemen memiliki nilai vektor yang riil contohnya seperti *data level* ekspresi gen pada tiap eksperimen yang berbeda. Yang kedua adalah *Similarity Data* yang seperti namanya merupakan *data* dari tingkat kesamaan tiap-tiap elemen, *data* ini berbentuk matrix m x m. Untuk memperoleh nilai dari *Similarity data*, *Fingerprint Data* dihitung menggunakan *Euclidean Distance* dengan rumus sebagai berikut :

$$s(x,y) = \sqrt{\sum_{i=1}^{n}(x^i - y^i)^2} \quad (2.1)$$

Dimana s adalah nilai *similarity* antara gen x dan y sedangkan i adalah urutan eksperimen yang diujikan.

Pada algoritma CLICK diasumsikan bahwa nilai kesamaan antara *mates* terdistribusi normal dengan *mean* μ_T dan variansi σ_T^2 sedangkan nilai ketidaksamaan antara non-*mates* terdistribusi normal dengan *mean* μ_F dan variansi σ_F^2 dimana $\mu_T > \mu_F$. Selain μ_T , σ_T^2, μ_F, dan σ_F^2 juga terdapat parameter p_{mates} yang merupakan nilai probabilitas dua buah elemen yang diambil secara *random* merupakan *mates*. Langkah pertama yang dilakukan pada algoritma ini adalah mengestimasi nilai dari lima parameter di atas, yaitu μ_T , σ_T^2, μ_F, σ_F^2 dan p_{mates} dengan menggunakan algoritma EM. Kemudian dilakukan penghitungan bobot yang akan menjadi nilai dari tiap *edge* dengan rumus berikut

$$wij = \log \frac{pmates\,\sigma F}{(1-pmates)\sigma T} + \frac{(Sij-\mu F)^2}{2\sigma F^2} - \frac{(Sij-\mu T)^2}{2\sigma T^2} \quad (2.2)$$

Langkah selanjutnya yaitu secara rekursif melakukan pemotongan graf dan pemeriksaan apakah graf sudah memenuhi kriteria yang diinginkan. Pemotongan graf dilakukan dengan menggunakan metode Min*Weight*Cut. Hasilnya merupakan partisi-partisi graf yang dapat terdiri dari beberapa *vertex* ataupun yang terdiri atas satu *vertex* saja (*singleton*). Setelah terbentuk partisi-partisi graf, dilakukan proses untuk menghilangkan *edge* yang memiliki bobot negatif , proses untuk menghilangkan *vertex* yang memiliki derajat kecil serta Basic CLICK pada setiap partisi nya.

Basic-CLICK(*G*):
If *V* (*G*) = *{v}* **then**
move *v* to the *singleton* set *R*.
Else if *G* is a kernel **then**
Output *V* (*G*).
Else
(*H, H'*) ←*MinWeightCut*(*G*).
Basic-CLICK(*H*).
Basic-CLICK(*H'*).

Kemudian dilakukan proses adopsi dimana *cluster-cluster* yang terbentuk dari partisi-partisi graf setelah melalui beberapa proses tadi melakukan pengadopsian terhadap *list singleton* yang ada jika memenuhi syarat. Proses selanjutnya adalah menggabungkan antar *cluster* dengan menggunakan syarat tertentu dan terakhir adalah proses adopsi lagi terhadap *list singleton* yang tersisa.

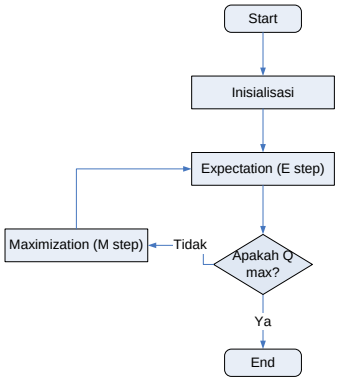
2.3 Algoritma EM (Expectation-Maximization)

Algoritma EM merupakan sebuah metode yang berguna dalam permasalahan *data* yang tidak lengkap (*incomplete data*). Algoritma ini dilakukan secara iterasi dimana dalam setiap iterasi terdapat 2 tahap, yaitu tahap Ekspektasi (E step) dan tahap Maksimisasi (M step). Algoritma EM ini hampir mirip dengan pendekatan *ad hoc* untuk proses estimasi dengan missing *data* yaitu (1) mengganti *missing value* dengan *estimated value*, (2) mengestimasi parameter, (3) mengestimasi ulang *missing value* tadi dengan menggunakan parameter baru yang diestimasi, (4) mengestimasi ulang parameter, dan seterusnya berulang-ulang

sampai dengan konvergen terhadap suatu nilai. Dalam tugas akhir ini setiap iterasi pada algoritma EM bertujuan untuk memaksimalkan fungsi :

$$Q(\theta|\theta^r) = \Sigma_{i<j} \Sigma^{1}_{t=0} g_t(S_{ij}) \log(f_t(S_{ij})p_t) \quad (2.3)$$

Dimana θ adalah set dari model parameter yang akan dicari, $\theta = \{ \mu_T, \mu_F, \sigma_T, \sigma_F, p_{mates} \}$, θ^r merupakan set model parameter pada iterasi ke r. S_{ij} adalah nilai *similarity* dari *edge* i dan j, f_0 dan f_1 sebagai PDF (*Probability Density Function*) dari *non-mates* dan *mates*, sedangkan $p_1 \equiv p_{mates}$ dan $p_0 \equiv 1 - p_1$. Sebelum memasuki algoritma EM, dilakukan inisialisasi model parameter terlebih dahulu.



Gambar 2-1: Proses Algoritma EM

2.3.1 Inisialisasi

Dalam tahap inisialisasi dilakukan penghitungan guna mendapatkan nilai awal yang akan menjadi *input* dari algoritma EM. Parameter pertama yang diestimasi yaitu p_{mates} yang merupakan nilai probabilitas dua buah elemen yang diambil secara *random* merupakan *mates*, $p_{mates} = 0.5$. Kemudian dilakukan penghitungan

$$mean(m) = (\Sigma_{i=1}^{n} \Sigma_{j=1}^{n} s_{ij}) / n \quad (2.4)$$
$$variansi(v) = \Sigma_{i=1}^{n} \Sigma_{j=1}^{n} (s_{ij}-m)^2 \quad (2.5)$$

Untuk inisialisasi nilai μ_T, μ_F, σ_T dan σ_F (asumsi $\sigma_T = \sigma_F = \sigma$) dilakukan ekstraksi dari persamaan berikut :

$$m = p_{mates}*\mu_T + (1 - p_{mates})*\mu_F \quad (2.6)$$
$$v = p_{mates}*(1 - p_{mates})*(\mu_T - \mu_F)^2 + \sigma^2 \quad (2.7)$$

Dari hasil tersebut dilakukan perbandingan dengan distribusi empiris yang kemudian diambil kombinasi yang terbaik untuk menjadi *input* algoritma EM.

2.3.2 Expectation (E Step)

Pada tahap *Expectation* (E step) dilakukan penghitungan f_0 dan f_1, dimana

$$f_0 = (1/ (\sqrt{2\pi\sigma_F}))*e^{(-(sij- \mu F)/2\sigma F\hat{}2)} \quad (2.8)$$
$$f_1 = (1/ (\sqrt{2\pi\sigma_T}))*e^{(-(sij- \mu T)/2\sigma T\hat{}2)} \quad (2.9)$$

Kemudian dilakukan penghitungan ekspektasi dan penghitungan nilai Q dimana $g_1 \equiv E$, $g_0 = 1 - g_1$ dan $t \equiv 0,1$ dengan rumus sebagai berikut :

$$E = (f_1 * s_{ij} * p_1) / f_1 * s_{ij} * p_1 + f_1 * s_{ij} * p_1 \quad (2.10)$$
$$Q(\theta|\theta^r) = \Sigma_{i<j} \Sigma^{1}_{t=0} g_t(S_{ij}) \log(f_t(S_{ij})p_t) \quad (2.11)$$

2.3.3 Maximization (M step)

Tahap *Maximization* (M step) bertugas menghitung parameter yang optimal untuk iterasi selanjutnya dengan menggunakan fungsi berikut :

$$\mu_t = \Sigma_{i<j} (g_t(S_{ij})S_{ij}) / \Sigma_{i<j} g_t(S_{ij}) \quad (2.12)$$
$$\sigma^2_T = \Sigma_{i<j} (g_t(S_{ij})(\mu_t - S_{ij})) / \Sigma_{i<j} g_t(S_{ij}) \quad (2.13)$$
$$p_t = \Sigma_{i<j} g_t(S_{ij}) / \Sigma_{i<j} 1 \quad (2.14)$$

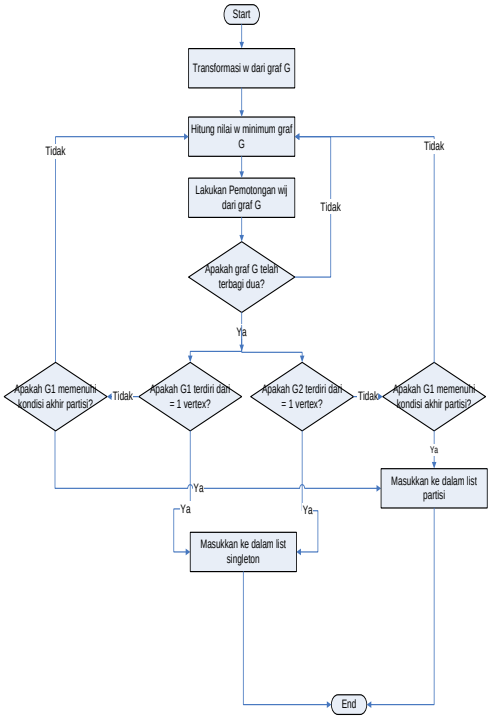
2.4 Algoritma *MinWeightCut*

Algoritma ini merupakan inti dari CLICK, dimana dilakukan pemotongan graf secara rekursif untuk memperoleh partisi-partisi graf yang selanjutnya akan diproses menjadi *cluster-cluster*. Permasalahan yang sering dihadapi dalam melakukan pemotongan graf adalah adanya bobot *edge* yang bernilai negatif. Oleh karena itu dalam algoritma ini dilakukan penanggulangannya. Adapun keseluruhan langkah dari algoritma *MinWeightCut* adalah sebagai berikut :

- Hitung w_{max} atau bobot *edge* maksimal.
- Transformasikan w dengan menggunakan rumus

$$f(w) = w' = w_{max} - w + \varepsilon, \ \varepsilon > 0 \quad (2.15)$$

- Hitung nilai w' yang paling minimal.
- Lakukan pemotongan *edge* yang memiliki nilai w' yang paling minimal secara rekursif hingga graf terbagi dua.
- Lakukan dua buah pemeriksaan, yaitu:
  - Apakah partisi yang terbentuk merupakan *singleton* (terdiri dari sebuah node) atau tidak, jika ya maka masukkan node ke dalam set *singleton*.
  - Jika partisi yang terbentuk bukan *singleton*, apakah partisi tersebut telah memenuhi kondisi akhir (W>0) dan memenuhi jumlah maksimal *cluster*, jika ya maka masukkan ke dalam list partisi, jika tidak maka lakukan pemotongan secara rekursif kembali.



Gambar 2-2: Proses Algoritma Min*Weight*Cut

2.5 Figure of Merit (FOM)

Figure of Merit atau FOM merupakan suatu metode yang digunakan untuk mengukur seberapa kuat daya prediksi dari suatu algoritma *clustering*. FOM digunakan dalam tugas akhir ini karena kemampuannya dalam mengukur solusi yang dihasilkan suatu algoritma *clustering* walaupun solusi *clustering* yang sebenarnya tidak diketahui [3,4]. Dalam melakukan pengukuran pada tugas akhir ini digunakan 2-norm FOM yang memiliki rumus penghitungan sebagai berikut :

$$FOM(e) = \Sigma^{m}_{e=1} (FOM(e,k)) \quad (2.16)$$

Dimana :

$$FOM(e,k) = \sqrt{[(1/n) * \Sigma_{i=0}^{k-1} \Sigma_{x \epsilon C} (R(x,e) - m_i(e))^2]} \quad (2.17)$$

Keterangan :

R = Fingerprint *Data* (matrix m x n)
e = Kondisi eksperimen yang dipilih
k = Jumlah *cluster* yang terbentuk
R(x,*e*) = Nilai level ekspresi gen x pada kondisi *e*
C = *Cluster*
m_i (*e*) = Rata level ekspresi gen pada kondisi *e* dalam *cluster i*

Metode FOM biasa digunakan untuk melakukan perbandingan antara beberapa algoritma sehingga dapat dilakukan pemilihan algoritma yang tepat untuk sebuah dataset terterntu [2]. Semakin rendah nilai FOM yang diberikan pada suatu algoritma *clustering*, semakin bagus daya prediksi algoritma tersebut.

3. Pengembangan Sistem

3.1 Tujuan Pembangunan Sistem

Sistem yang dibangun ini bertujuan untuk melakukan proses klasterisasi yang lebih mengkhususkan pada *data-data* gen makhluk hidup. Proses *clustering* dalam sistem ini dilakukan dengan menggunakan algoritma CLICK yang berbasis kepada probabilitas dan konektivitas antar *vertex* dalam graf, dimana *vertex* disini merupakan elemen *data* gen yang akan di *clustering*. Dengan adanya penghitungan probabilitas diharapkan bahwa hasil *clustering* dari algoritma ini menjadi lebih akurat. Selain itu sistem ini juga bertujuan untuk mengetahui performansi algoritma CLICK dalam melakukan *cluster*ing data gen yang bersifat heterogen dan homogen sehingga dapat digunakan dalam kasus *clustering* lainnya dengan lebih spesifik dan sesuai untuk ke depannya.

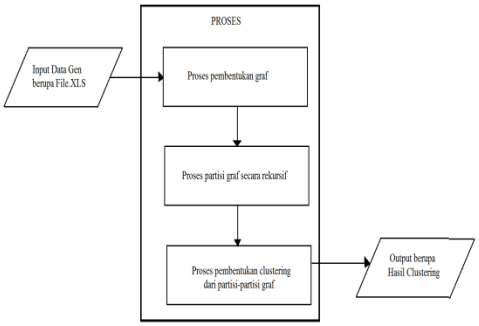
3.2 Deskripsi dan Pemodelan Sistem

Sistem yang dibangun pada tugas akhir ini adalah sistem yang menggunakan algoritma CLICK (*Cluster*ing Identification via Connectivity Kernel) dan bertujuan untuk melakukan *clustering* terhadap *data* gen. *Input* dari sistem ini berupa *fingerprint data* gen. Dari *input* tersebut dilakukan *preproccessing* untuk menghasilkan *similarity data*. Kemudian dilakukan estimasi parameter sistem, yaitu σ^2_T, σ^2_F, μ_T, μ_F, dan P_{mates}. Proses estimasi parameter ini dilakukan dalam dua tahap yang meliputi tahap Inisialisasi parameter dan tahap Estimasi parameter dengan menggunakan algoritma EM (*Expectation-Maximization*).

Selanjutnya dilakukan penghitungan bobot *edge* yang kemudian akan digunakan untuk menentukan *edge* manakah yang harus dihilangkan dari graf. Proses menghilangkan *edge* dari graf ini menggunakan algoritma MinWeightCut dan dilakukan secara rekursif hingga terbentuk *list* yang terdiri dari partisi-partisi graf. Kemudian untuk setiap parisi graf yang dihasilkan dilakukan proses untuk menghilangkan *edge*s yang bernilai negatif, proses untuk memisahkan *vertex* dengan derajat kecil dari graf, serta pengaplikasian algoritma BasicCLICK(C). Hasilnya adalah *list cluster*, sedangkan *vertex* yang terpisah dan menjadi *singleton* dimasukkan dalam *list* tersendiri.

Proses selanjutnya adalah melakukan adopsi dari list *singleton* oleh yang ada dengan syarat jumlah bobot *edge* yang menghubungkan *singleton* ke suatu *cluster* > 0. Kemudian dengan syarat yang sama dilakukan penggabungan antar *cluster* dan terakhir dilakukan adopsi *singleton* yang tersisa. Sistem dalam tugas akhir ini dibangun dengan menggunakan bahasa pemrograman MATLAB.

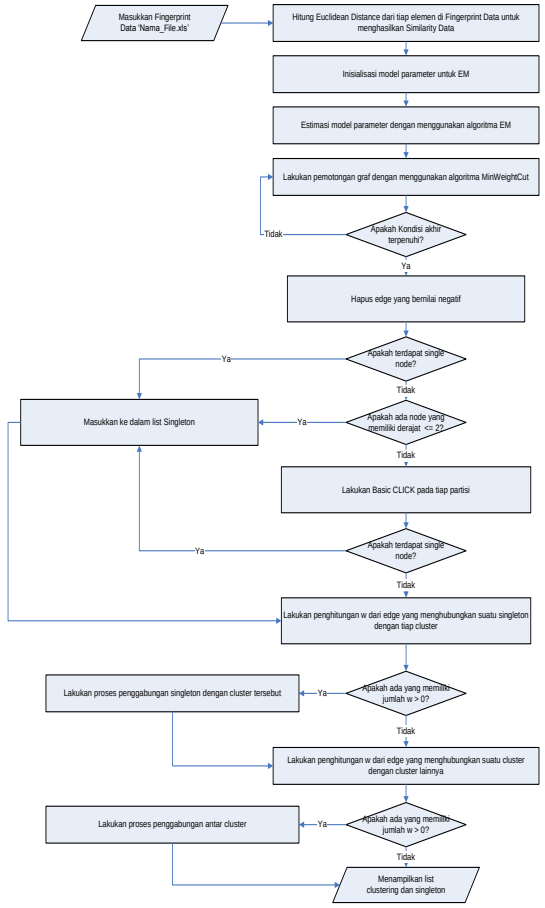
Berikut merupakan gambaran system secara keseluruhan:



Gambar 3-1: Pemodelan Sistem

3.3 PemodelanAlgoritma

Berikut adalah gambar alur dari algoritma CLICK atau *Cluster Identification via Connectivity Kernel* :



Gambar 3-2: Pemodelan Algoritma

3.4 Contoh Kasus

Pada batasan masalah ditulis bahwa jumlah minimal data adalah 16 buah gen, namun pada contoh kasus berikut ini diasumsikan bahwa batasan tersebut tidak berlaku dan telah dilakukan beberapa perubahan untuk menyesuaikan dengan jumlah data tersebut. Berikut contoh kasus beserta langkah-langkah pemecahannya dengan menggunakan algoritma CLICK :

1. Inputkan *Fingerprint data* dalam format .xls

Tabel 3-1: Fingerprint Data

	Uji1	Uji2	Uji3	Uji4
Gen1	0.6348661790	0.5834499920	-0.1410508230	-0.5383577210
Gen 2	0.4625551630	0.0451330370	-0.3206082550	-0.3126573580
Gen 3	0.7085918910	0.1488758720	-0.7515368540	-1.5059367050
Gen 4	-0.3417512150	-0.6202151680	-1.4429495760	-0.4430108350
Gen 5	0.2142124510	-0.3773183770	0.0108737280	-0.8024811600
Gen 6	-0.0444781560	-0.5376055380	-0.9321074430	-0.7458148770
Gen 7	-0.4856394360	-0.7040114340	-0.7814982730	0.0567684320

2. Kemudian dihasilkan *Similarity data* dengan menggunakan *Euclidean Distance*.

Tabel 3-2: *Similarity* Data

	Gen1	Gen2	Gen3	Gen4	Gen5	Gen6	Gen7
Gen1	0	0.6345	1.2260	2.0264	1.0921	1.5450	1.9176
Gen2	0.6345	0	1.2964	1.5382	0.7680	1.0762	1.3450
Gen3	1.2260	1.2964	0	1.8173	1.2638	1.2840	2.1439
Gen4	2.0264	1.5382	1.8173	0	1.6158	0.6692	0.8455
Gen5	1.0921	0.7680	1.2638	1.6158	0	0.9924	1.4009
Gen6	1.5450	1.0762	1.2840	0.6692	0.9924	0	0.9429
Gen7	1.9176	1.3450	2.1439	0.8455	1.4009	0.9429	0

3. *Generate* kelima parameter yaitu μ_T , σ_T^2, μ_F, σ_F^2 dan p_{mates} secara otomatis dengan menggunakan algoritma EM.

- Inisialisasi parameter

Inisialisasi dilakukan dengan cara mengekstrak nilai μ_T dan μ_F dari persamaan $m = p_{mates}*\mu_T + (1 - p_{mates})*\mu_F$ dan $v = p_{mates}*(1 - p_{mates})*(\mu_T - \mu_F)^2 + \sigma^2$ dimana nilai μ_F dienumerasi dari nilai 0 hingga nilai maksimal *similarity* dan nilai $\mu_T > \mu_F$. Nilai p_{mates} diperoleh dengan cara mengenumerasi semua kemungkinan nilai p_{mates} (0.1 - 1) dan mengambil semua kemungkinan kombinasi nilai parameter. Kemudian semua kemungkinan kombinasi parameter tersebut dibandingkan dengan distribusi empiris, yaitu dimana setiap parameter dienumerasi dan dicari antara parameter yang dienumerasi dengan semua kombinasi nilai parameter yang memiliki jarak yang paling minimum.

- Algoritma EM

Dalam algoritma ini dilakukan iterasi dimana setiap iterasi bertujuan untuk membuat nilai dari fungsi $Q(\theta|\theta^r) = \Sigma_{i<j} \Sigma^1_{t=0} g_t(S_{ij}) \log(f_t(S_{ij})p_t)$ menjadi konvergen dan maksimal. f_1 adalah fungsi pdf dari *mates* dan f_0 adalah fungsi pdf dari *non-mates*. $g_1 = ((f_1*p_1)/((f1*p_1)+(f_0*p_0)))$ dan $g_0 = 1- g_1$. Pada setiap perpindahan iterasi dilakukan optimasi parameter dengan menggunakan fungsi berikut :

$\mu_t = \Sigma_{i<j} (g_t(S_{ij})S_{ij}) / \Sigma_{i<j} g_t(S_{ij})$

$\sigma^2_T = \Sigma_{i<j} (g_t(S_{ij})(\mu_t - S_{ij})) / \Sigma_{i<j} g_t(S_{ij})$

$p_t = \Sigma_{i<j} g_t(S_{ij}) / \Sigma_{i<j} 1$

Dari algoritma tersebut dihasilkan parameter sebagai berikut :

μ_T : 1.6599

σ_T^2 : 0.6287

μ_F : 2.1000

σ_F^2 : 0.6234

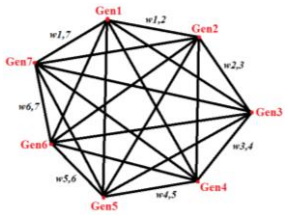
p_{mates} : 0.8168

4. Dilakukan penghitungan bobot *edge* yang menghubungkan antar gen.

Tabel 3-3: Bobot (w) dari Tiap *Edge*

	Gen1	Gen2	Gen3	Gen4	Gen5	Gen6	Gen7
Gen1	0	1.4865	1.4861	1.5093	1.4849	1.4921	1.5046
Gen2	1.4865	0	1.4871	1.4919	1.4851	1.4848	1.4878
Gen3	1.4861	1.4871	0	1.5006	1.4866	1.4869	1.5150
Gen4	1.5093	1.4919	1.5006	0	1.4940	1.4861	1.4847
Gen5	1.4849	1.4851	1.4866	1.4940	0	1.4845	1.4888
Gen6	1.4921	1.4848	1.4869	1.4861	1.4845	0	1.4846
Gen7	1.504	1.4878	1.5150	1.4847	1.4888	1.4846	0

5. Dari tabel matrix *adjacency* di atas telah terbentuk graf imajiner dengan tiap gen menjadi *vertexnya*



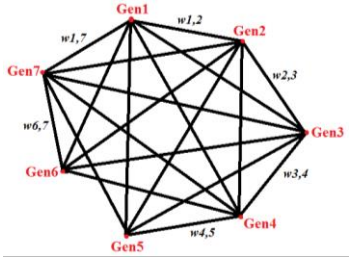
Gambar 3-3: Graf Imajiner

6. Kemudian dilakukan pemotongan graf dimulai dari edge yang memiliki bobot yang paling minimum, dapat dilihat pada table bahwa nilai bobot minimum adalah 1.4845 atau bobot dari edge 5-6.

Tabel 3-4: Bobot yang Paling Minimum (*Highlight*)

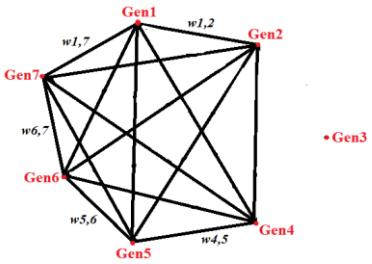
	Gen1	Gen2	Gen3	Gen4	Gen5	Gen6	Gen7
Gen1	0	1.4865	1.4861	1.5093	1.4849	1.4921	1.5046
Gen2	1.4865	0	1.4871	1.4919	1.4851	1.4848	1.4878
Gen3	1.4861	1.4871	0	1.5006	1.4866	1.4869	1.5150
Gen4	1.5093	1.4919	1.5006	0	1.4940	1.4861	1.4847
Gen5	1.4849	1.4851	1.4866	1.4940	0	1.4845	1.4888
Gen6	1.4921	1.4848	1.4869	1.4861	1.4845	0	1.4846
Gen7	1.504	1.4878	1.5150	1.4847	1.4888	1.4846	0

Maka edge tersebut akan dihapus dari graf



Gambar 3-4: Graf Imajiner Setelah Pemotongan Pertama

7. Demikian seterusnya penghapusan berlangsung secara rekursif hingga graf terbagi dua, kemudian dibentuk graf lengkap pada tiap bagian graf tersebut, sehingga tiap vertex terhubung dengan verex lain dalam satu bagian. Dan dilakukan pemeriksaan apakah telah memenuhi syarat sebagai suatu partisi dimana jumlah W > 0.

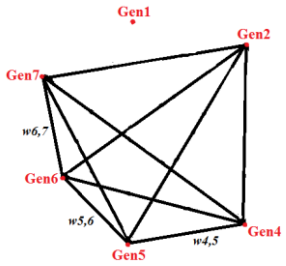


Gambar 3-5: Graf Imajiner Setelah Pemotongan Rekursif Pertama

Pada graf di atas terlihat bahwa pada pemotongan rekursif pertama menghasilkan dua bagian yaitu vertex gen 3 dan kumpulan vertex 1,2,4,5,6,7.

8. Setelah itu dilakukan pemeriksaan apakah bagian graf tersebut merupakan Singleton (terdiri dari satu node), sudah memenuhi syarat akhir partisi (W lebih besar dari 0), atau kah perlu dilakukan pembagian lagi. Dari gambar 3-5 tersebut dapat dilihat bahwa Gen3 masuk ke dalam *list* singleton sedangkan bagian graf yang terdiri dari vertex Gen1, Gen2, Gen4, Gen5, Gen6 dan Gen7 masih harus dibagi lagi.

9. Kemudian dari hasil pembagian graf lagi, diketahui bahwa dihasilkan sebuah partisi yang terdiri dari Gen2, Gen4, Gen5, Gen6, dan Gen7. Sedangkan Gen1 masuk ke dalam *list* Singleton juga.



Gambar 3-6: Graf Imajiner Setelah Pemotongan Rekursif Kedua

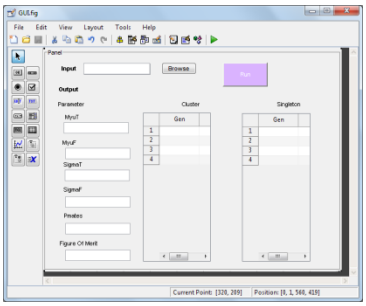
10. Setelah dihasilkan partisi-partisi, dilakukan proses selanjutnya yaitu penghilangan *edge* yang bernilai negatif, penghilangan *vertex* yang berderajat dua, serta penerapan Basic Click.

11. Terakhir dilakukan proses Adopsi Singleton oleh partisi-partisi yang ada serta proses Merge antar partisi yang ada. Dari proses ini dihasilkan satu buah *cluster* yang terdiri dari Gen2, Gen4,

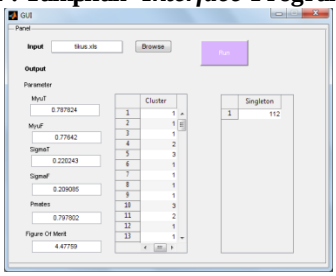
Gen5, Gen6 dan Gen7 serta *list* Singleton yang terdiri dari Gen1 dan Gen3.

3.5 Perancangan *Interface*

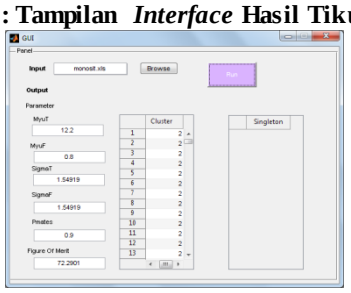
Tampilan *interface* yang telah dibuat untuk sistem adalah sebagai berikut :



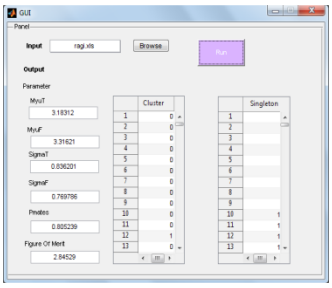
Gambar 3-7: Tampilan *Interface* Program



Gambar 3-8: Tampilan *Interface* Hasil Tikus



Gambar 3-9: Tampilan *Interface* Hasil Monosit



Gambar 3-10: Tampilan *Interface* Hasil Ragi

3.6 Lingkungan Implementasi Sistem

Lingkungan implementasi sistem terbagi menjadi dua yaitu lingkungan perangkat lunak dan perangkat keras.

- Lingkungan Perangkat Lunak:

- Sistem Operasi : Windows7

- *Software* MATLAB

- Lingkungan Perangkat Keras:

- *Processor* Intel Pentium(R) Dual Core @ 2,20 GHz

- *Memory* : 1GB

- *Hard Disk* : 300 GB

4 Implementasi dan Analisis Hasil Pengujian

4.1 Lingkungan Pengujian

Semua pengujian terhadap sistem dilakukan pada lingkungan pengujian sebagai berikut:

- *Processor* Intel Pentium(R) Dual Core @ 2,20 GHz
- *Memory* : 1GB
- *Hard Disk* : 300 GB
- Sistem operasi : Windows7

4.2 *Data* Pengujian

Untuk melakukan pengujian, dipilih 3 buah *dataset* yang telah disediakan oleh *website* http://www.math.unipa.it/~lobosco/genclust/ [1]. *Dataset* terdiri dari *m* buah gen atau baris dan *n* buah jenis eksperimen atau kolom.

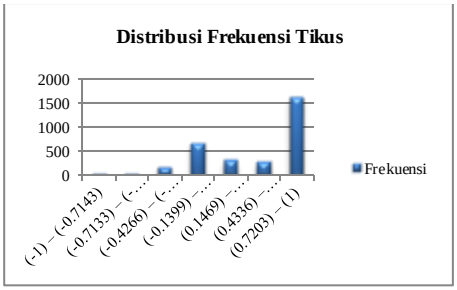
Berikut adalah *dataset* yang digunakan berikut deskripsinya :

Data Tikus.xls

- Jumlah Gen = 112
- Jumlah Eksperimen = 17
- Mean = 0.2870
- Variansi = 0.1738

Tabel 4-1 : Distribusi Frekuensi Data Tikus

Kelas	Frekuensi
(-1) – (-0.7143)	11
(-0.7133) – (-0.4276)	37
(-0.4266) – (-0.1409)	181
(-0.1399) – (0.1459)	664
(0.1469) – (0.4326)	311
(0.4336) – (0.7193)	282
(0.7203) – (1)	1622



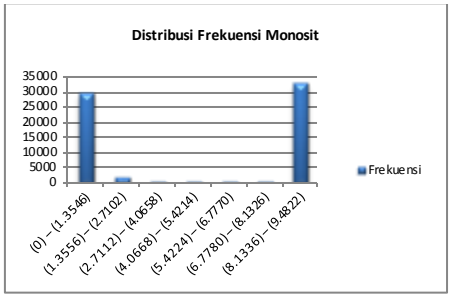
Gambar 4-1: Grafik Distribusi Frekuensi Tikus

Data Monosit.xls

- Jumlah Gen = 235
- Jumlah Eksperimen = 139
- Mean = 0.5371
- Variansi = 0.7083

Tabel 4-2 : Distribusi Frekuensi Data Monosit

Kelas	Frekuensi
(0) – (1.3546)	29580
(1.3556) – (2.7102)	2031
(2.7112) – (4.0658)	621
(4.0668) – (5.4214)	276
(5.4224) – (6.7770)	89
(6.7780) – (8.1326)	44
(8.1336) – (9.4822)	32621



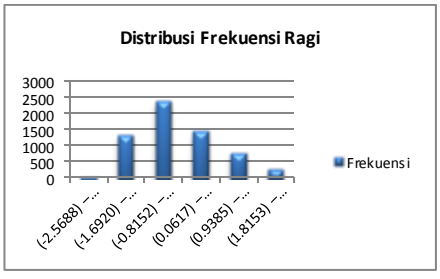
Gambar 4-2: Grafik Distribusi Frekuensi Monosit

Data Ragi.xls

- Jumlah Gen = 384
- Jumlah Eksperimen = 17
- Mean = 0.0033
- Variansi = 0.9956

Tabel 4-3 : Distribusi Frekuensi Data Ragi

Kelas	Frekuensi
(-2.5688) – (-1.6930)	70
(-1.6920) – (-0.8162)	1354
(-0.8152) – (0.0607)	2425
(0.0617) – (0.9375)	1478
(0.9385) –(1.8143)	808
(1.8153) – (2.6912)	316
(2.6922) – (3.5620)	6212



Gambar 4-3: Grafik Distribusi Frekuensi Ragi

Dari *data* di atas dapat dilihat bahwa Tikus memiliki nilai variansi yang kecil serta penyebaran data yang konvergen atau memusat pada nilai maksimum, sehingga lebih bersifat homogen. Monosit memiliki data yang lebih heterogen karena nilai variansinya yang lebih besar dari Tikus serta konvergen pada nilai maksimum dan nilai minimumnya. Kemudian yang terakhir adalah Ragi yang memiliki nilai variansi paling besar antara ketiga data dan memiliki persebaran yang lebih merata sehingga bersifat paling heterogen.

4.3 Skenario Pengujian

Skenario pengujian dilakukan pada ketiga jenis *data* yang telah dijelaskan sebelumnya dan dilakukan dengan menggunakan penghitungan 2-norm *Figure of Merit* (FOM) dengan rumus sebagai berikut :

$$FOM(e) = \Sigma^{m}_{e=1} (FOM(e,k))$$

Dalam melakukan skenario pengujian ini, pertama-tama masukkan *input* berupa fingerprint *data* dan lakukan *clusterisasi* dengan algoritma CLICK yang telah diimplementasikan. Kemudian hitung nilai FOM dari *cluster-cluster* yang terbentuk. Lakukan skenario ini untuk kedua *data* lainnya.

4.4 Analisis Hasil Pengujian

Dari hasil pengujian didapatkan hasil *clustering* sebagai berikut :

***Data* Tikus.xls**

- Jumlah *cluster* = 3
- Jumlah *singleton* = 1
- Parameter
  - MyuT = 0.787823968
  - MyuF = 0.776419981
  - SigmaT = 0.220243416
  - SigmaF = 0.209084825
  - Pmates = 0.797801541

Berikut hasil *clusterisasi data* Tikus dalam bentuk *list array* yang memuat nama-nama gen dalam data Tikus serta dijabarkan dari atas ke bawah dan dari kiri ke kanan :

Hasil anggota gen Tikus dalam *Cluster* 1

keratin	GRa3	NMDA2B	NGF	InsR
cellubrevin	GRa4	NMDA2C	NT3	IGFR1
nestin	GRa5	NMDA2D	BDNF	IP3R1
L1	GRb1	nAChRa2	CNTF	IP3R2
NFL	GRb2	nAChRa4	trk	IP3R3
NFM	GRb3	nAChRa5	trkB	cyclin_B
NFH	GRg2	nAChRa6	MK2	H2AZ
S100_beta	GRg3	nAChRa7	GDNF	cjun
GFAP	mGluR1	nAChRd	EGF	cfos
MOG	mGluR2	nAChRe	bFGF	Brm
GAD67	mGluR4	mAChR2	aFGF	SOD
G67I80/86	mGluR5	mAChR3	PDGFb	CCO2
G67I86	mGluR6	mAChR4	EGFR	SC1
ChAT	mGluR7	5HT1b	PDGFR	SC2
ACHE	mGluR8	5HT1c	TGFR	SC6
TH	NMDA1	5HT2	Ins1	SC7
GRa2	NMDA2A	5HT3	Ins2	

Hasil anggota gen Tikus dalam *Cluster* 2

MAP2	NOS
neno	nAChRa3
GAD65	trkC
GAT1	

Hasil anggota gen Tikus dalam *Cluster* 3

GAP43	GRa1	PTN	IGF_II	statin
synaptophysin	GRg1	PDGFa	IGFR2	TCP
pre-GAD67	mGluR3	FGFR	CRAF	actin
ODC	CNTFR	IGF_I	cyclin_A	CCO1

Hasil anggota gen Tikus dalam List Singleton

DD63_2

***Data* Monosit.xls**

- Jumlah *cluster* = 2
- Jumlah *singleton* = 0
- Parameter
- MyuT = 12.2
- MyuF = 0.8
- SigmaT = 1.549193338
- SigmaF = 1.549193338
- Pmates = 0.9

Berikut hasil *clusterisasi* data Monosit dalam bentuk *list array* yang memuat nama-nama gen dalam data Monosit serta dijabarkan dari atas ke bawah dan dari kiri ke kanan :

Hasil anggota gen Monosit dalam *Cluster* 1

194	129	273	42	123	108	133	311	116	15	47	83	166
160	244	295	54	124	113	136	312	122	16	52	85	168
572	248	298	78	146	117	137	313	143	17	53	91	174
4	322	315	89	161	120	140	318	169	21	55	95	175
32	27	48	90	162	125	144	3	196	23	57	101	176
45	58	60	126	177	127	147	5	199	25	59	104	178
118	77	65	131	189	145	153	19	207	26	61	107	180
66	86	112	134	206	148	155	22	220	28	62	109	183
100	130	142	141	222	151	156	24	221	29	63	110	186
165	149	173	185	252	154	157	35	226	30	64	114	193
167	179	212	188	272	182	190	44	242	31	68	115	2050
20	251	259	2	282	192	223	69	255	33	70	119	
43	256	260	6	8	213	238	76	257	34	71	121	
79	96	300	9	50	214	245	97	262	36	73	128	
80	105	1	18	56	215	247	99	270	37	74	132	
49	197	7	72	87	40	253	102	10	38	75	139	
93	240	12	88	92	41	265	106	11	39	81	163	
103	264	14	94	98	67	290	111	13	46	82	164	

Hasil anggota gen Monosit dalam *Cluster* 2

302	150
51	158
135	170
307	172

***Data* Ragi.xls**

- Jumlah *cluster* = 1
- Jumlah *singleton* = 370
- Parameter
- MyuT = 3.183120
- MyuF = 3.316210
- SigmaT = 0.836201398
- SigmaF = 0.769785678
- Pmates = 0.805239102

Berikut hasil *clusterisasi* data Ragi dalam bentuk *list array* yang memuat nama-nama gen dalam data Ragi serta dijabarkan dari atas ke bawah dan dari kiri ke kanan :

Hasil anggota gen Ragi dalam *Cluster* 1

14	226	336
36	251	346
193	257	354
209	295	372
222	317	

Hasil anggota gen Ragi dalam Singleton

1	31	62	91	120	149	178	211	244	277	308	341	377
2	32	63	92	121	150	179	212	245	278	309	342	378
3	33	64	93	122	151	180	213	246	279	310	343	379
4	35	65	94	123	152	181	214	247	280	311	345	380
5	37	66	95	124	153	182	215	248	281	312	347	381
6	38	67	96	125	154	183	216	250	282	313	348	382
7	39	68	97	126	155	184	217	252	283	314	349	383
8	40	69	98	127	156	185	218	253	284	316	350	384
9	41	70	99	128	157	186	219	254	285	318	351	374
10	42	71	100	129	158	187	221	256	286	319	353	375
11	43	72	101	130	159	188	223	258	287	320	355	339
12	44	73	102	131	160	189	225	259	288	321	356	340
13	45	74	103	132	161	190	227	260	289	322	357	306
15	46	75	104	133	162	192	228	261	290	323	358	307
16	47	76	105	134	163	194	229	262	291	324	359	275
17	48	77	106	135	164	195	230	263	292	325	360	276
18	49	78	107	136	165	196	231	264	294	326	361	242
19	50	79	108	137	166	197	232	265	296	327	362	243
20	51	80	109	138	167	198	233	266	297	328	363	208
21	52	81	110	139	168	199	234	267	298	329	364	210
22	53	82	111	140	169	200	235	268	299	330	365	34
23	54	83	112	141	170	201	236	269	300	331	366	191
24	55	84	113	142	171	202	237	270	301	332	367	
25	56	85	114	143	172	203	238	271	302	333	368	
26	57	86	115	144	173	204	239	272	303	335	369	
27	58	87	116	145	174	205	240	273	304	337	371	
28	59	88	117	146	175	206	241	274	305	338	373	
29	60	30	61	90	89	119	118	148	147	177	176	
293	315	334	344	352	370	376	255	249	220	224	207	

Kemudian setelah dilakukan penghitungan nilai *Figure of Merit* (FOM) didapatkan hasil sebagai berikut :

Tabel 4-4 : Nilai FOM, Mean FOM, H_{Ave} dan S_{Ave}

	FOM	Mean FOM	H_{Ave}	S_{Ave}
Ragi	2.84	0.1494	5.2611	0.3877
Monosit	72.29	0.5200	10.9785	11.5200
Tikus	4.47	0.2629	1.6270	0.8262

Keterangan :

- Mean FOM dihitung untuk mendapatkan nilai yang lebih spesifik untuk tiap eksperimen serta untuk menghilangkan pengaruh banyak eksperimen terhadap nilai FOM yang dihasilkan.

Mean FOM = FOM/jumlah eksperimen (3.1)

- H_{Ave} atau *homogeneity of a clustering* didefinisikan sebagai nilai similarity rata-rata antara mates.

$H_{Ave} = (\Sigma_{i,j \text{ mates. } i<j} Si(F(i),F(j))) / M$ (3.2)

Dimana :

M = jumlah pasangan mates

F = *Fingerprint Data*

Si = *Similarity Data*

- S_{Ave} atau *separation of a clustering* didefinisikan sebagai nilai similarity rata-rata antara non-mates.

$S_{Ave} = (2*(\Sigma_{i,j \text{ non mates. } i<j} Si(F(i),F(j)))) / (n(n-1)-2*M)$ (3.3)

Dari tabel hasil pengujian serta penghitungan FOM di atas dapat dilihat bahwa *data* Monosit yang sifat heterogen maupun homogennya berada di antara dua data lainnya memiliki nilai

Mean FOM yang paling tinggi yaitu sebesar 0.52 dengan jumlah *cluster* sebanyak dua buah dan tanpa singleton. Kemudian *data* dengan nilai Mean FOM tertinggi kedua adalah *data* Tikus yang bersifat paling homogen memiliki Mean FOM sebesar 0.0257 dengan jumlah *cluster* sebanyak tiga buah dan satu buah singleton. Terakhir data Ragi yang bersifat paling heterogen memiliki nilai Mean FOM paling kecil yaitu sebesar 0.0154 dengan jumlah *cluster* sebanyak satu buah dan singleton sebanyak 370 buah. Penghitungan kekuatan prediksi tersebut diperkuat dengan adanya nilai S_{Ave} dan H_{Ave}, dimana semakin kecil nilai S_{Ave} dan semakin besar nilai H_{Ave} maka semakin bagus clustering yang dilakukan. Pada data Monosit yang memiliki nilai FOM paling rendah dihasilkan nilai S_{Ave} yang lebih besar daripada H_{Ave} sehingga dapat dilihat bahwa cluster yang dihasilkan tidak begitu baik. Sedangkan untuk kedua data lainnya nilai S_{Ave} lebih kecil daripada H_{Ave} sehingga dapat dilihat bahwa untuk kedua data tersebut, algoritma CLICK menghasilkan clusterisasi yang cukup baik.

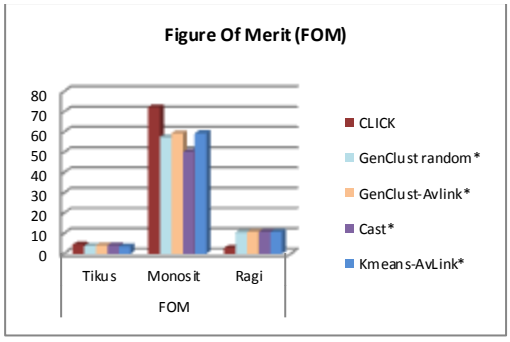
Seperti yang telah dijelaskan pada bab 2 bahwa suatu algoritma memiliki kekuatan prediksi yang baik jika FOM bernilai kecil, hal ini juga dapat diterapkan pada nilai Mean FOM, sehingga dapat disimpulkan bahwa tingkat heterogen atau homogennya suatu data tidak menentukan nilai dari FOM data tersebut. Hal ini dapat dilihat dari Mean FOM Monosit yang sifat homogenitasnya berada di antara dua data lainnya justru memiliki nilai yang paling tinggi.

Kemudian untuk semakin menekankan mengenai pengaruh sifat heterogen dan homogen suatu data terhadap nilai FOM serta untuk melihat apakah benar CLICK memiliki keunggulan dalam melakukan *clustering* data gen, dapat dilihat pada tabel dan grafik perbandingan antara algoritma CLICK dengan algoritma K-Means, Cast dan GenClust yang telah dibuat dan diuji oleh Alessandra Raimondi *et al* berikut :

Tabel 4-5 : Nilai FOM CLICK, GenClust, K-Means dan Cast

Algoritma	FOM		
	Tikus	Monosit	Ragi
CLICK	4.47	72.29	2.84
GenClust random*	3.89	57.49	10.6
GenClust-Avlink*	4.07	59.33	10.804
Cast*	3.98	50.21	10.84
Kmeans-AvLink*	3.71	59.49	10.73

*Sumber = GenClust: A Genetic Algorithm For Clustering Gene Expression Data.[4]



Gambar 4-4 : Grafik Perbandingan nilai *Figure Of Merit* (FOM)

Dari tabel dan grafik perbandingan di atas dapat dilihat bahwa nilai FOM untuk *data* Monosit dengan menggunakan algoritma CLICK, GenClust, Cast maupun K-Means memiliki nilai yang jauh lebih tinggi dibanding dengan kedua *data* lainnya. Sehingga dapat disimpulkan bahwa sifat heterogen atau homogen suatu *data* tidak mempengaruhi nilai FOM adalah benar. Kemudian juga dapat dilihat bahwa untuk *data* Tikus nilai FOM untuk algoritma CLICK sedikit lebih besar namun tidak terlalu jauh berbeda dengan ketiga algoritma lainnya. Untuk *data* Monosit, nilai FOM algoritma CLICK jauh lebih tinggi dari ketiga algoritma lainnya sedangkan untuk *data* Ragi algoritma CLICK memiliki nilai yang jauh lebih kecil dari ketiga algoritma yang lainnya. Dari hasil tersebut dapat disimpulkan bahwa untuk *data* yang memiliki jumlah eksperimen yang sama, algoritma CLICK memiliki keunggulan dalam melakukan *clusterisasi* data gen yang bersifat heterogen di antara ketiga algoritma lainnya. Sedangkan untuk data yang bersifat homogen, performansi algoritma CLICK untuk melakukan *clusterisasi* masih kurang baik dibanding ketiga algoritma lainnya.

5. Kesimpulan dan Saran

5.1 Kesimpulan

Setelah melakukan implementasi, pengujian, dan analisis, dapat ditarik kesimpulan sebagai berikut :

a. Sifat homogen dan heterogen suatu data tidak berpengaruh terhadap nilai FOM data tersebut setelah dilakukan *clusterisasi*. Hal ini dapat dilihat dari data Monosit yang menghasilkan nilai FOM tertinggi walaupun sifat homogenitasnya berada di antara kedua data lainnya.

b. Untuk jumlah eksperimen yang sama, algoritma CLICK memiliki performansi yang lebih baik dalam melakukan clustering data gen yang bersifat heterogen dibanding dengan algoritma GenClust, K-Means dan Cast. Sedangkan untuk data yang bersifat homogen, algoritma CLICK memiliki performansi yang kurang baik dibanding dengan ketiga algoritma lainnya.

5.2 Saran

Setelah melakukan tugas akhir ini maka saran yang dapat diberikan untuk pengembangan lebih lanjut adalah

- Sebaiknya dilakukan efisiensi waktu dan memori yang lebih baik dalam pengimplementasian algoritma CLICK ini.

- Dalam pengembangan selanjutnya mungkin dapat dilakukan penghitungan akurasi terhadap solusi *cluster* yang sebenarnya.

Daftar Pustaka

[1]http://www.math.unipa.it/~lobosco/*genclust*/ (diakses tanggal 21 Oktober 2010)

[2]http://smartstat.wordpress.com/2010/03/27/ukuran-penyebaran-measures-dispersion/ (diakses tanggal 26 Januari 2012)

[3] Adi Maron-Katz, Roded Sharan, Ron Shamir. 2003. *CLICK And EXPANDER: A System For Clustering And Visualizing Gene Expression Data*.

[4] Alessandra Raimondi, Davide Scaturro, Giosué Lo Bosco, Raffaele Giancarlo, Vito Di Gesú. 2005. *GenClust: A Genetic Algorithm For Clustering Gene Expression Data*.

[5] D. R. Haynor, K. Y. Yeung, W. L. Ruzzo. 2001. *Validating Clustering for Gene Expression Data*.

[6] Dali Wang, Habtom Ressom, Mohamad Musavi, Cristian Domnisoru. 2002. *Double Self-Organizing Maps to Cluster Gene Expression Data*.

[7] Roded Sharan, Ron Shamir. *CLICK: A Clustering Algorithm for Gene Expression Analysis*.

[8] Roded Sharan. 2002. *Graph Modification Problems and their Applications to Genomic Research*.

[9] Ron Shamir. 2005. *CLICK: Cluster Identification via Connectivity Kernels*.