APLIKASI ROUGH SET UNTUK MEMPREDIKSI PRESTASI CALON ANGGOTA KELOMPOK PROGRAMMING (STUDI KASUS : STMIK PELITA NUSANTARA)

Muhammad Romi Syahputra

Teknik Informatika
STMIK Pelita Nusantara Medan, Jl. Iskandar Muda No. 1 Medan, Sumatera Utara 20154, Indonesia

moehromi89@gmail.com

Abstract

Data mining telah berkembang pesat dan menambah nilai suatu informasi yang tersimpan dalam database. Salah satu algoritma data mining yang cukup sederhana adalah Rough Set. Penerapan dari algoritma Rough Set ini adalah prediksi prestasi calon anggota kelompok programming STMIK Pelita Nusantara. Setiap penerimaan anggota baru, STMIK Pelita Nusantara selalu mengadakan seleksi yang terdiri dari: Test Kemampuan Logika, Kemampuan Penyusunan Algoritma, Perintah Dasar Bahasa Pemrograman, dan Pengetahuan Tentang Komputer. Hasil test ini dapat memberikan hasil yang cukup variatif, dalam pengertian bahwa ada kalanya seorang calon anggota lemah di satu bidang, tetapi memiliki kelebihan di bidang lainnya dengan prestasi yang cukup beragam. Berdasarkan pertimbangan tersebut maka diperlukan aplikasi prediksi ini untuk menggali data-data yang ada sebelumnya di dalam database dengan menggunakan algoritma rough set sehingga dapat diperoleh hasil berupa prediksi prestasi calon anggota kelompok programming STMIK Pelita Nusantara.

Kata Kunci : Data mining, Rough set

I. PENDAHULUAN

Data mining merupakan bidang yang berkembang pesat seiring dengan perkembangan teknologi informasi yang melibatkan pemakaian *database* berskala besar maupun kecil. Informasi yang tersimpan dalam *database* menjadi tidak berguna seiring berjalannya waktu.

Data mining dapat meningkatkan nilai tambah suatu *database*. Kita dapat menggali informasi yang tersimpan dalam *database* yang terakumulasi dalam jangka waktu lama untuk mendapatkan informasi tambahan.

Banyak algoritma mengimplementasikan *data mining*. Salah satu algoritma yang cukup sederhana dan cukup mudah untuk diimplementasikan adalah algoritma *Rough Set*. *Rough Set* merepresentasikan set data sebagai sebuah tabel, di mana baris dalam tabel merepresentasikan objek dan kolom-kolom merepresentasikan atribut dari objek-objek tersebut.

Salah satu bidang penerapan dari algoritma *Rough Set* adalah prediksi prestasi calon anggota kelompok p*rogramming* STMIK *Pelita Nusantara* berdasarkan data-data yang sudah ada sebelumnya yang tersimpan di dalam *database*. STMIK *Pelita Nusantara* perlu melakukan prediksi prestasi calon anggota kelompok *programming* karena kelompok ini merupakan ujung tombak STMIK *Pelita Nusantara* di dalam mengikuti kompetisi bidang pemrograman yang banyak diselenggarakan oleh institusi pemerintah maupun antar perguruan tinggi. Peminat k kelompok *Programming* pada STMIK *Pelita Nusantara* tiap tahun mengalami peningkatan. Setiap penerimaan anggota baru, STMIK *Pelita Nusantara* selalu mengadakan seleksi yang terdiri dari: Test Kemampuan Logika, Kemampuan Penyusunan Algoritma, Perintah Dasar Bahasa Pemrograman, dan Pengetahuan Tentang Komputer. Hasil test ini dapat memberikan hasil yang cukup variatif, dalam pengertian bahwa ada kalanya seorang calon anggota lemah di satu bidang, tetapi memiliki kelebihan di bidang lainnya dengan prestasi yang cukup beragam. Mengingat luasnya permasalahan yang berkaitan dengan prediksi prestasi calon anggota kelompok *Programming* STMIK *Pelita Nusantara* maka peneliti merasa perlu untuk membatasi ruang lingkup permasalahan yang akan dibahas dalam penelitian ini, yaitu:

a. Atribut kondisi yang digunakan terdiri dari Test Kemampuan Logika, Kemampuan Penyusunan Algoritma, Perintah Dasar Bahasa Pemrograman, dan Pengetahuan Tentang Komputer dari anggota kelompok *programming*.
b. Atribut Keputusan yang digunakan adalah prestasi dari anggota kelompok *programming*.
c. Data-data yang digunakan adalah merupakan data hasil seleksi anggota kelompok

programming STMIK *Pelita Nusantara* dan prestasi dari anggota tersebut.

Untuk implementasi dapat menggunakan perangkat lunak Rosetta.

1.1 Model, Analisa, Desain dan Implementasi

Secara sederhana *data mining* adalah penambangan atau penemuan informasi baru dengan mencari pola atau aturan tertentu dari sejumlah data yang sangat besar. [1] *Data mining* juga disebut sebagai serangkaian proses untuk menggali nilai tambah berupa pengetahuan yang selama ini tidak diketahui secara manual dari suatu kumpulan data. [3]

Data mining, sering juga disebut sebagai *Knowledge Discovery in Database* (KDD). KDD adalah kegiatan yang meliputi pengumpulan, pemakaian data, historis untukmenemukan keteraturan, pola atau hubungan dalam set data berukuran besar. [4]

Rough Set dibangun oleh Zdzislaw Pawlak diawal tahun 1980-an. Filosofi dari metode ini adalah bahwa informasi (knowledge, data) bisa diasosiasikan dengan objek. [2] Dalam *Rough Set*, sebuah set data direpresentasikan sebagai sebuah tabel, dimana baris dalam tabel merepresentasikan objek dan kolom-kolom merepresentasikan atribut dari objek-objek tersebut.

Adapun tahapan di dalam penggunaan algoritma *Rough Set* ini sebagai berikut:

a. *Data Selection* (Pemilihan data yang akan digunakan)
b. Pembentukan *Decision System* yang berisikan atribut kondisi dan atribut keputusan.
c. Pembentukan *Equivalence Class*, yaitu dengan menghilangkan data yang berulang.
d. Pembentukan *Discernibility Matrix Modulo D*, yaitu matriks yang berisikan perbandingan antar data yang berbeda atribut kondisi dan atribut keputusan.
e. Menghasilkan *reduct* dengan menggunakan aljabar boolean.
f. Menghasilkan *rule* (pengetahuan).

II. TEORI

2.1 Data Selection

Data yang digunakan adalah merupakan data hasil seleksi dari anggota kelompok *programming* dalam kurun waktu 2010-2012 yang terdiri dari: Test Kemampuan Logika, Kemampuan Penyusunan Algoritma, Perintah Dasar Bahasa Pemrograman, dan Pengetahuan Tentang Komputer beserta dengan prestasi mereka ketika menjadi anggota kelompok *programming*.

2.2.Pembentukan *Decision System*

Adapun *Decision System* dari perancangan aplikasi prediksi prestasi calon anggota kelompok *Programming* STMIK *Pelita Nusantara* ini adalah terdiri dari:

a. Atribut Kondisi: Test Kemampuan Logika, Kemampuan Penyusunan Algoritma, Perintah Dasar Bahasa Pemrograman, dan Pengetahuan Tentang Komputer
b. Atribut Keputusan: Prestasi dari Anggota kelompok p*rogramming*.

Adapun *Decision System* yang ada dapat dilihat pada Tabel 1, dengan contoh data yang akan digunakan adalah sebanyak 10 (sepuluh) data

TABEL I. DECISION SYSTEM

Nama Anggota	Test Logika	Algoritma	Bahasa Pemrograman	Pengetahuan Komputer	**Prestasi**
Andy	Baik	Baik	S. Baik	S. Baik	**S. Baik**
Indra	Cukup	Cukup	Baik	Baik	**Baik**
Yulia	Cukup	Cukup	Cukup	Baik	**Cukup**
Andang	Cukup	Baik	Baik	S. Baik	**S. Baik**
Anizar	Cukup	Cukup	Baik	Baik	**Baik**
Budi	Cukup	S. Baik	Kurang	S. Baik	**Baik**
Rini	Cukup	Baik	Kurang	S. Baik	**Cukup**
Susi	Baik	Kurang	S. Baik	Cukup	**Baik**
Jhoni	Baik	Kurang	Cukup	S. Baik	**Baik**
Syafri	Kurang	Cukup	Baik	S. Baik	**Baik**

2.3. Pembentukan *Equivalence Class*

Pembentukan *Equivalence Class* dilakukan dengan cara menghilangkan data yang memiliki kesamaan, seperti data antara Indra dan Anizar memiliki kesamaan, maka pada *Equivalence Class* data tersebut menjadi tinggal 1 (satu) *Record*. Adapun hasil dari pembentukan *Equivalence Class* dapat dilihat pada Tabel 2.

TABEL II. EQUIVALENCE CLASS

	A	B	C	D	**K**
EC1	Baik	Baik	S. Baik	S. Baik	**S. Baik**
EC2	Cukup	Cukup	Baik	Baik	**Baik**
EC3	Cukup	Cukup	Cukup	Baik	**Cukup**
EC4	Cukup	Baik	Baik	S. Baik	**S. Baik**
EC5	Cukup	S. Baik	Kurang	S. Baik	**Baik**
EC6	Cukup	Baik	Kurang	S. Baik	**Cukup**
EC7	Baik	Kurang	S. Baik	Cukup	**Baik**
EC8	Baik	Kurang	Cukup	S. Baik	**Baik**
EC9	Kurang	Cukup	Baik	S. Baik	**Baik**

Pada bagian *Equivalence Class* ini untuk mempermudah penulisan, maka atribut kondisi memakai simbol A, B, C, dan D sehingga:

a. Atribut A mewakili Test Logika
b. Atribut B mewakili Penyusunan Algoritma
c. Atribut C mewakili Perintah Dasar Bahasa Pemrograman
d. Atribut D mewakili Pengetahuan tentang Komputer

2.4 Pembentukan *Discernibility Matrix ModulD*

Discernibility Matrix Modulo D adalah suatu

matriks yang berisikan perbandingan antar data yang berbeda atribut kondisi dan atribut keputusan. Data dengan atribut kondisi yang berbeda, tetapi atribut keputusannya sama tetap dianggap sama. Adapun *Discernibility Matrix Modulo D* dapat dilihat pada Tabel 3.

Tabel 3. Discernibility Matrix Modulo D

	EC1	EC2	EC3	EC4	EC5	EC6	EC7	EC8	EC9
EC1	X	ABCD	ABCD	X	ABC	AC	BD	BC	ABC
EC2	ABCD	X	C	BD	X	BCD	X	X	X
EC3	ABCD	C	X	BCD	BCD	X	ABCD	ABD	ACD
EC4	X	BD	BCD	X	BC	C	ABCD	ABC	ABC
EC5	ABC	X	BCD	BC	X	B	X	X	X
EC6	AC	BCD	X	C	B	X	ABCD	ABC	ABC
EC7	BD	X	ABCD	ABCD	X	ABCD	X	X	X
EC8	BC	X	ABD	ABC	X	ABC	X	X	X
EC9	ABC	X	ACD	ABC	X	ABC	X	X	X

Cara membaca *Discernibility Matrix Modulo D* adalah:

a. Simbol X menunjukkan bahwa data tersebut tidak memiliki perbedaan atribut keputusan ataupun atribut kondisi.
   Sebagai contoh, EC1 bila dibandingkan dengan EC1 maka tidak ada perbedaan atribut kondisi.
   EC2 dibandingkan dengan EC5 memiliki atribut keputusan yang sama yaitu **Baik** sehingga dianggap sama dan diberi simbol X
b. EC1 dan EC2 memiliki perbedaan atribut keputusan, dan memiliki perbedaan atribut kondisi untuk atribut A, B, C, dan D. Sehingga pada perpotongan antara *Cell* EC1 dan EC2 diisikan dengan ABCD

2.5 Menghasilkan *Reduct* denganMenggunakan Aljabar Boolean

Adapun beberapa teorema *boolean* yang dipakai di dalam algoritma *Rough Set* ini dapat dilihat pada Tabel 4.

Tabel 4. Prinsip Dasar Aljabar Boolean

1. HK. KOMUTATIF A + B = B + A A . B = B . A	6. HK. IDENTITAS A + A = A A . A = A
2. HK. ASSOSIATIF (A+B)+C = A+(B+C) (A.B) . C = A . (B.C)	7.0 + A = A ----- 1. A = A 1 + A = 1 ----- 0 . A = 0
3. HK. DISTRIBUTIF A . (B+C) = A.B + A.C A + (B.C) = (A+B) . (A+C)	8.A' + A = 1 A' . A =0
4. HK. NEGASI (A') = A' (A')' = A	9.A + A' . B = A + B A . (A + B)= A . B
5. HK. ABRSORPSI A+ A.B = A A.(A+B) = A	10. DE MORGAN'S (A+ B)' = A' . B' (A . B)' = A' + B'

Adapun *reduct* yang dihasilkan berdasarkan prinsip Aljabar Boolean dapat dilihat pada Tabel 5.

Tabel 5. *Reduct* yang Dihasilkan dengan prinsip Aljabar Boolean

Class	CNF of Boolean Function	Prime Implicant	Reducts
EC1	(A+B+C+D) * (A+B+C+D) * (A+B+C) * (A+C) * (B+D) * (B+C) * (A+B+C)	AB + BC + CD	{A,B}, {B,C}, {C,D}
EC2	(A+B+C+D) * C * (B+D) * (B+C+D)	C * (B+D) = CB + CD	{B,C}, {C,D}
EC3	(A+B+C+D) * C * (B+C+D) * (B+C+D) * (A+B+C+D) * (A+B+D) * (A+C+D)	BC + CD + AC	{B,C}, {C,D}, {A,C}
EC4	(B+D) * (B+C+D) * (B+C) * C * (A+B+C+D) * (A+B+C) * (A+B+C)	BC + CD	{B,C}, {C,D}
EC5	(A+B+C) * (B+C+D) * (B+C) * B	B	{B}
EC6	(A+C) * (B+C+D) * C * B * (A+B+C+D) * (A+B+C) * (A+B+C)	BC	{B,C}
EC7	(B+D) * (A+B+C+D) * (A+B+C+D) * (A+B+C+D)	B + D	{B}, {D}
EC8	(B+C) * (A+B+D) * (A+B+C) * (A+B+C)	B + CD + ACD	{B}, {C ,D}, {A,C,D}
EC9	(A+B+C) * (A+C+D) * (A+B+C) * (A+B+C)	A+ C + BD	A, C, {B,D}

Adapun cara menghasilkan *reduct* dengan menggunakan prinsip aljabar boolean dapat dijelaskan sebagai berikut. Misalkan untuk EC5.

1. Dari *Discernibility Matrix Modulo D* untuk EC5 di peroleh *Boolean Function* sebagai berikut.
   (A+B+C) * (B+C+D) * (B+C) * B
2. Langkah-langkah penyederhanaannya adalah sebagai berikut.
   - Kerjakan dulu operasi (A+B+C) * B
     Dapat disubstitusi menjadi:
     B * (A+1+C)
     Sesuai dengan aturan 1+A = 1
     Maka menjadi: B * 1 = B
   - Hasil operasi ini menyisakan:
     B * (B+C+D) * (B+C)
   - Kerjakan lagi operasi B * (B+C+D)
     Dapat disubstitusi menjadi:
     B * (1+C+D)
     Sesuai dengan aturan 1+A = 1
     Maka menjadi: B*1 = B
   - Hasil dari operasi ini menyisakan:
     B * (B+C)
   - Kerjakan lagi operasi B * (B+C)
     Dapat disubstitusi menjadi:
     B * (1+C)
     Sesuai dengan aturan 1+A=1
     Maka menjadi: B * 1 = B
   - Maka hasil akhir penyederhanaan adalah:
     (A+B+C) * (B+C+D) * (B+C) * B
     = B, yang akan menjadi *Reducts*

   Cara yang sama dapat diterapkan untuk *equivalence class* yang lain.

2.6. Menghasilkan Rule

Adapun *rule* yang dihasilkan berdasarkan *Reduct* terdiri dari kombinasi atribut sebagai berikut.

1. {B}= {Penyusunan Algoritma}
2. {D} = {Pengetahuan Komputer}
3. {A,B} = {Test Logika, Penyusunan Algoritma}
4. {B,C} = {Penyusunan Algoritma, Perintah Dasar Bahasa Pemrograman}
5. {C,D} = {Perintah Dasar Bahasa Pemrograman, Pengetahuan Komputer}
6. {A,C} = {Test Logika, Perintah Dasar Bahasa Pemrograman}
7. {B,D} = {Penyusunan Algoritma, Pengetahuan Komputer}
8. {A,C,D} = {Test Logika, Perintah Dasar Bahasa Pemrograman, Pengetahuan Komputer}

Sehingga *rule* yang dihasilkan adalah berdasarkan pada *equivalence class* dengan membandingkannya dengan kombinasi atribut yang ada, sehingga diperoleh hasil sebagai berikut.

1. {B} = {Penyusunan Algoritma}
   - Jika mahasiswa memiliki kemampuan penyusunan algoritma yang baik maka kemungkinan prestasinya adalah sangat baik atau cukup
   - Jika mahasiswa memiliki kemampuan penyusunan algoritma yang cukup maka kemungkinan prestasinya adalah cukup atau baik.
   - Jika mahasiswa memiliki kemampuan penyusunan algoritma yang sangat baik maka kemungkinan prestasinya adalah baik
2. {B, C} = {Penyusunan Algoritma, Perintah Dasar Bahasa Pemrograman
   - Jika Hasil Penyusunan Agoritma adalah Baik dan Penguasaan Perintah Dasar Bahasa Pemrograman adalah Sangat Baik maka kemungkinan prestasinya adalah Sangat Baik
   - Jika Hasil Penyusunan Agoritma adalah cukup dan Penguasaan Perintah Dasar Bahasa Pemrograman adalah Baik maka kemungkinan prestasinya adalah Baik
   - Jika Hasil Penyusunan Agoritma adalah Cukup dan Penguasaan Perintah Dasar Bahasa Pemrograman adalah Cukup maka kemungkinan prestasinya adalah Cukup
   - Jika Hasil Penyusunan Agoritma adalah Baik dan Penguasaan Perintah Dasar Bahasa Pemrograman adalah Baik maka kemungkinan prestasinya adalah Sangat Baik
   - Jika Hasil Penyusunan Agoritma adalah Cukup dan Penguasaan Perintah Dasar Bahasa Pemrograman adalah Baik maka kemungkinan prestasinya adalah Cukup
   - Jika Hasil Penyusunan Agoritma adalah kurang dan Penguasaan Perintah Dasar Bahasa Pemrograman adalah Sangat Baik maka kemungkinan prestasinya adalah Baik
   - Jika Hasil Penyusunan Agoritma adalah Kurang dan Penguasaan Perintah Dasar Bahasa Pemrograman adalah Cukup maka kemungkinan prestasinya adalah Baik.

Pembentukan *rule* untuk kombinasi atribut yang lain adalah sama.

2.4 Perancangan dan Implementasi

Perancangan dan Implementasi aplikasi yang bekerja dengan metode *rough set* ini dilakukan dengan menggunakan *software Rosetta*. Dalam hal ini aplikasi *Rosetta* yang digunakan adalah Rosetta 1.4.4.1. Aplikasi Rosetta ini mendukung *database* sebagai berikut.

1. Microsoft Access
2. Microsoft Excell
3. Dbase
4. Visual Foxpro
5. dll

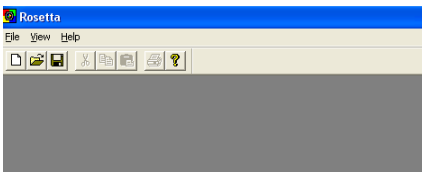
Adapun langkah-langkah untuk implementasi dari aplikasi *Rough Set* dengan menggunakan Rosetta ini adalah sebagai berikut.

1. Ketikkan data yang ada pada salah satu *database* yang dipilih, sebagai contoh misalkan data yang ada diketikkan pada lembar kerja *Microsoft Excel* seperti yang terlihat pada Gambar 1.

	A	B	C	D	E	F
1	Nama Anggota	Test Logika	Algoritma	Bahasa Pemrograman	Pengetahuan Komputer	**Prestasi**
2	Andy	Baik	Baik	S. Baik	S. Baik	**S. Baik**
3	Indra	Cukup	Cukup	Baik	Baik	**Baik**
4	Yulia	Cukup	Cukup	Cukup	Baik	**Cukup**
5	Andang	Cukup	Baik	Baik	S. Baik	**S. Baik**
6	Anzar	Cukup	Cukup	Baik	Baik	**Baik**
7	Budi	Cukup	S. Baik	Kurang	S. Baik	**Baik**
8	Rini	Cukup	Baik	Kurang	S. Baik	**Cukup**
9	Susi	Baik	Kurang	S. Baik	Cukup	**Baik**
10	Jhoni	Baik	Kurang	Cukup	S. Baik	**Baik**
11	Syafri	Kurang	Cukup	Baik	S. Baik	**Baik**

Gambar 1. Pengetikan Data pada Lembar Kerja MS Excel

1. Jalankan Aplikasi Rosetta, pada saat pertama kali dijalankan maka tampilannya adalah seperti pada Gambar 2.

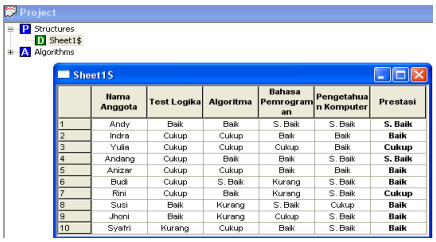


Gambar 2. Tampilan Awal Rosetta

2. Setelah itu Klik Menu File – New, maka tampilannya adalah seperti pada Gambar 3.

Gambar 3. Tampilan Setelah Menu File-New Diklik

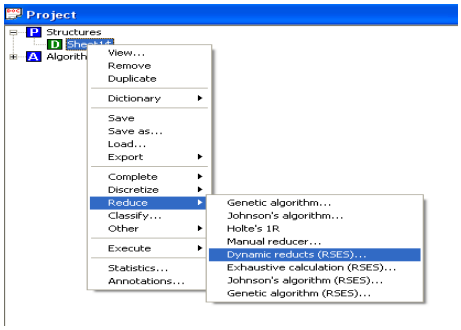
3. Setelah itu klik kanan pada Bagi Structure pilih ODBC, tentukan klik tombol Open Database – Machine Data Source Pilih MS Excel, kemudian arahkan pada lembar kerja MS Excel tempat kita menyimpan data. Hasilnya adalah seperti terlihat pada Gambar 4.



	Nama Anggota	Test Logika	Algoritma	Bahasa Pemrograman	Pengetahuan Komputer	Prestasi
1	Andy	Baik	Baik	S. Baik	S. Baik	S. Baik
2	Indra	Cukup	Cukup	Baik	Baik	Baik
3	Yulia	Cukup	Cukup	Cukup	Baik	Cukup
4	Andang	Cukup	Baik	Baik	S. Baik	S. Baik
5	Anizar	Cukup	Cukup	Baik	Baik	Baik
6	Budi	Cukup	S. Baik	Kurang	S. Baik	Baik
7	Rini	Cukup	Baik	Kurang	S. Baik	Cukup
8	Susi	Baik	Kurang	S. Baik	Cukup	Baik
9	Jhoni	Baik	Kurang	Cukup	S. Baik	Baik
10	Syafri	Kurang	Cukup	Baik	S. Baik	Baik

Gambar 4. Tampilan setelah dilakukan pemilihan data pada Rosetta

4. Klik kanan pada Sheet 1 (lembar kerja MS Excel), Pilih Reduces – Dynamic Reducts, seperti yang terlihat pada Gambar 5.

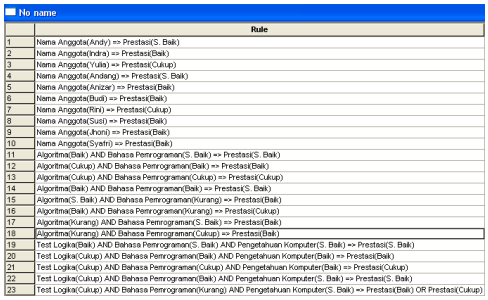


Gambar 5. Pemilihan Dynamic Reducts untuk Menghasilkan Reducts

5. Setelah itu akan dihasilkan Dynamic Reducts seperti yang terlihat pada Gambar 6.

Gambar 6. Tampilan Reducts yang Dihasilkan pada Sheet 1

6. Klik Kanan pada Reducts yang dihasilkan kemudian pilih Generate Rules, maka akan dihasilkan *rule* (pengetahuan) seperti yang terlihat pada Gambar 7.



	Rule
1	Nama Anggota(Andy) => Prestasi(S. Baik)
2	Nama Anggota(Indra) => Prestasi(Baik)
3	Nama Anggota(Yulia) => Prestasi(Cukup)
4	Nama Anggota(Andang) => Prestasi(S. Baik)
5	Nama Anggota(Anizar) => Prestasi(Baik)
6	Nama Anggota(Budi) => Prestasi(Baik)
7	Nama Anggota(Rini) => Prestasi(Cukup)
8	Nama Anggota(Susi) => Prestasi(Baik)
9	Nama Anggota(Jhoni) => Prestasi(Baik)
10	Nama Anggota(Syafri) => Prestasi(Baik)
11	Algoritma(Baik) AND Bahasa Pemrograman(S. Baik) => Prestasi(S. Baik)
12	Algoritma(Cukup) AND Bahasa Pemrograman(Baik) => Prestasi(Baik)
13	Algoritma(Cukup) AND Bahasa Pemrograman(Cukup) => Prestasi(Cukup)
14	Algoritma(Baik) AND Bahasa Pemrograman(Baik) => Prestasi(S. Baik)
15	Algoritma(S. Baik) AND Bahasa Pemrograman(Kurang) => Prestasi(Baik)
16	Algoritma(Baik) AND Bahasa Pemrograman(Kurang) => Prestasi(Cukup)
17	Algoritma(Kurang) AND Bahasa Pemrograman(S. Baik) => Prestasi(Baik)
18	Algoritma(Kurang) AND Bahasa Pemrograman(Cukup) => Prestasi(Baik)
19	Test Logika(Baik) AND Bahasa Pemrograman(S. Baik) AND Pengetahuan Komputer(S. Baik) => Prestasi(S. Baik)
20	Test Logika(Cukup) AND Bahasa Pemrograman(Baik) AND Pengetahuan Komputer(Baik) => Prestasi(Baik)
21	Test Logika(Cukup) AND Bahasa Pemrograman(Cukup) AND Pengetahuan Komputer(Baik) => Prestasi(Cukup)
22	Test Logika(Cukup) AND Bahasa Pemrograman(Baik) AND Pengetahuan Komputer(S. Baik) => Prestasi(S. Baik)
23	Test Logika(Cukup) AND Bahasa Pemrograman(Kurang) AND Pengetahuan Komputer(S. Baik) => Prestasi(Baik) OR Prestasi(Cukup)

Gambar 7. Rule yang Dihasilkan

Pada Gambar 7 terlihat bahwa untuk atribut {B,C} = {Penyusunan Algoritma, Perintah Dasar Bahasa Pemrograman), untuk *rule* (pengetahuan) yang dihasilkan dengan pengerjaan secara *manual* maupun menggunakan aplikasi Rosetta 1.4.4.1 memberikan hasil yang sama. Sehingga disimpulkan bahwa implementasi aplikasi prediksi prestasi calon anggota kelompok *programming* STMIK *Pelita Nusantara* dengan menggunakan aplikasi Rosetta 1.4.4.1 telah dapat berjalan dengan baik

III. HASIL DAN DISKUSI

Berdasarkan hasil analisa yang dilakukan oleh peneliti maka diperoleh hasil bahwa penerapan aplikasi *Rough Set* untuk memprediksi prestasi calon anggota kelompok p*rogramming* STMIK *Pelita Nusantara* ini telah dapat berjalan dengan baik. Untuk meningkatkan akurasi maka diharapkan agar dapat menambah jumlah data dengan jumlah atribut yang semakin banyak sehingga *rule* yang dihasilkan akan semakin akurat.

IV. KESIMPULAN

Adapun kesimpulan dari hasil penelitian adalah sebagai berikut.

a. Penerapan algoritma *Rough Set* untuk memprediksi prestasi calon anggota kelompok *programming* STMIK *Pelita Nusantara* dengan menggunakan Rosetta 1.4.4.1 telah dapat berjalan dengan baik.
b. Untuk kesempurnaan dari aplikasi yang dirancang diharapkan agar variabel *input* dapat ditambah dengan melibatkan nilai variabel tridharma perguruan tinggi lainnya.
c. Untuk program aplikasi yang dirancang ini maka semakin banyak *rule* yang ada akan memberikan hasil yang semakin akurat.

V. DAFTAR PUSTAKA

[1] Davis, D.N, T.T. Nguyen, 2009, ”Generating and Verifying Risk Prediction Models Using data Mining: A Case Study From Cardiovascular Medicine”, IGI Global Inc
[2] Listiana, Nila, Wieik Anggraeni, dan Ahmad Muhlason, 2011, ”Implementasi Algoritma Rough Set untuk Deteksi dan Penanganan Dini Penyakit Sapi”, ITS, Surabaya
[3] Pramudiono, I., 2007, Apa itu data mining?, http://datamining.japati.net/cgi-bin/indodm.cgi?bacaarsip&1155527614&artikel, tanggal terakhir akses 16 Januari 2007
[4] Santosa, Budi, 2007, ”Data Mining Teknik Pemanfaatan Data untuk Keperluan Bisnis”, Penerbit Graha Ilmu, Yogyakarta
[5] Wirdasri, Dian dan Ahmad Calam, 2011, ”Penerapan Data Mining untuk Mengolah Data Penempatan Buku di Perpustakaan SMK TI PAB 7 Lubuk Pakam dengan Metode Association Rule”, Jurnal SAINTIKOM, STMIK Triguna Dharma