Perbandingan Algoritma Reduksi Tipe pada Fuzzy Tipe-2

Humaira
Jurusan Teknologi Informasi di Politeknik Negeri Padang Indonesia
mira.humaira@gmail.com

ABSTRACT

Type-2 Fuzzy became populer because it's capability handles uncertainty more better. It effects computational cost. This research insurance that there is algorithm having good performance ie. EIASC. It has short time if count of N is small.

Index Terms—EIASC Algorithm, Type-2 Fuzzy

ABSTRAK

Fuzzy Tipe-2 semakin populer karena kemampuannya untuk mengatasi ketidakpastian yang semakin baik. Hal ini berdampak pada biaya komputasi Fuzzy Tipe-2. Dari penelitian ini membuktikan ada algoritma yang memiliki performa yang paling bagus yaitu EIASC. Algoritma ini memiliki waktu proses relatif kecil jika jumlah N nya juga kecil.

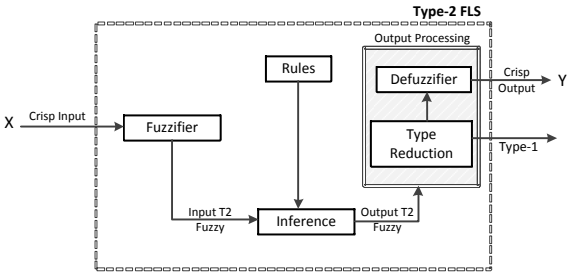
Kata Kunci—Algoritma EIASC, Fuzzy Tipe-2

1. PENDAHULUAN

Tipe-2 *Fuzzy Logic* (T2 FL) awalnya diperkenalkan oleh Zadeh. Saat itu kemampuan komputerisasi terbatas sehingga pemakaian *fuzzy* tipe-2 tidak begitu populer. Namun sejak tahun 2000-an *fuzzy* tipe-2 sudah banyak menjadi pilihan bagi peneliti meskipun membutuhkan komputasional yang kompleks.

T2 FL memiliki performansi yang lebih baik dari Tipe-1 *Fuzzy Logic* (T1 FL). Ada beberapa cara untuk memodelkan T2 FL yaitu general T2, Interval Type-2 (IT2) dan yang terbaru Quasi-T2[6]. General T2 FL sulit untuk diimplementasikan karena secara matematis masih rumit sehingga membuat komputasi menjadi kompleks. Kebanyakan peneliti menggunakan IT2 karena mudah untuk dipahami. Disamping itu komputasi dengan Interval T2 FL sangat mudah diatur yang menjadikan IT2 FL mudah diimplementasikan [*2*].

Diagram blok T2 FLS pada Gambar 1 mirip dengan T1 FLS. Perbedaan T2 FLS dengan T1 FLS hanya pada bagian proses keluaran. Pada T2 FLS dari himpunan *fuzzy* melalui dua tahapan proses sebelum menjadi himpunan tegas. Proses pengurangan tipe untuk merubah keluaran*fuzzy* tipe-2 menjadi tipe-1 kemudian dilanjutkan ke proses defuzzifikasi untuk mendapatkan himpunan tegas.



Gambar**1.** Sistem *fuzzy* tipe-2

Sebelum mendapatkan hasil akhir berupa nilai *crisp*, ada dua tahapan yang dilakukan yaitu (1) *Type-reduction*(tipe reduksi), merupakan langkah pertama pengolahan *output* dengan menghitung *centroid* dari IT2 dan (2) Defuzzifikasi, untuk memperoleh nilai *output* (*crisp*)[2].

Reduksi tipe dapat diselesaikan dengan beberapa metode antara lain [2][7] [8].

- Reduksi tipe Centroid(*center-of-sums*)

Centroid dihitung dari kombinasi area *output* yang sebelumnya didapatkan dari aktifasi setiap kaidah berdasarkan *firing strength*.

- Reduksi tipe pusat himpunan(*center-of-sets*)

Metode ini menghitung *centroid* setiap konsekuen dari kaidah yang aktif .

- Reduksi tipe ketinggian (*height*)

Keluaran setiap kaidah digantikan dengan *singleton* pada titik maksimum keanggotaan lalu dihitung *centroid* nya.

Karnik bersama Mendel telah memberikan solusi untuk mengatasi masalah komputasi dalam melakukan kalkulasi *centroid* pada IT2,yang dikenal dengan Algoritma KM (Karnik-Mendel). Ide utama algoritma KM yaitu untuk menemukan *switch point* untuk yl dan yr.

Beberapa tahun belakangan ini, banyak algoritma *type-reduction*memiliki efisiensi yang lebih baik dari algoritma KM. Adapun algoritma *type-reduction*selain KM yaitu **[***10***]**.

- *Enhanced Karnik Mendel Algorithms* (EKMA).
- EKMA *with New Initialization* (EKMANI).
- *Iterative Algorithm with Stop Condition* (IASC).
- *Enhanced IASC* (EIASC).

Penelitian ini akan membandingkan tiga dari algoritma diatas dalam hal kecepatan waktu komputasi.

2. LITERATUR

- Algoritma Karnik-Mendel (KMA)

KM terdiri dari dua bagian, satu untuk menghitung y_l dan lainnya untuk menghitung y_r. Ini merupakan ide utama KM untuk menemukan *switch point*untuk l dan r.

Algoritma KMA mendapatkan titik $y_l \; dan \; y_r$ berikut. KMA untuk menghitung y_l.
urutkan $\underline{x}_i \;\; (i = 1,2, \dots, N)$ dengan syarat $\underline{x}_1 < \underline{x}_2 < \cdots < \underline{x}_N$. indeks bobot W_i disesuaikan dengan indeks $\underline{x}_i$
Inisialisasi w_i dengan

$$\boldsymbol{w_i = \frac{\underline{w}_i + \overline{w}_i}{2} \qquad i = 1, 2, \dots, N}$$

kemudian hitung

$$\boldsymbol{y = \frac{\sum_{i=1}^{N} \underline{x}_i w_i}{\sum_{i=1}^{N} w_i}}$$

Cari *switch point* $l \; (1 \le l \le N-1)$

$$\boldsymbol{\underline{x}_l < y \le \underline{x}_{l+1}}$$

Set

$$\boldsymbol{w_i = \begin{cases} \overline{w}_i, & n \le l \\ \underline{w}_i, & n > l \end{cases}}$$

Dan hitung

$$\boldsymbol{y' = \frac{\sum_{i=1}^{N} \underline{x}_i w_i}{\sum_{i=1}^{N} w_i}}$$

Jika $y' = y$ maka berhenti dan set $y_l = y$. Jika tidak set $y = y'$ dan ulangi langkah ke-3
KMA untuk menghitung y_r.
urutkan $\overline{x}_i \;\; (i = 1,2, \dots, N)$ dengan syarat $\overline{x}_1 < \overline{x}_2 < \cdots < \overline{x}_N$. indeks bobot W_i disesuaikan dengan indeks $\overline{x}_i$
Inisialisasi w_i dengan

$$\boldsymbol{w_i = \frac{\underline{w}_i + \overline{w}_i}{2} \qquad i = 1, 2, \dots, N}$$

(2.1)

kemudian hitung

$$\boldsymbol{y = \frac{\sum_{i=1}^{N} \overline{x}_i w_i}{\sum_{i=1}^{N} w_i}}$$

Cari *switch point* $r \; (1 \le r \le N-1)$

$$\boldsymbol{\overline{x}_r < y \le \overline{x}_{r+1}}$$

Set

$$\boldsymbol{w_i = \begin{cases} \underline{w}_i, & n \le r \\ \overline{w}_i, & n > r \end{cases}}$$

Dan hitung

$$\boldsymbol{y' = \frac{\sum_{i=1}^{N} \overline{x}_i w_i}{\sum_{i=1}^{N} w_i}}$$

Jika $y' = y$ maka berhenti dan set $y_r = y$. Jika tidak set $y = y'$ dan ulangi langkah ke-3

Algoritma KM berjalan baik pada hampir semua kasus namun terkadang KM terjebak dalam looping tak terbatas pada saat y' tidak pernah sama dengan y.

- AlgoritmaEnhanced Karnik-Mendel (EKMA)

Algoritma EKMA mendapatkan titik $y_l \; dan \; y_r$ sebagai berikut. EKMA untuk menghitung y_l.
urutkan $\underline{x}_i \;\; (i = 1,2, \dots, N)$ dengan syarat $\underline{x}_1 < \underline{x}_2 < \cdots < \underline{x}_N$. indeks bobot W_i disesuaikan dengan indeks $\underline{x}_i$
set $l = [\frac{N}{2.4}]$ dan hitung

$$\boldsymbol{a = \sum_{i=1}^{l} \underline{x}_i \overline{w}_i + \sum_{i=l+1}^{N} \underline{x}_i \underline{w}_i}$$

$$\boldsymbol{b = \sum_{i=1}^{l} \overline{w}_i + \sum_{i=l+1}^{N} \underline{w}_i}$$

$$\boldsymbol{y = a/b}$$

cari $l' \in [1, N-1]$

$$\boldsymbol{\underline{x}_{l'} < y \le \underline{x}_{l'+1}}$$

jika $l' = l$, berhenti; jika tidak lanjutkan
hitung $s = sign(l' - l)$ dan

$$\boldsymbol{a' = a + s \sum_{i=\min(l,l')+1}^{\max(l,l')} \underline{x}_i (\overline{w}_i - \underline{w}_i)}$$

$$\boldsymbol{b' = b + s \sum_{i=\min(l,l')+1}^{\max(l,l')} (\overline{w}_i - \underline{w}_i)}$$

$$\boldsymbol{y' = a'/b'}$$

set $y = y', a = a', b = b' dan \; l = l'$. Ulangi langkah 3

EKMA untuk menghitung y_r.
urutkan $\overline{x}_i \;\; (i = 1,2, \dots, N)$ dengan syarat $\overline{x}_1 < \overline{x}_2 < \cdots < \overline{x}_N$. indeks bobot W_i disesuaikan dengan indeks $\overline{x}_i$
set $r = [\frac{N}{1.7}]$ dan hitung

$$\boldsymbol{a = \sum_{i=1}^{r} \overline{x}_i \underline{w}_i + \sum_{i=r+1}^{N} \overline{x}_i \overline{w}_i}$$

$$\boldsymbol{b = \sum_{i=1}^{r} \underline{w}_i + \sum_{i=r+1}^{N} \overline{w}_i}$$

$$\boldsymbol{y = a/b}$$

cari $r \in [1, N-1]$

$$\boldsymbol{\overline{x}_{r'} < y \le \overline{x}_{r'+1}}$$

jika $r' = r$ maka berhenti. jika tidak lanjutkan proses berikutnya
hitung $s = sign(r' - r)$ dan

$$\boldsymbol{a' = a - s \sum_{i=\min(r,r')+1}^{\max(r,r')} \overline{x}_i (\overline{w}_i - \underline{w}_i)}$$

$$\boldsymbol{b' = b - s \sum_{i=\min(r,r')+1}^{\max(r,r')} (\overline{w}_i - \underline{w}_i)}$$

$$\boldsymbol{y' = a'/b'}$$

set $= y', a = a', b = b' dan \; r = r'$. ulangi langkah 3

EKMA merupakan perbaikan dari KMA. Pertama inisialisasi yang lebih baik sehingga mengurangi jumlah iterasi. Kedua kondisi berhenti diubah untuk menghilangkan iterasi yang tidak penting. Terakhir teknik perhitungan subtle berfungsi untuk mengurangi biaya komputasi setiap iterasi.

- EnhancedIterative Algorithm with Stop Condition (EIASC)

Algoritma EIASC mendapatkan nilai $y_l \; dan \; y_r$ sebagai berikut. Langkah-langkah menghitung y_l.
Urutkan $\underline{x}_i \;\; (i = 1,2, \dots, N)$ dengan syarat $\underline{x}_1 < \underline{x}_2 < \cdots < \underline{x}_N$. indeks bobot W_i disesuaikan dengan indeks $\underline{x}_i$.
Inisialisasi

$$\boldsymbol{a = \sum_{i=1}^{N} \underline{x}_i \underline{w}_i}$$

$$\boldsymbol{b = \sum_{i=1}^{N} \underline{w}_i}$$

$$\boldsymbol{y_l = \underline{x}_N}$$

$$\boldsymbol{l = 0}$$

Hitung

$\boldsymbol{l = l + 1}$

$\boldsymbol{a = a + \underline{x}_l(\overline{w}_l - \underline{w}_l)}$

$\boldsymbol{b = b + \overline{w}_l - \underline{w}_l}$

$\boldsymbol{c = a/b}$

Jika $c > y_l$, set $l = l - 1$ dan berhenti; jika tidak set $y_l = c$ dan ulangi langkah 3.

Langkah-langkah menghitung y_r.

urutkan $\overline{x}_i$ $(i = 1,2, \dots, N)$ dengan syarat $\overline{x}_1 < \overline{x}_2 < \cdots < \overline{x}_N$. indeks bobot W_i disesuaikan dengan indeks $\overline{x}_i$

inisialisasi

$\boldsymbol{a = \sum_{i=1}^{N} \overline{x}_i \underline{w}_i}$

$\boldsymbol{b = \sum_{i=1}^{N} \underline{w}_i}$

$\boldsymbol{y_r = \overline{x}_1}$

$\boldsymbol{r = N}$

hitung

$\boldsymbol{a = a + \overline{x}_r(\overline{w}_r - \underline{w}_r)}$

$\boldsymbol{b = b + \overline{w}_r - \underline{w}_r}$

$\boldsymbol{c = a/b}$

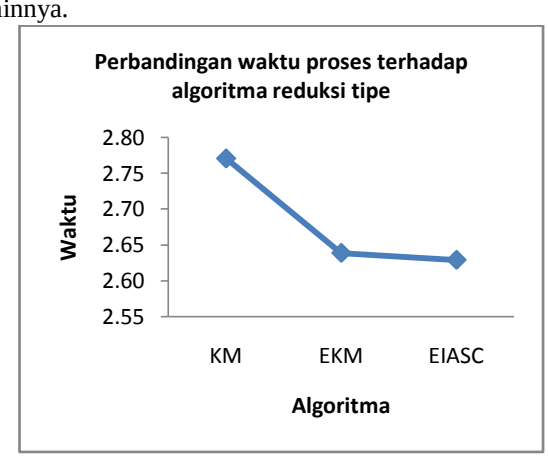
$\boldsymbol{r = r - 1}$

jika $c < y_r$, set $r = r + 1$ dan berhenti, jika tidak set $y_r = c$ dan ulangi langkah 3.

3. PEMBAHASAN

Ada banyak algoritma untuk menyelesaikan masalah reduksi tipe. Pada simulasi ini akan menggunakan tiga algoritma yaitu Karnik-Mendel (KM), *Enhanced* Karnik-Mendel (EKM) dan *Enhanced Iterative Algorithm with Stop Condition*(EIASC).

Dari ketiga algoritma tersebut dapat dilihat pada. Gambar2. Dari grafik menunjukkan bahwasanya KM memberikan waktu proses lebih lama dibandingkan EKM dan EIASC. Algoritma KM memiliki waktu proses jauh lebih lama dibandingkan dua algoritma lainnya.



Gambar**2.** Perbandingan rata-rata waktu proses terhadap algoritma reduksi tipe

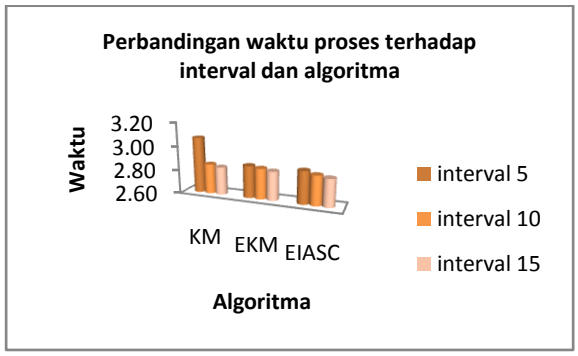
Sistem fuzzy tipe-2 ini menggunakan jenis fungsi keanggotaan interval. Fungsi keanggotaan interval memiliki keanggotaan upper (UMF) dan juga keanggotaan lower (LMF).Simulasi berikutnya melihat pengaruh waktu pemakaian algoritma terhadap interval. Interval disini merupakan jarak antara UMF dan LMF.

TABEL 1. WAKTU PROSES TERHADAP INTERVAL

	interval 5	interval 10	interval 15
KM	3,07	2,85	2,83
EKM	2,87	2,86	2,84
EIASC	2,87	2,85	2,83

Dari tabel 1 terlihat bahwasanya interval terkecil yaitu 5 sangat mempengaruhi pemilihan dari algoritma reduksi tipe. Semakin besar interval maka banyak N juga semakin besar.

Algoritma KM memiliki waktu paling lama yaitu 3.07. Seperti dapat diamati pada gambar 3. Hal ini disebabkan bahwasanya semua algoritma reduksi tipe di atas yang mempunyai tingkat efisiensi terbaik yaitu EIASC jika N<100. Penelitian mengenai perbandingan performansi algoritma tersebut menyebutkan bahwa algoritma EIASC dapat menghemat biaya komputasional sampai 50% [5].



Gambar 3. Perbandingan Waktu Proses Terhadap Interval dan Algoritma Reduksi Tipe

Berdasarkan penelitian sebelumnya terbukti bahwa algoritma EIASC dapat menghemat waktu komputasi sampai 50%.

4. KESIMPULAN DAN SARAN

Dari simulasi yang dilakukan dapat ditarik kesimpulan bahwa penggunaan algoritma reduksi tipe untuk komputasi N dalam jumlah besar tidak mempengaruhi waktu proses. Namun jika N nya kecil sebaiknya menggunakan algoritma EIASC untuk mendapatkan waktu komputasi yang relatif sedikit.

Diharapkan untuk penelitian berikutnya dapat diamati seberapa besar pengaruh N terhadap waktu komputasi.

5. DAFTAR PUSTAKA

[1] J.S.R Jang, C.T Sun, and E Mizutani, *Neuro fuzzy and Soft computing*. US: Prentice hall, 1997.

[2] Jerry M Mendel, Robert I Jhon, and Feilong Liu, "Interval type-2 fuzzy logic systems made simple," *IEEE transactions on fuzzy systems vol.14 no.6*, pp. 808-821, 2006.

[3] Jerry M Mendel, "Type-2 Fuzzy sets and systems:How to learn

about them," *IEEE SCM eNewsletter*, 2009.

[4] Jerry M.Mendel and Robert I. Bob Jhon, "Type-2 Fuzzy Sets Made Simple," *IEEE Transaction on Fuzzy Systems Vol.10 No.2*, pp. 117-127, 2002.

[5] Dongrui Wu and Maowen Nie, "Comparison and Practical Implementation of type Reduction Algorithms for Type-2 Fuzzy Sets and Systems," 2011.

[6] Jerry M Mendel and Feilong Liu, "On New Quasi-Type-2 Fuzzy Logic System," Proceeding on Fuzzy Systems, June 2008.

[7] Dongrui Wu and Jerry M Mendel, "Enhanced Karnik-Mendel Algorithms," IEEE on Fuzzy Systems, vol. 17, no. 4, pp. 923-934, August 2009.

[8] Nilesh N Karnik, Jerry M Mendel, and Qilian Liang, "Type-2 Fuzzy Logic Systems," IEEE Transaction on Fuzzy Systems, vol. 7, no. 6, pp. 643-658, Dec 1999.

[9] Juan R Castro, Oscar Castillo, and Luis G Martinez, "Interval Type-2 Fuzzy Logic Toolbox," Engineering Letter, August 2007.

[10] Dongrui Wu, "A brief tutorial on Interval type-2 fuzzy sets and systems," 2011.

[11] Dongrui Wu, "On the Fundamental Differences between Interval Type-2 and Type-1 Fuzzy Logic Controllers," IEEE Transactions on Fuzzy Systems, pp. 1-17, 2011.

[12] Matlab R2010b. (2010) Documentation Fuzzy Logic.

Humairadilahirkan di kota Padang pada tanggal 19 Maret 1981. Penulis menamatkan pendidikan S1 di Institut Teknologi Telkom jurusan Teknik Informatika tahun kelulusan 2004. Kemudian melanjutkan ke jenjang S2 jurusan Teknik Informatika sub Teknologi Informasi di ITB dan di wisuda pada juli 2012.

Dia bekerja sebagai salah satu dosen tetap jurusan Teknologi Informasi pada Politeknik Negeri Padang sejak tahun 2006 sampai sekarang.