# BAB 4
# IMPLEMENTASI

Pada bab ini akan dijelaskan secara terperinci proses implementasi dari hasil perancangan penelitian segmentasi dokumen yang telah dijelaskan pada Bab 3. Proses implementasi yang akan dijelaskan terdiri dari implementasi proses pengolahan data yang dilakukan sebelum data tersebut siap digunakan pada percobaan, dan implementasi *genetic algorithm* sebagai metode yang digunakan pada penelitian segmentasi dokumen ini. Keseluruhan proses implementasi ini akan dilakukan dengan menggunakan bahasa pemrograman JAVA dan *library* ECJ 18 yang juga berbasiskan JAVA untuk implementasi *genetic algorithm*. Hasil yang diperoleh dari implementasi proses pengolahan data adalah dokumen-dokumen yang siap digunakan pada percobaan. Hasil yang diperoleh dari implementasi *genetic algorithm* adalah sistem yang dapat melakukan segmentasi dokumen secara otomatis dan hasil segmentasi dari masing-masing *test case* yang diujikan.

## 4.1 PENGOLAHAN DATA

Sebelum kumpulan dokumen-dokumen yang dimiliki digunakan pada percobaan ini, perlu dilakukan beberapa persiapan pada dokumen-dokumen tersebut. Persiapan-persiapan tersebut selain bertujuan untuk mempersiapkan data agar siap digunakan, juga bertujuan untuk membantu menghasilkan hasil segmentasi yang baik. Keseluruhan sistem untuk melakukan proses persiapan pada dokumen-dokumen tersebut dibuat dengan menggunakan bahasa pemrograman JAVA.

### 4.1.1 PENGGABUNGAN DOKUMEN

Dokumen yang digunakan sebagai bahan percobaan adalah dokumen-dokumen hasil penggabungan dari beberapa dokumen (dokumen hasil *append* beberapa dokumen). Sistem segmentasi dokumen ini akan berusaha menemukan letak segmen antar dokumen-dokumen penyusunnya. Proses pemilihan jenis dokumen dan jumlah dokumen yang akan di-*append* akan mengikuti hasil rancangan percobaan yang telah dijelaskan pada subbab 3.2.2. Pemilihan

dokumen yang akan di-*append* dilakukan secara *random* dari kumpulan dokumen yang telah ditentukan sesuai dengan perancangan. P*seudocode* yang digunakan dalam pembuatan sistem untuk melakukan penggabungan dokumen-dokumen tersebut dapat dilihat pada Gambar 4.1 sebagai berikut:

```
function  APPEND-DOCUMENT (directoryPath, numberOfDocumentUsed,
          numberOfDocument) return  documents
          inputs : directoryPath, directory that contains documents
                   numberOfDocumentUsed, number of document that will be
                   appended
                   numberOfDocument, number of finished document
          fileName[] ← file names in specified directory
          for     i ← 1 to numberOfDocument
                  document ← empty document
                  for      j← 1 to numberOfDocumentUsed
                           x ← random number
                           read file fileName[x]
                           append file to document
          return  documents
```

**Gambar 4.1 Pseudocode untuk proses *append* dokumen**

Sistem yang digunakan akan menghasilkan dua buah *output* dokumen, yaitu dokumen yang akan digunakan pada percobaan, dan dokumen acuan/kunci dimana terdapat informasi letak segmen sebenarnya. Dokumen acuan tersebut akan digunakan pada proses evaluasi hasil segmentasi, yaitu untuk menentukan *precision* dan *recall* dari hasil segmentasi yang diberikan oleh sistem segmentasi dokumen ini. Dengan pemilihan komponen dokumen secara *random* diharapkan dapat membantu menghasilkan komponen dokumen yang cukup bervariasi.

### 4.1.2 PEMBUANGAN *STOPWORD*, *STEMMING*, DAN PEMISAHAN ANTAR KALIMAT

Setelah menyiapkan dokumen-dokumen yang akan digunakan pada percobaan ini, berikutnya dilakukan pengolahan pada dokumen-dokumen tersebut. Pengolahan yang dilakukan pada dokumen-dokumen tersebut diantaranya pembuangan *stopword*, *stemming*, dan pemisahan antar kalimat. Persiapan-

persiapan ini dilakukan untuk mendukung proses segmentasi dokumen yang dilakukan.

Salah satu hal yang dilakukan pada proses segmentasi dokumen adalah penghitungan nilai *similarity* antar kalimat-kalimat penyusun suatu dokumen. Sebelum melakukan penghitungan nilai *similarity* tersebut, perlu dilakukan penghitungan bobot tiap kata pada dokumen tersebut untuk metode *cosine similarity*. Hal ini dilakukan untuk menilai seberapa penting kata-kata tersebut pada suatu dokumen. Penghitungan bobot kata-kata ini dilakukan berdasarkan frekuensi kemunculan kata-kata tersebut. Kata-kata yang hampir selalu muncul disetiap dokumen tentunya bukan merupakan kata-kata yang menunjukan karakteristik suatu dokumen, dengan kata lain kata-kata tersebut dianggap kurang penting. Dengan metode pembobotan kata ini maka kata-kata yang sering muncul tersebut (biasanya dikenal dengan *stopword*) akan memiliki bobot yang kecil dibandingkan dengan kata-kata lain yang hanya muncul di beberapa dokumen saja. Untuk mengefisienkan proses penghitungan bobot, maka kata-kata yang sudah teridentifikasi sebagai kumpulan kata-kata yang termasuk dalam kategori *stopword* pada suatu bahasa akan langsung dibuang/dihilangkan dari dokumen karena kata-kata tersebut sudah tentu akan memiliki bobot kecil dibandingkan kata-kata lainnya (berdasarkan frekuensi kemunculannya). Beberapa contoh kata-kata yang termasuk dalam *stopword* dapat dilihat pada Gambar 4.2 sebagai berikut:

yang, di, dan, itu, dengan, untuk, tidak, ini, dari, dalam, akan, pada, juga, saya, ke, karena, tersebut, bisa, ada, mereka, lebih, kata, tahun, sudah, atau, saat, oleh, menjadi, orang, ia, telah, adalah, seperti, sebagai, bahwa, dapat, para, harus, namun, kita, dua, satu, masih, hari, hanya, mengatakan, kepada, kami, setelah, melakukan, lalu, belum, lain, dia, kalau, terjadi, banyak, menurut, anda, hingga, tak, baru, beberapa, ketika, saja, jalan, sekitar, secara, dilakukan, sementara, tapi, sangat, hal, sehingga, seorang, bagi, besar, lagi, selama, antara, waktu, sebuah, jika, sampai, jadi, terhadap, tiga, serta, pun, salah, merupakan, atas, sejak, membuat, baik, memiliki, kembali, selain, tetapi, pertama, kedua, memang, pernah, apa, mulai, sama, tentang, bukan, agar, semua, sedang, kali, kemudian, hasil, sejumlah, juta, persen.

**Gambar 4.2 Contoh kata-kata yang termasuk *stopword***

Daftar lengkap kata-kata yang termasuk kedalam *stopword* bahasa Indonesia dapat dilihat pada Lampiran 1.

Untuk membantu meningkatkan hasil proses penghitungan nilai *similarity* antar kalimat penyusun suatu dokumen, maka dilakukan proses *stemming* pada dokumen-dokumen yang akan digunakan pada percobaan ini. *Stemming* adalah suatu proses untuk membentuk suatu kata menjadi bentuk dasar dari kata tersebut (kata dasar). Pada percobaan ini *stemmer* yang digunakan adalah *stemmer* dikembangkan oleh Muhammad Ichsan dan Mirna Adriani (Ichsan & Adriani, 2006). Contoh hasil keluaran dari *stemmer* yang digunakan pada percobaan ini dapat dilihat pada Gambar 4.3:


**Gambar 4.3 Contoh *output* proses *stemming***

Dengan proses *stemming* ini maka semua kata akan dibentuk menjadi kata dasarnya, sehingga dapat membantu mengoptimalkan penghitungan nilai *similarity* karena kata-kata yang bermakna hampir sama namun berbeda karena adanya penambahan imbuhan akan sama-sama dibentuk menjadi kata dasar yang sama (kata "memakan" dan "dimakan" akan diubah menjadi kata "makan"). Contoh kegunaan proses *stemming* pada penghitungan nilai *similarity* dapat dilihat pada Gambar 4.4 sebagai berikut:

Tanpa proses *stemming*

Kalimat 1: saya memakan bubur

Kalimat 2: saya makan bubur

Nilai similarity kedua kalimat diatas adalah < 1

Dengan proses *stemming*

Kalimat 1: saya memakan bubur → saya makan bubur

Kalimat 2: saya makan bubur → saya makan bubur

Nilai similarity kedua kalimat diatas = 1

**Gambar 4.4 Contoh kegunaan proses *stemming* pada penghitungan nilai *similarity***

Letak segmen/batas antar dokumen-dokumen penyusun terletak diantara kalimat-kalimat tertentu pada dokumen tersebut. Untuk mempermudah representasi letak segmen pada suatu dokumen, akan dilakukan proses pemisahan pada setiap kalimat sehingga menghasilkan dokumen dengan hanya satu kalimat disetiap barisnya. Contoh dokumen yang dihasilkan setelah proses pemisahan antar kalimat dapat dilihat pada Gamabr 4.5 sebagai berikut:

Kontrol transaksi keuangan yang baik harus bisa menghasilkan pos terbuka seminimal mungkin. Ini hanya bisa dihasilkan jika diantaranya bank mempunyai Sistem Rekonsiliasi yang baik dan terpadu. Pada dasarnya nilai ekonomis suatu sistem informasi bisa dievaluasi dengan melihat dampak ekonomisnya. Demikian pula Sistem Rekonsiliasi pada sebuah bank ataupun institusi keuangan lainnya. Metode tradisional biasanya berdasar pada model ROI (Return On Investment).

↓

Kontrol transaksi keuangan yang baik harus bisa menghasilkan pos terbuka seminimal mungkin.
Ini hanya bisa dihasilkan jika diantaranya bank mempunyai Sistem Rekonsiliasi yang baik dan terpadu.
Pada dasarnya nilai ekonomis suatu sistem informasi bisa dievaluasi dengan melihat dampak ekonomisnya.
Demikian pula Sistem Rekonsiliasi pada sebuah bank ataupun institusi keuangan lainnya.
Metode tradisional biasanya berdasar pada model ROI (Return On Investment).

**Gambar 4.5 Contoh dokumen setelah dilakukan proses pemisahan antar kalimat**

Proses pemisahan antar kalimat ini pun akan membantu mempermudah penerapan proses segmentasi dokumen, terutama pada tahap penghitungan nilai *similarity* karena proses penghitungan nilai *similarity* tersebut berdasarkan pada kalimat-kalimat penyusun dokumen tersebut.

Ketiga proses yang telah dijelaskan sebelumnya akan dilakukan secara bersamaan dengan menggunakan sebuah sistem pengolah dokumen. *Pseudocode* sistem pengolah dokumen yang digunakan pada percobaan ini dapat dilihat pada Gambar 4.6 sebagai berikut:

```
function PREPROCESSING (directoryPath, dictionaryFile, stopwordListFile) return
    documents
  inputs : directoryPath, directory that contains document
           dictionaryFile, file that contains reference word for stemming
           stopwordListFile, file that contains list of stopword
  repeat
      sentence ← empty string
      for i from 1 to number of Line in document
          line ← read line in a document
          for j from 1 to number of word in line
              if (word contains ./?/! && next word start with uppercase)
                  remove punctuation and stem word
                  if (word is stopword)
                      delete word
                  else
                      append word to sentence
                  write sentence to output document
                  sentence ← empty string
              else
              remove punctuation and stem word
                  if (word is stopword)
                      delete word
                  else
              append word to sentence
  until  all documents in directoryPath
  return documents
```

**Gambar 4.6 Pseudocode untuk proses pengolahan dokumen**

## 4.2 ECJ 18

Pada tahap implementasi penelitian ini digunakan sebuah *library* untuk membantu melakukan implementasi *genetic algorithm*. *Library* yang digunakan adalah ECJ 18, yang merupakan suatu *library* untuk penelitian dibidang *evolutionary computation* yang dikembangkan dengan menggunakan bahasa pemrograman JAVA [2]. *Library* ECJ 18 ini mengutamakan fleksibilitas dan efisiensi dalam pengembangannya, sehingga diharapkan dapat memudahkan para pengguna untuk melakukan modifikasi pada saat implementasi. Pada ECJ 18, semua konfigurasi yang dibutuhkan ditulis pada suatu *parameter file*, yang mudah untuk dimodifikasi sesuai dengan kebutuhan. ECJ 18 dikembangkan pada lab *evolutionary computation* George Mason Univeristy.

### 4.2.1 REPRESENTASI INDIVIDU/*CHROMOSOME* DAN *GENETIC OPERATOR*

Seperti telah dijelaskan pada subbab 3.3.1 mengenai representasi individu dan *genetic operator* yang digunakan pada percobaan ini, representasi yang digunakan merupakan suatu *binary string* yang setiap elemennya merepresentasikan kalimat pada dokumen *input* proses segmentasi ini. Secara terperinci representasi yang digunakan adalah untuk Dokumen dengan $ns$ kalimat akan direpresentasikan dengan vektor biner $\vec{x}$ dengan $(ns-1)$ elemen. $x_i = 1$ menunjukan bahwa terdapat segmen diantara kalimat ke $i$ dan ke $i+1$ dan sebaliknya untuk $x_i = 0$ menunjukan bahwa tidak terdapat segmen diantara kalimat ke $i$ dan ke $i+1$. Contoh representasi individu yang digunakan pada penelitian ini dapat dilihat pada Gambar 4.7 sebagai berikut:

[2] Lihat http://www.cs.gmu.edu/~eclab/projects/ecj/

**Dokumen:**

Pagi hari ibu pergi kepasar bersama adik
Ketika ibu pergi ayah sedang minum kopi
Presiden Indonesia sekarang adalah SBY
SBY merupakan presiden ke enam Indonesia

**Contoh Representasi Individu :** 0 1 0

**Penjelasan :**

Terdapat segmen diantara kalimat ke 2 dan ke 3 pada dokumen tersebut, sehingga dokumen hasil segmentasi adalah sebagai berikut:

pagi hari ibu pergi ke pasar bersama adik
ketika ibu pergi ayah sedang minum kopi
---
presiden Indonesia sekarang adalah sby
sby merupakan presiden keenam Indonesia

**Gambar 4.7 Contoh representasi individu pada penelitian**

Pada *parameter file* yang digunakan sebagai *input*, representasi individu ini dinyatakan dalam dua baris *statement*, yaitu:

pop.subpop.0.species.ind = ec.vector.BitVectorIndividual

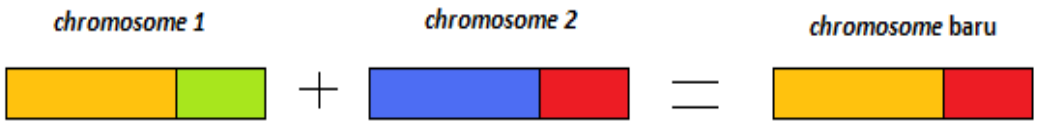
pop.subpop.0.species.genome-size = *x*

Dengan *x* menyatakan besar/panjang dari *chromosome binary string* yang digunakan pada proses segmentasi dokumen.

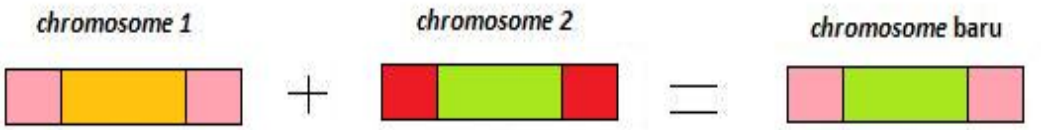
*Genetic Operator* yang digunakan pada pecobaan ini terdiri dari *tournament selection* untuk proses seleksi dan *single point crossover* atau *two point crossover* untuk proses *crossover*. *Tournament selection* adalah suatu proses pemilihan individu berdasarkan *fitness score* yang dimiliki oleh masing-masing individu. Penggunaan metode seleksi ini pada ECJ 18 dilakukan dengan menuliskan *statement* sebagai berikut pada *parameter file*:

pop.subpop.0.species.pipe.source.0.source.0 = ec.select.TournamentSelection

*Crossover* yang digunakan pada percobaan ini adalah *single point crossover* dan *two point crossover*. *Single point crossover* adalah proses persilangan dengan menggunakan satu buah batas sebagai batasan persilangan, sedangkan *two point crossover* adalah proses persilangan dengan menggunakan dua buah batas sebagai batas persilangan. Contoh proses yang terjadi pada *single point crossover* dan *two point crossover* dapat dilihat pada Gambar 4.8 dan Gambar 4.9 sebagai berikut:




**Gambar 4.8 *Single point crossover***




**Gambar 4.9 *Two point crossover***

Penggunaan metode *crossover* ini pada ECJ 18 dilakukan dengan menuliskan *statement* sebagai berikut pada *parameter file*:

pop.subpop.0.species.pipe.source.0 = ec.vector.breed.VectorCrossoverPipeline

pop.subpop.0.species.crossover-type = *one/two*

Dengan *one* menyatakan bahwa metode *crossover* yang digunakan pada proses ini adalah *single point crossover*, atau *two* untuk tipe *two point crossover*.

### 4.2.2 *FITNESS FUNCTION*

Pada percobaan ini, dilakukan percobaan-percobaan dengan variasi terhadap metode penghitungan *sentece similarity* dan *fitness function*. Metode Penghitungan *sentence similarity* yang digunakan terdiri dari *cosine similarity* dengan pembobotan TF-IDF dan TF-IDF (*Word* Frequency), dan *dice coefficient*. *Fitness function* yang digunakan terdiri dari 4 jenis *fitness function*, yaitu: optimisasi *internal cohesion*, optimisasi *dissimilarity*, optimisasi *internal cohesion*

dan *dissimilarity* dengan kombinasi linear, dan optimisasi *internal cohesion* dan *dissimilarity* dengan pendekatan SPEA 2.

TF-IDF dan TF-IDF (*Word Frequency*) merupakan dua buah metode penghitungan bobot suatu kata dengan memperhitungkan jumlah kemunculan kata tersebut pada setiap dokumen yang ada. Hal ini dapat membatu untuk menentukan seberapa penting kata tersebut untuk merepresentasikan topik suatu dokumen. Semakin tinggi bobot suatu kata pada suatu dokumen, berarti kata tersebut cukup berperan dalam menentukan topik suatu dokumen. *Pseudocode* yang digunakan pada implementasi pembobotan kata dengan TF-IDF dan TF-IDF (*Word Frequency*) dapat dilihat pada Gambar 4.10 dan Gambar 4.11 sebagai berikut:

```
function weightingTFIDF(sentence) returns weight
        inputs: sentence, a number of words
        repeat
        word ←single word in sentence
        sumInSentence ← calculate number of word in sentence
        sumInAllSentence ← calculate number of word in all sentece
        weight = sumInSentence * (1/ sumInAllSentence)
        until all the words in the sentece has been evaluated
        return weight
```

**Gambar 4.10 *Pseudocode* untuk proses pembobotan kata dengan TF-IDF**

```
function weightingTFIDFWordFreq(sentence) returns weight
        inputs: sentence, a number of words
        repeat
        word ←single word in sentence
        sumInSentence ← calculate number of word in sentence
        numberOfWord ← calculate total word in all sentence
        weight = sumInSentence * (numberOfWord / sumInSentence)
        until all the words in the sentece has been evaluated
        return weight
```

**Gambar 4.11 *Pseudocode* untuk proses pembobotan kata dengan (*Word Frequency*)**

*Cosine similarity* dan *dice coefficient* merupakan dua buah metode untuk mengukur tingkat kemiripan antar kalimat. Metode ini digunakan untuk menghitung tingkat keterkaitan antara satu kalimat dengan kalimat lainnya. Pada

penghitungan *similarity* dengan metode *dice coefficient* tidak digunakan proses pembobotan kalimat, karena pada metode ini yang diperhitungkan adalah jumlah kata-kata yang sama pada kalimat-kalimat yang dihitung nilai *similarity*-nya. *Pseudocode* yang digunakan untuk implementasi *cosine similarity* dan *dice coefficient* pada percobaan ini dapat dilihat pada Gambar 4.12 dan Gambar 4.13 sebagai berikut:

```
function diceCoefficient (sentence1, sentence2) returns similarity
        inputs: sentence1, a sentence in a document
                sentence2, a sentence in a document
        numOfSimWord ← calculate number of similar word in sentence1 and
        sentence2
        numOfWord1 ←calculate number of word in sentence1
        numOfWord2 ←calculate number of word in sentence2
        similarity = (2 * numOfSimWord) / (numOfWord1 + numOfWord2)
        return similarity
```

**Gambar 4.12 *Pseudocode* penghitungan *similarity* dengan *cosine similarity***

```
function cossineSimilarity(sentence1, sentence2) returns similarity
        inputs: sentence1, a sentence in a document
                sentence2, a sentence in a document
        repeat
        sumOfSimWord += weight of word in sentence1 * weight of word in sentence2
        until all the same word in the sentences have been evaluated
        repeat
        totalWeight1 ← sum of square weight for every word in sentence1
        until all word in sentence1 have been evaluated
        repeat
        totalWeight2 ← sum of square weight for every word in sentence2
        until all word in sentence2 have been evaluated
        similarity = sumOfSimWord / sqrt(totalWeight1 * totalWeight2)
        return similarity
```

**Gambar 4.13 *Pseudocode* penghitungan *similarity* dengan *dice coefficient***

*objective function* yang digunakan pada percobaan ini menggunakan *objective function* yang diterapkan pada penelitian SegGen (Lamprier et al., 2007), yaitu *internal cohesion* dan *dissimilarity*. Beberapa variasi yang terapkan pada

*fitness function* percobaan ini dibandingkan dengan pada penelitian SegGen adalah optimisasi *internal cohesion*, optimisasi *dissimilarity*, kombinasi linear antara *internal cohesion* dan *dissimilarity*, dan pendekatan SPEA 2, dimana pada penelitian SegGen hanya diterapkan pendekatan SPEA sebagai *fitness function*. *Pseudocode* yang digunakan untuk melakukan penghitungan nilai *internal cohesion* dan *dissimilarity* pada penelitian ini dapat dilihat pada Gambar 4.14 sebagai berikut:

```
function internalCohesion(document, chromosome) returns internalCohesion
        inputs: document, document used  in this proses
                chromosome, genetic representation
        numberOfSegment ← calculate number of segment from chromosome
        for i ← 1 to numberOfSegment
                sumSim ← sum of all combination of sentence similarity in a segment
                sumCouple ← sum of number of possible couple of sentence in a segment
        internalCohesion = sumSim / sumCouple
        return internalCohesion
```

```
function dissimilarity (document, chromosome) returns dissimilarity
        inputs: document, document used  in this proses
                chromosome, genetic representation
        for i ← 1 to number of segment in document - 1
                sumSimSeg += segmentSimilarity(document, segment i, segment i + 1)
        dissimilarity = 1 – (sumSimSeg / number of segment in document – 1)
        returns dissimilarity
```

```
function segmentSimilarity (document, segment1, segment2) return simSeg
        inputs  document, document used  in this proses
                segment1, several sentences that build a segment of document
                segment2, several sentences that build a segment of document
        for i ← 1 to number of sentence in segment1
                for j ← 1 to number of sentence in segment2
                        sim ← calculate similarity between sentence i and sentence j
                        sumSim += sim
        simSeg = sumSim / (number of sentence in segment1*number of sentence in
        segment2)
        return simSeg
```

**Gambar 4.14 *Pseudocode* penghitungan *internal cohesion* dan *dissimilarity***

# BAB 5
# HASIL DAN PEMBAHASAN

Pada bab ini akan dijelaskan mengenai hasil dari percobaan segmentasi dokumen yang telah dilakukan. Pembahasan yang dilakukan mencakup perbandingan hasil dari variabel eksperimentasi (*fitness function*, metode penghitungan *similarity*, *genetic operator*) dan variasi *test case*. Pada pembahasan terakhir akan dibahas perbandingan hasil segmentasi antara metode *genetic algorithm* (menggunakan konfigurasi *genetic operator* terbaik) dan metode Texttiling. Setelah membahas semua hasil percobaan maka akan dilakukan pembahasan mengenai analisa kesalahan dan rangkuman hasil percobaan.

## 5.1 RAGAM ASPEK PEMBAHASAN HASIL SEGMENTASI DOKUMEN

Seperti telah dijelaskan pada subbab 3.2.2, 3.3 dan 3.3.2 mengenai variabel-variabel yang digunakan pada percobaan ini, terdapat 4 jenis variabel jumlah segmen yang diujikan, 3 jenis variabel kemiripan dokumen penyusun, 17 jenis variabel *fitness function* yang digunakan, 3 jenis variabel metode penghitungan similarity, 3 jenis variabel probabilitas mutasi, 2 jenis variabel tipe *crossover*, 3 jenis variabel jumlah iterasi, dan 4 jenis variabel ukuran populasi. Pada percobaan terakhir akan dibandingkan hasil segmentasi antara metode *genetic algorithm* dan metode Texttiling. Variasi pada jumlah segmen bertujuan untuk mengetahui hasil segmentasi jika metode segmentasi ini diterapkan pada dokumen dengan jumlah segmen yang beragam (kecil – besar), pembahasan lebih lanjut akan dilakukan pada subbab 5.7. Variasi pada kemiripan dokumen penyusun dokumen bertujuan untuk menguji kemampuan metode segmentasi ini jika dihadapkan pada dokumen-dokumen penyusun dengan tingkat kemiripan topik beragam (sangat mirip, mirip, dan tidak mirip), pembahasan lebih lanjut akan dilakukan pada subbab 5.8. Variasi pada *fitness function* bertujuan untuk menentukan metode segmentasi yang paling baik, pembahasan lebih lanjut akan dilakukan pada subbab 5.2 dan subbab 5.5. Variasi pada metode penghitungan *similarity* bertujuan untuk menentukan metode penghitungan *similarity* yang memberikan hasil terbaik, pembahasan lebih lanjut akan dilakukan pada subbab

5.3. Variasi pada probabilitas mutasi, tipe *crossover*, ukuran populasi, jumlah iterasi bertujuan untuk menentukan kombinasi variabel-variabel tersebut yang memberikan hasil terbaik, pembahasan lebih lanjut akan dilakukan pada subbab 5.4 dan subbab 5.6. Pembahasan hasil percobaan membandingkan hasil segmentasi antara metode *genetic algorithm* dengan Texttiling akan dilakukan pada subbab 5.9. Daftar rangkuman keseluruhan variabel percobaan yang diterapkan pada eksperimen ini dapat dilihat pada Tabel 5.1 berikut ini:

**Tabel 5.1 Tabel rangkuman keseluruhan variabel percobaan**

| Variabel | Nilai |
|---|---|
| ***Fitness function*** | Dissimilarity |
| | Internal Cohesion |
| | Kombinasi Linear 1:1 |
| | Kombinasi Linear 1:2 |
| | Kombinasi Linear 1:6 |
| | Kombinasi Linear 1:8 |
| | SPEA2 - 1:1 |
| | SPEA2 - 1:2 |
| | SPEA2 - 1:3 |
| | SPEA2 - 1:4 |
| | SPEA2 - 1:5 |
| | SPEA2 - 1:6 |
| | SPEA2 - 1:7 |
| | SPEA2 - 1:8 |
| | SPEA2 - 1:9 |
| | SPEA2 - 1:10 |
| | SPEA2 - 1:100 |
| **Jumlah iterasi** | 100 |
| | 500 |
| | 1000 |
| **Ukuran populasi** | 20 |
| | 30 |
| | 40 |

| | |
|---|---|
| | 50 |
| **Metode Penghitungan *similarity*** | Dice Coefficient |
| | Cosine Similarity - TF IDF |
| | Cosine Similarity - TF IDF (Word Frequency) |
| **Tipe *Crossover*** | One Point Crossover |
| | Two Point Crossover |
| **Probabilitas Mutasi** | 0.05 |
| | 0.09 |
| | 0.2 |
| **Jumlah Segmen** | 2 |
| | 5 |
| | 10 |
| | 30 |
| **Kemiripan Dokumen Penyusun Dokumen** | Mirip |
| | Sangat Mirip |
| | Tidak Mirip |
| **Metode** | Genetic Algorithm |
| | Texttiling |

Setiap percobaan dilakukan terhadap beberapa *test case* yang berbeda dengan percobaan dilakukan sebanyak lebih dari satu kali. Percobaan berulang kali ini dilakukan karena pada *genetic algorithm* terdapat fungsi *random* sebagai bagian dari algoritma tersebut, sehingga untuk evaluasi diperlukan eksekusi lebih dari satu kali sehingga dapat merepresentasikan hasil yang sebenarnya. Hasil evaluasi ditentukan dengan pengambilan nilai rata-rata dari keseluruhan hasil eksekusi tersebut. Evaluasi hasil segmentasi dilakukan dengan menghitung nilai *precision* dan *recall untuk* setiap konfigurasi hasil segmentasi.

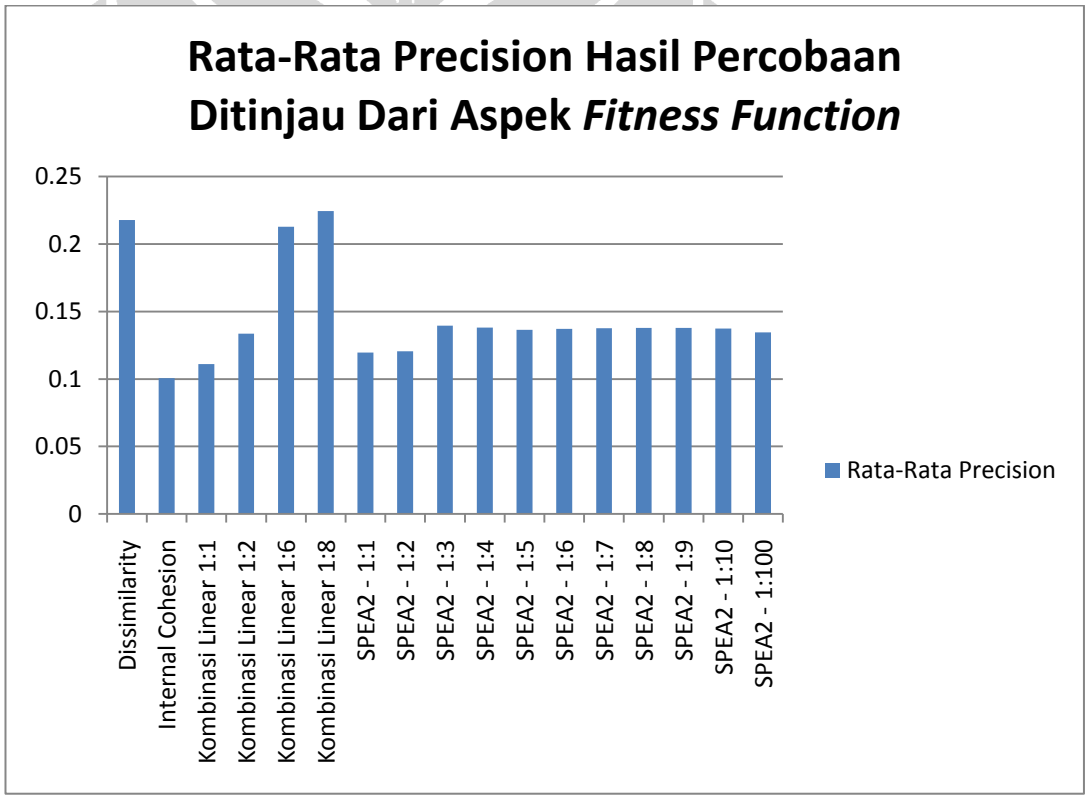
## 5.2 HASIL SEGMENTASI DOKUMEN DITINJAU DARI ASPEK *FITNESS FUNCTION*

Percobaan ini bertujuan untuk membandingkan beberapa jenis *fitness function* yang dapat digunakan pada metode segmentasi dokumen ini. *Fitness function* yang akan dibandingkan diantaranya adalah optimisasi *dissimilarity*, optimisasi *internal cohesion*, optimisasi dengan kombinasi linear antara *internal cohesion* dan *Dissimilarity* dengan perbandingan tertentu (1:1, 1:2, 1:6, dan 1:8), dan SPEA 2 dengan proses aggregasi pada populasi terakhir (perbandingan aggregasi 1:1, 1:2, 1:3, 1:4, 1:5, 1:6, 1:7, 1:8, 1:9, 1:10, dan 1:100). Pada percobaan ini dokumen yang digunakan adalah dokumen dengan 2 segmen (3 dokumen penyusun) dan domain dokumen penyusun adalah tidak mirip (campuran antara artikel media massa dan abstrak tulisan ilmiah). Percobaan ini dilakukan dengan menggunakan 5 buah *test case* berbeda, dan untuk masing-masing *test case* percobaan dilakukan sebanyak 5 kali. Data yang disajikan adalah data rata-rata *precision* dan *recall* hasil segmentasi dari keseluruhan percobaan. Tabel hasil segmentasi terhadap dokumen-dokumen tersebut dengan menggunakan berbagai *fitness function* dapat dilihat pada Tabel 5.2 sebagai berikut:

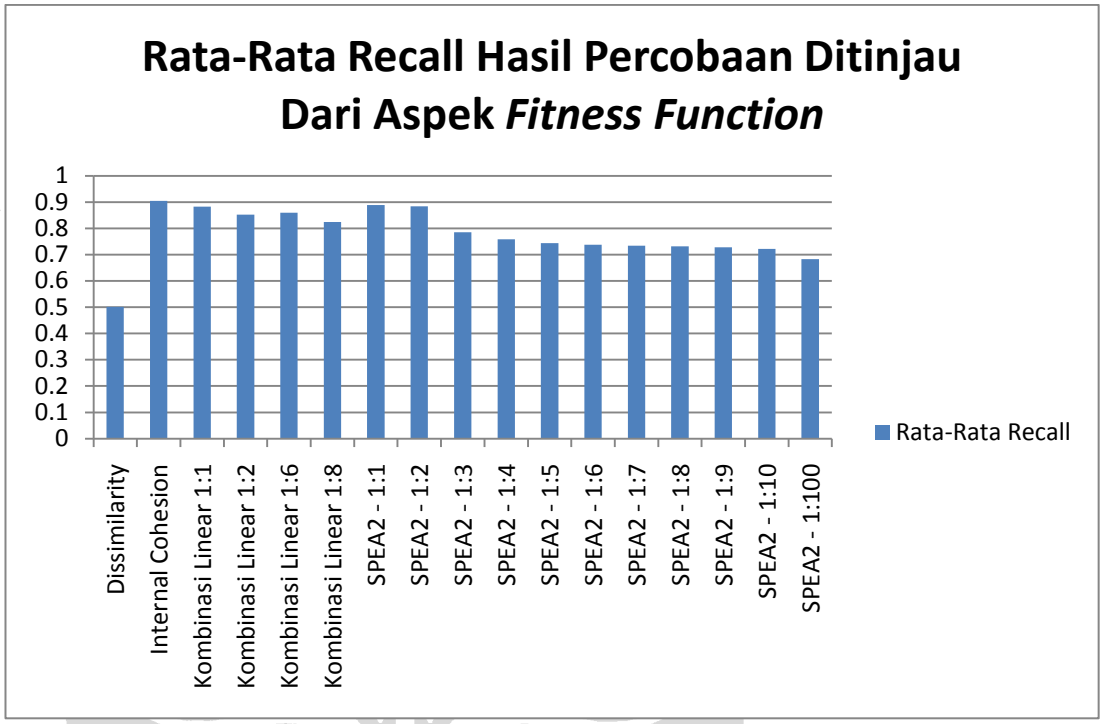
**Tabel 5.2 Tabel nilai rata-rata *precision* dan *recall* hasil percobaan ditinjau dari aspek *fitness function***

| *Fitness function* | Rata-Rata Precision | Rata-Rata Recall |
|---|---|---|
| Dissimilarity | 0.218 | 0.501 |
| Internal Cohesion | 0.100 | 0.904 |
| Kombinasi Linear 1:1 | 0.111 | 0.883 |
| Kombinasi Linear 1:2 | 0.134 | 0.852 |
| Kombinasi Linear 1:6 | 0.213 | 0.859 |
| Kombinasi Linear 1:8 | 0.224 | 0.824 |
| SPEA2 - 1:1 | 0.120 | 0.888 |
| SPEA2 - 1:2 | 0.121 | 0.884 |

| | | |
|---|---|---|
| SPEA2 - 1:3 | 0.140 | 0.786 |
| SPEA2 - 1:4 | 0.138 | 0.759 |
| SPEA2 - 1:5 | 0.136 | 0.744 |
| SPEA2 - 1:6 | 0.137 | 0.738 |
| SPEA2 - 1:7 | 0.138 | 0.734 |
| SPEA2 - 1:8 | 0.138 | 0.731 |
| SPEA2 - 1:9 | 0.138 | 0.728 |
| SPEA2 - 1:10 | 0.137 | 0.722 |
| SPEA2 - 1:100 | 0.134 | 0.683 |




**Gambar 5.1 Grafik nilai rata-rata *precision* hasil percobaan ditinjau dari aspek *fitness function***




**Gambar 5.2 Grafik nilai rata-rata *recall* hasil percobaan ditinjau dari aspek *fitness function***

Dari data yang disajikan pada Tabel 5.2, Gambar 5.1, dan Gambar 5.2, dapat dilihat bahwa nilai rata-rata *precision* tertinggi diperoleh dengan menggunakan pendekatan Kombinasi Linear 1:8, yaitu dengan nilai 0.224, sedangkan nilai rata-rata *recall* tertinggi diperoleh dengan menggunakan pendekatan optimisasi *internal cohesion* dengan nilai 0.904. Untuk *fitness function* optimisasi *internal cohesion*, meskipun memiliki nilai *recall* yang tinggi namun nilai *precision*-nya cukup rendah yaitu 0.100. Hal tersebut mungkin disebabkan oleh karakteristik optimisasi *internal cohesion* yang cenderung menempatkan banyak segmen pada dokumen sehingga nilai *recall* akan tinggi namun nilai *precision* akan semakin rendah, oleh karena itu pendekatan kombinasi linear 1:8 dapat pilih sebagai pendekatan yang lebih baik dilihat dari aspek ini.

Berdasarkan teori, SPEA 2 merupakan pendekatan yang lebih baik dibandingkan dengan pendekatan kombinasi linear, hal ini disebabkan karena pada pendekatan SPEA 2 karakteristik setiap individu dipertimbangkan untuk menentukan individu yang lebih baik. Berdasarkan hal tersebut maka pendekatan SPEA 2 dengan perbandingan yang seimbang dengan kombinasi linear akan kembali diujikan dengan *test case* yang berbeda. Pada percobaan ini konfigurasi

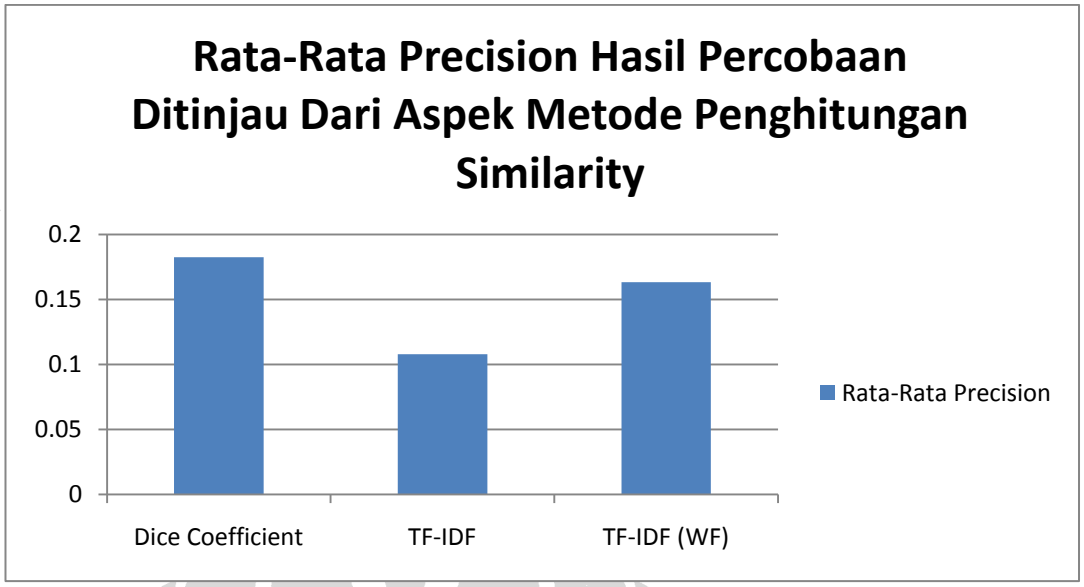
*genetic operator* yang digunakan masih menggunakan konfigurasi *default* yang diberikan oleh ECJ 18, pemilihan konfigurasi *default* ini dilakukan karena pada awalnya penulis tidak memiliki dasar pertimbangan untuk menentukan nilai-nilai *genetic operator* yang akan digunakan. Oleh karena itu perlu diujicobakan kembali kedua metode diatas dengan *testcase* yang berbeda dan *genetic operator* yang seragam untuk menentukan hasil yang lebih representatif . Hasil ujicoba kembali 2 pendekatan ini akan dibahas pada subbab 5.4.

## 5.3 HASIL SEGMENTASI DOKUMEN DITINJAU DARI ASPEK METODE PENGHITUNGAN *SIMILARITY*

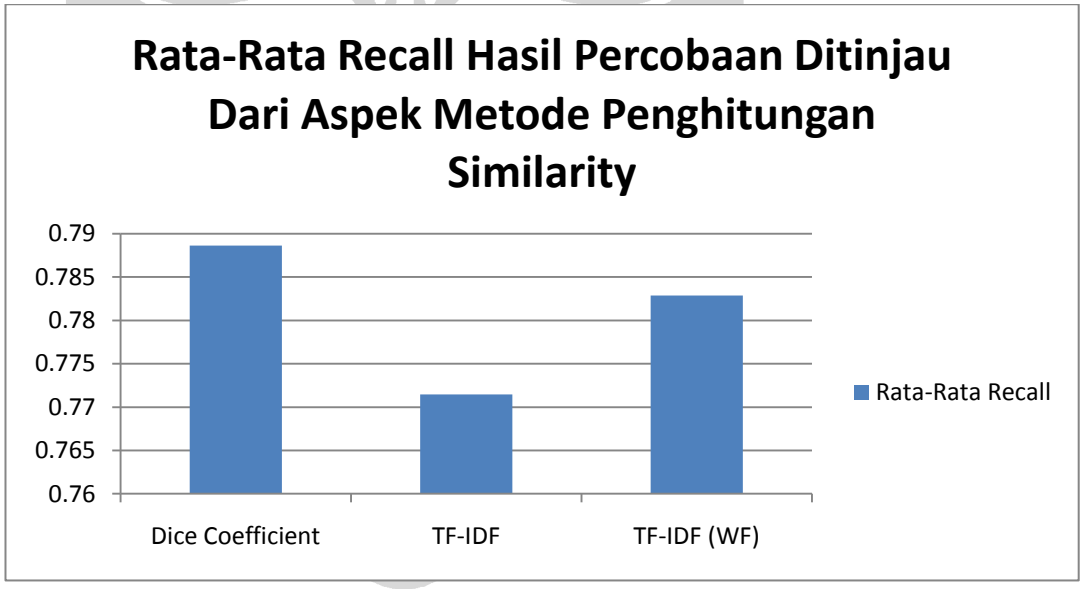
Percobaan ini bertujuan untuk membandingkan beberapa jenis metode penghitungan *similarity* yang dapat digunakan pada metode segmentasi dokumen ini. Beberapa metode penghitungan *similarity* yang akan dibandingkan adalah *Dice Coefficient*, *Cosine Similarity* dengan pembobotan TF IDF, dan *Cosine Similarity* dengan pembobotan TF IDF (*Word Frequency*). Pada percobaan ini dokumen yang digunakan adalah dokumen dengan 2 segmen (3 dokumen penyusun) dan domain dokumen penyusun adalah tidak mirip (campuran antara artikel media massa dan abstrak tulisan ilmiah). Percobaan ini dilakukan dengan menggunakan 5 buah *test case* berbeda, dan untuk masing-masing *test case* percobaan dilakukan sebanyak 5 kali. Data yang disajikan adalah data rata-rata *precision* dan *recall* hasil segmentasi dari keseluruhan percobaan. Tabel hasil segmentasi dengan menggunakan beberapa metode penghitungan nilai *similarity* yang berbeda dapat dilihat pada Tabel 5.3 sebagai berikut:

**Tabel 5.3 Tabel nilai rata-rata *precision* dan *recall* hasil percobaan ditinjau dari aspek metode penghitungan *similarity***

| Metode Penghitungan *Similarity* | Rata-Rata Precision | Rata-Rata Recall |
|---|---|---|
| Dice Coefficient | 0.182 | 0.789 |
| Cosine Similarity TF-IDF | 0.108 | 0.771 |
| Cosine Similarity TF-IDF (Word Freq) | 0.163 | 0.783 |




**Gambar 5.3 Grafik nilai rata-rata *precision* hasil percobaan ditinjau dari aspek metode penghitungan *similarity***




**Gambar 5.4 Grafik nilai rata-rata *recall* hasil percobaan ditinjau dari aspek metode penghitungan *similarity***

Dari data yang disajikan pada Tabel 5.3, Gambar 5.3, dan Gambar 5.4, dapat dilihat bahwa nilai *precision* dan *recall* tertinggi diperoleh dengan menggunakan metode *dice coefficient*, yaitu dengan nilai 0.182 untuk nilai rata-rata *precision* dan 0.789 untuk nilai rata-rata *recall*. Berdasarkan hasil yang diperoleh maka dapat disimpulkan bahwa metode penghitungan *similarity* yang memberikan hasil terbaik adalah metode penghitungan *similarity* dengan metode *dice coefficient*.

Hal ini dirasa berkaitan dengan proses penghitungan nilai *similarity* dengan metode *dice coefficient* yang benar-benar hanya memperhitungkan kemunculan kata—kata yang sama pada kedua kalimat. Dengan proses tersebut maka tingkat kemiripan antar kalimat dapat tergambarkan dengan jelas. Hal ini berbeda dengan hasil yang dilaporkan pada tesis Edison Pardengganan Siahaan (Siahaan, 2008), dimana pada penelitian tersebut diperoleh kesimpulan bahwa metode penghitungan *similarity* yang memberikan hasil segmentasi terbaik adalah TF-IDF (*Word Frequency*).

## 5.4 HASIL SEGMENTASI DOKUMEN DITINJAU DARI ASPEK UKURAN POPULASI DAN JUMLAH ITERASI YANG DIGUNAKAN

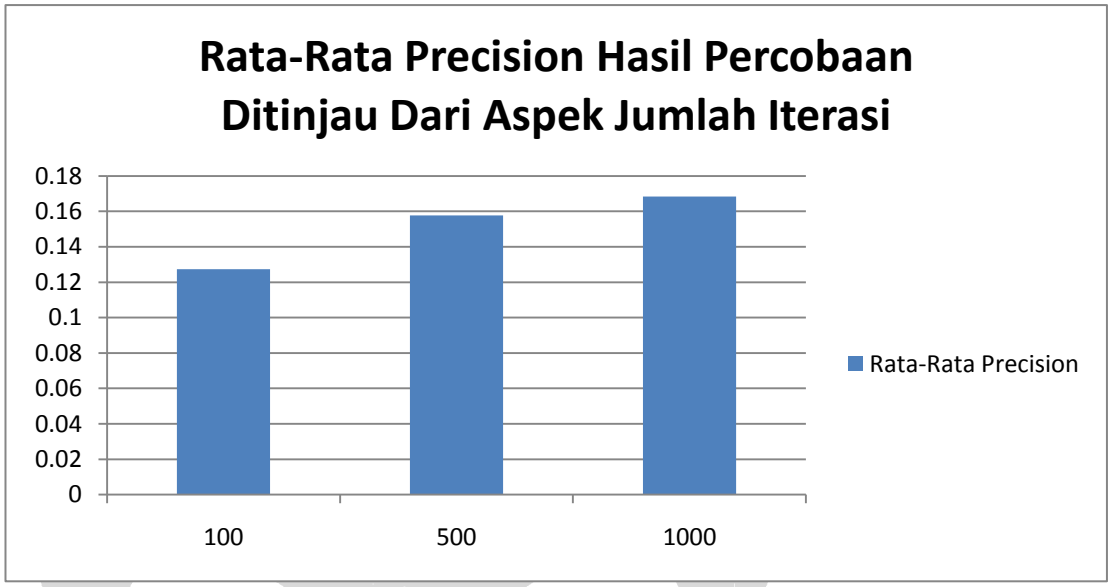
Percobaan ini bertujuan untuk membandingkan beberapa variasi pada jumlah iterasi dan ukuran populasi agar dapat diketahui jumlah iterasi dan ukuran populasi yang dapat memberikan hasil segmentasi terbaik. Pada percobaan ini dokumen yang digunakan adalah dokumen dengan 2 segmen (3 dokumen penyusun) dan domain dokumen penyusun adalah tidak mirip (campuran antara artikel media massa dan abstrak tulisan ilmiah). Percobaan ini dilakukan dengan menggunakan 5 buah *test case* berbeda, dan untuk masing-masing *test case* percobaan dilakukan sebanyak 5 kali. Data yang disajikan adalah data rata-rata *precision* dan *recall* hasil segmentasi dari keseluruhan percobaan. Tabel hasil segmentasi terhadap dokumen-dokumen tersebut dengan variasi terhadap jumlah iterasi dan ukuran populasi dapat dilihat pada Tabel 5.4 sebagai berikut:

**Tabel 5.4 Tabel nilai rata-rata *precision* dan *recall* hasil percobaan ditinjau dari aspek jumlah iterasi**

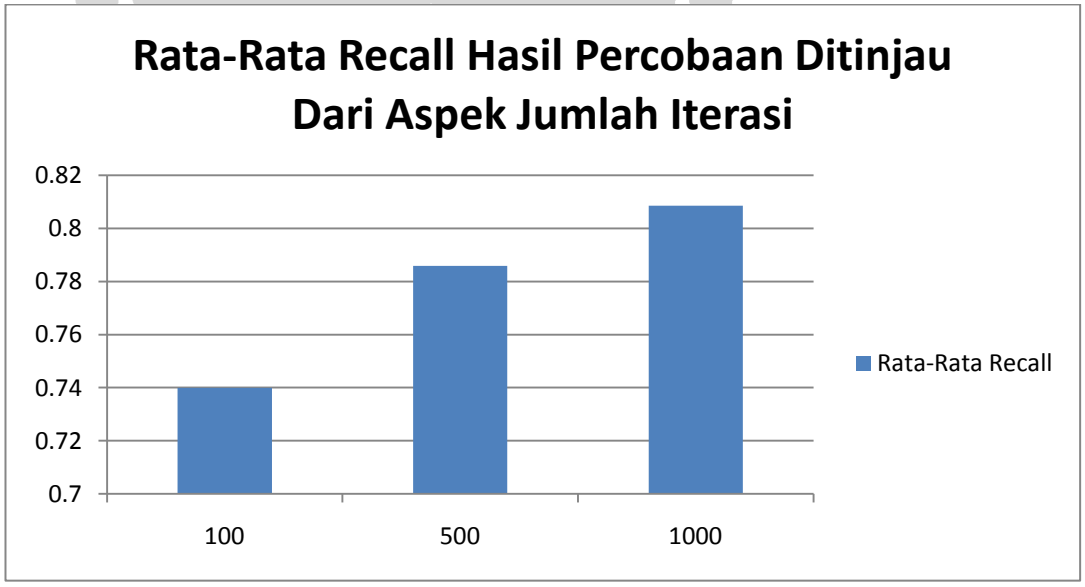
| Jumlah iterasi | Rata-Rata Precision | Rata-Rata Recall |
|---|---|---|
| 100 | 0.127 | 0.740 |
| 500 | 0.158 | 0.786 |
| 1000 | 0.168 | 0.808 |

**Tabel 5.5 Tabel nilai rata-rata *precision* dan *recall* hasil percobaan ditinjau dari aspek ukuran populasi**

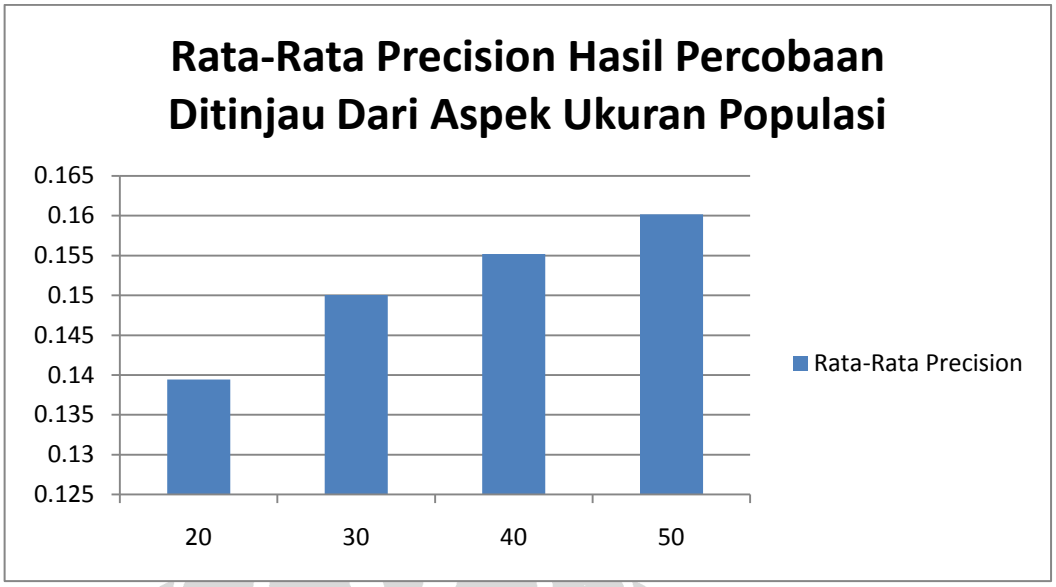
| Ukuran populasi | Rata-Rata Precision | Rata-Rata Recall |
|---|---|---|
| 20 | 0.139 | 0.764 |
| 30 | 0.150 | 0.775 |
| 40 | 0.155 | 0.792 |
| 50 | 0.160 | 0.793 |



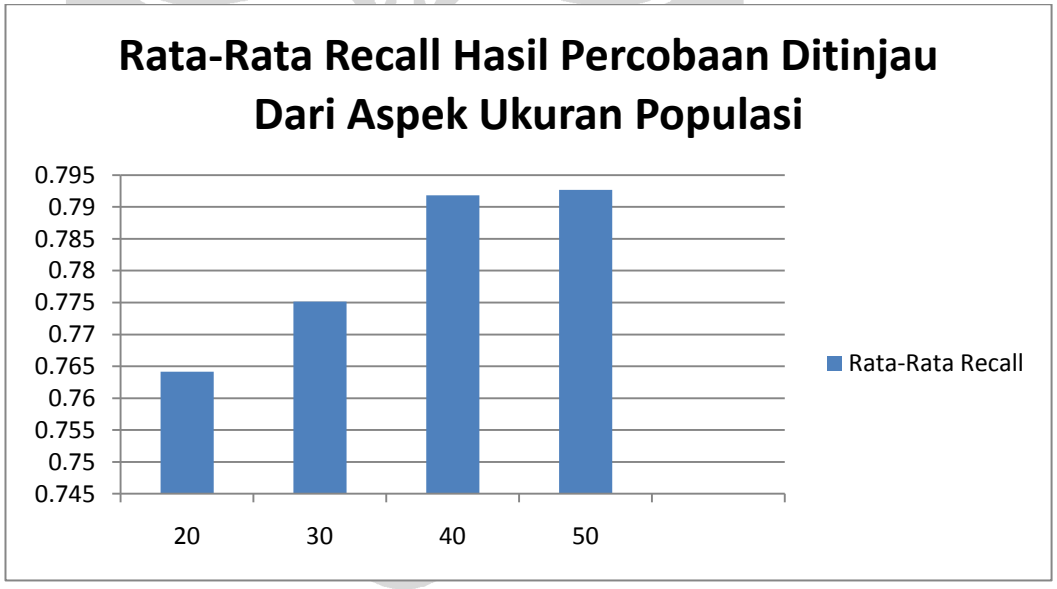

**Gambar 5.5 Grafik nilai rata-rata *precision* hasil percobaan ditinjau dari aspek jumlah iterasi**




**Gambar 5.6 Grafik nilai rata-rata *recall* hasil percobaan ditinjau dari aspek jumlah evlusi**




**Gambar 5.7 Grafik nilai rata-rata *precision* hasil percobaan ditinjau dari aspek ukuran populasi**




**Gambar 5.8 Grafik nilai rata-rata *recall* hasil percobaan ditinjau dari aspek ukuran populasi**

Dari data yang disajikan pada Tabel 5.4, Gambar 5.5, dan Gambar 5.6, dapat dilihat bahwa nilai *precision* dan *recall* tertinggi diperoleh dengan menggunakan jumlah iterasi 1000, yaitu dengan nilai 0.168 untuk nilai rata-rata *precision* dan 0.808 untuk nilai rata-rata *recall*, sedangkan dari data yang disajikan pada Tabel 5.8, Tabel 5.9, gambar 5.7, dan gambar 5.8, dapat dilihat bahwa nilai *precision* dan *recall* tertinggi diperoleh dengan menggunakan ukuran populasi 50, yaitu

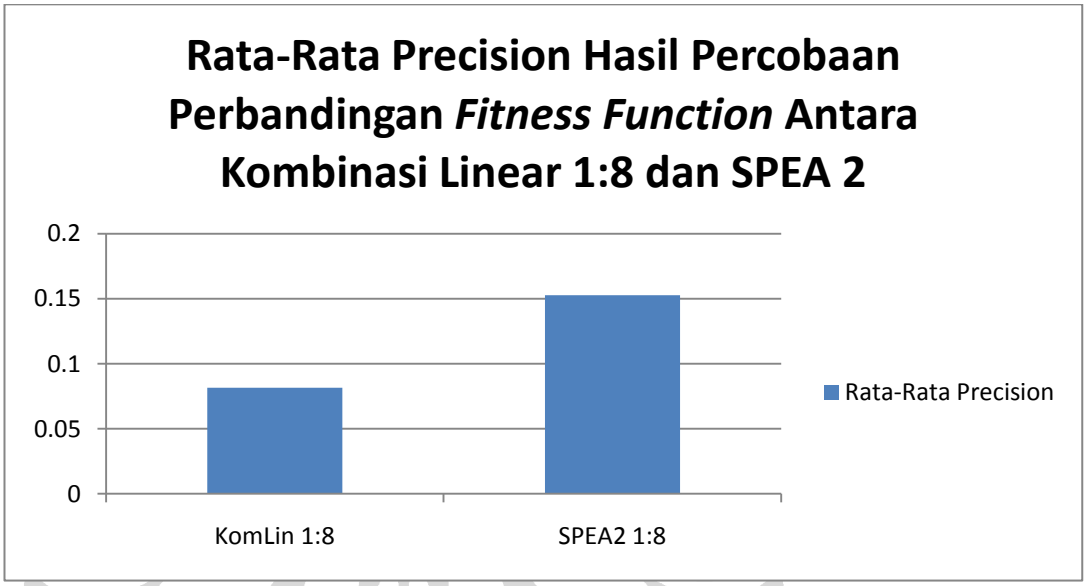
dengan nilai 0.160 untuk nilai rata-rata *precision* dan 0.793 untuk nilai rata-rata *recall*. Berdasarkan hasil yang diperoleh maka dapat disimpulkan bahwa jumlah iterasi dan ukuran populasi yang memberikan hasil segmentasi terbaik adalah jumlah iterasi 1000 dan ukuran populasi 50. Hal ini berkaitan dengan semakin banyak ukuran populasi dan semakin banyak jumlah iterasi, maka *search space* dari algoritma ini akan bertambah sehingga keragaman akan benar-benar terjamin.

## 5.5 HASIL SEGMENTASI DOKUMEN UNTUK MEMBANDINGKAN *FITNESS FUNCTION* KOMBINASI LINEAR 1:8 DAN SPEA 2 DENGAN AGGREGASI 1:8

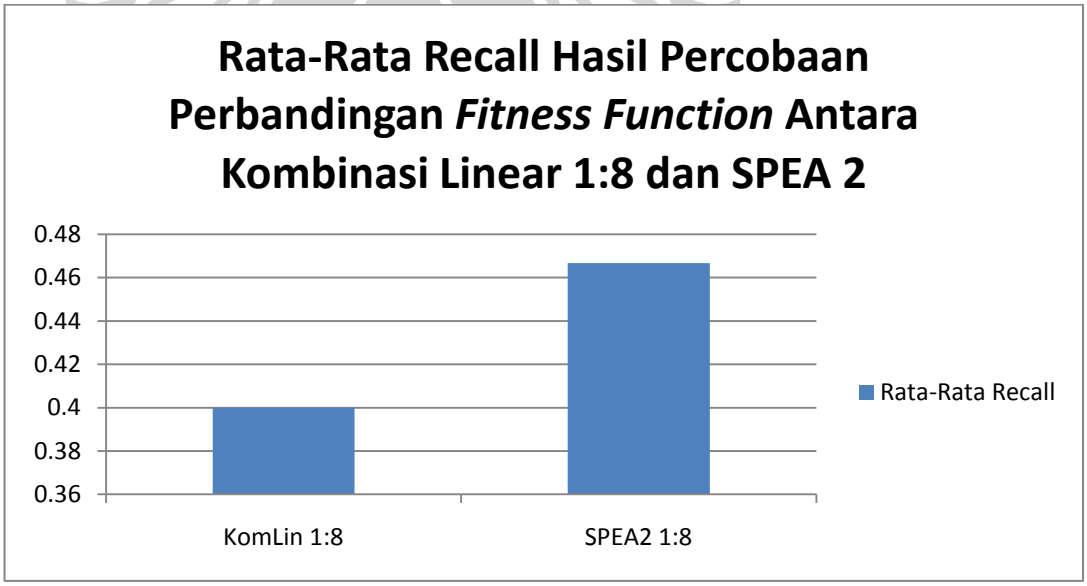
Percobaan ini bertujuan untuk membandingkan *fitness function* kombinasi linear 1:8 dan SPEA 2 dengan aggregasi 1:8 seperti telah dijelaskan sebelumnya pada subbab 5.2. Pada percobaan ini dokumen yang digunakan adalah dokumen dengan 5 segmen (6 dokumen penyusun) dan dokumen dengan 10 segmen (11 dokumen penyusun). Domain dokumen penyusun adalah tidak mirip (campuran antara artikel media massa dan abstrak tulisan ilmiah). Metode penghitungan *similarity* yang digunakan adalah *dice coefficient. Genetic operator* yang digunakan pada percobaan ini adalah *single point crossover* untuk tipe crossover, dan 0.05 untuk probabilitas mutasi, ukuran populasi sebanyak 50 individu, dan jumlah iterasi sebanyak 1000 iterasi. Percobaan ini dilakukan dengan menggunakan total 4 buah *test case* berbeda (2 *test case* untuk dokumen dengan 5 segmen dan 2 *test case* untuk dokumen dengan 10 segmen), dan untuk masing-masing *test case* percobaan dilakukan sebanyak 3 kali. Data yang disajikan adalah data rata-rata *precision* dan *recall* hasil segmentasi dari keseluruhan percobaan. Tabel hasil segmentasi terhadap dokumen-dokumen tersebut dengan menggunakan *fitness function* kombinasi linear 1:8 dan SPEA 2 dengan aggregasi 1:8 dapat dilihat pada Tabel 5.6 sebagai berikut:

**Tabel 5.6 Tabel nilai rata-rata *precision* dan *recall* hasil percobaan perbandingan *fitness function* antara kombinasi linear 1:8 dan SPEA 2**

| *Fitness Function* | Rata-Rata Precision | Rata-Rata Recall |
|---|---|---|
| Kombinasi Linear 1:8 | 0.081 | 0.400 |
| SPEA2 1:8 | 0.153 | 0.467 |




**Gambar 5.9 Grafik nilai rata-rata *precision* hasil percobaan perbandingan *fitness function* antara kombinasi linear 1:8 dan SPEA 2**




**Gambar 5.10 Grafik nilai rata-rata *precision* hasil percobaan perbandingan *fitness function* antara kombinasi linear 1:8 dan SPEA 2**

Dari data yang disajikan pada Tabel 5.6, Gambar 5.9, dan Gambar 5.10, dapat dilihat bahwa nilai *precision* dan *recall* tertinggi diperoleh dengan menggunakan pendekatan SPEA 2 dengan aggregasi 1:8, yaitu dengan nilai 0.153 untuk nilai rata-rata *precision* dan 0.467 untuk nilai rata-rata *recall.* Berdasarkan hasil yang diperoleh maka dapat disimpulkan bahwa *fitness function* yang memberikan hasil terbaik adalah *fitness function* SPEA 2 dengan aggregasi 1:8. Hasil yang diperoleh pada percobaan ini berbeda dengan hasil percobaan yang diperoleh pada subbab 5.2 dimana pada percobaan tersebut *fitness function* kombinasi linear 1:8 memberikan hasil segmentasi yang lebih baik. Hal ini disebabkan pada percobaan ini pemilihan *genetic operator* yang digunakan diseragamkan pada kedua metode tersebut untuk memberikan hasil yang lebih representatif, dimana pada percobaan sebelumnya *genetic operator* yang digunakan masih menggunakan konfigurasi *default.*

## 5.6 HASIL SEGMENTASI DOKUMEN DITINJAU DARI ASPEK *GENETIC OPERATOR* YANG DIGUNAKAN

Percobaan ini bertujuan untuk membandingkan beberapa variasi pada *genetic operator* yang digunakan (tipe *crossover* dan probabilitas mutasi) agar dapat diketahui nilai *genetic operator* yang dapat memberikan hasil segmentasi yang terbaik. Konfigurasi percobaan yang digunakan pada percobaan ini adalah *genetic algorithm* dengan *fitness function* SPEA 2 menggunakan aggregasi 1:8, metode penghitungan *similarity* dengan *dice coefficient*, ukuran populasi sebanyak 50 individu, jumlah iterasi sebanyak 1000 iterasi. Pada percobaan aspek probabilitas mutasi, tipe *crossover* yang digunakan adalah *single point crossover*, sedangkan pada percobaan aspek crossover, probabilitas mutasi yang digunakan adalah 0.09. Pada percobaan ini dokumen yang digunakan adalah dokumen dengan 2 segmen (3 dokumen penyusun) dan domain dokumen penyusun adalah tidak mirip (campuran antara artikel media massa dan abstrak tulisan ilmiah). Percobaan ini dilakukan dengan menggunakan 3 buah *test case* berbeda, dan untuk masing-masing *test case* percobaan dilakukan sebanyak 3 kali. Data yang disajikan adalah data rata-rata *precision* dan *recall* hasil segmentasi dari keseluruhan percobaan. Tabel hasil segmentasi terhadap dokumen-dokumen

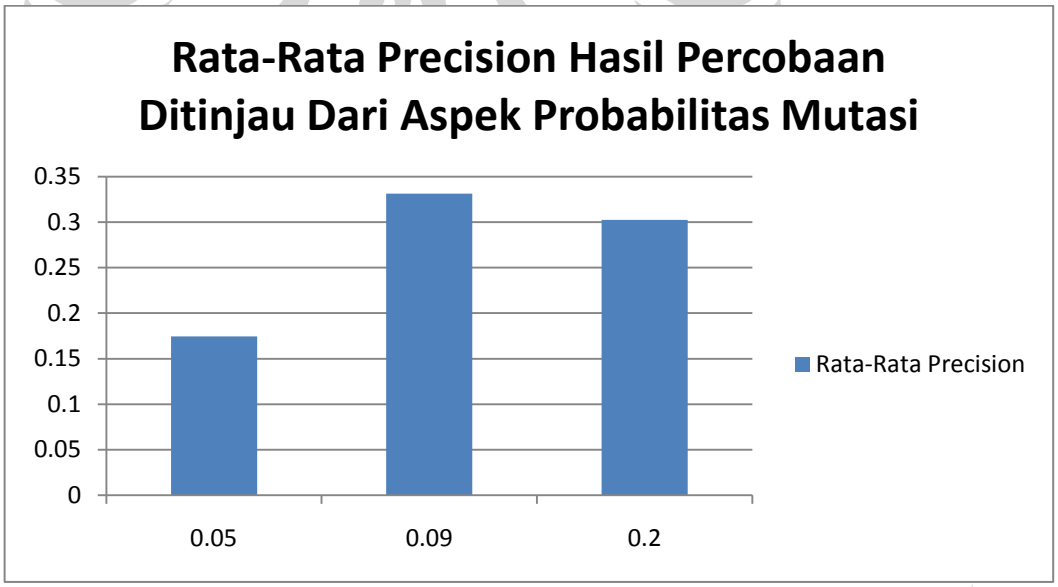
tersebut dengan variasi pada tipe *crossover* dan probabilitas mutasi dapat diliha pada Tabel 5.7, Tabel 5.8 sebagai berikut:

**Tabel 5.7 Tabel nilai rata-rata *precision* dan *recall* hasil percobaan ditinjau dari aspek probabilitas mutasi**

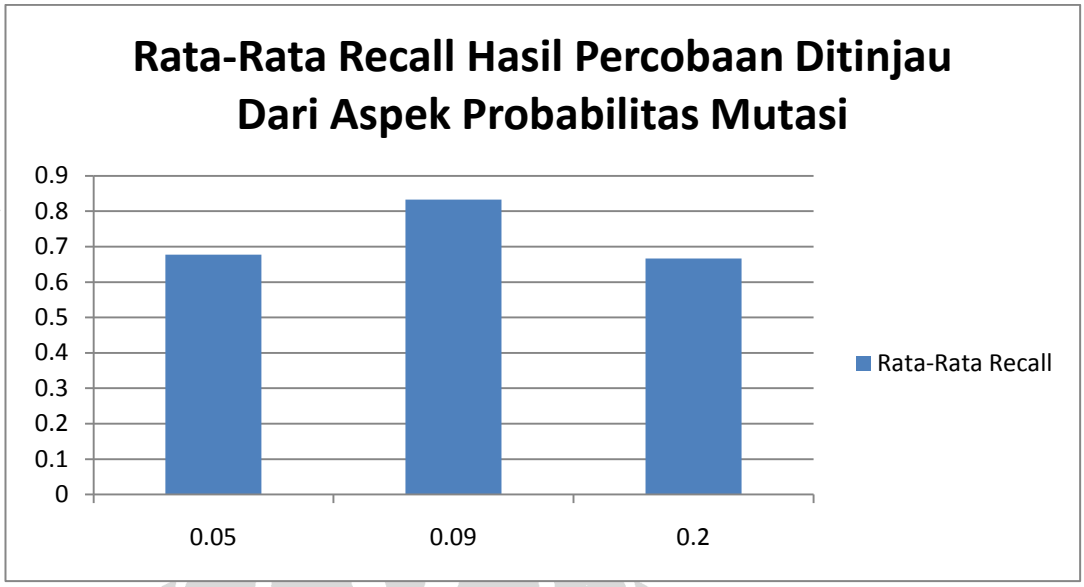
| Probabilitas Mutasi | Rata-Rata Precision | Rata-Rata Recall |
|---|---|---|
| 0.05 | 0.175 | 0.678 |
| 0.09 | 0.331 | 0.833 |
| 0.2 | 0.302 | 0.667 |

**Tabel 5.8 Tabel nilai rata-rata *precision* dan *recall* hasil percobaan ditinjau dari aspek tipe *crossover***

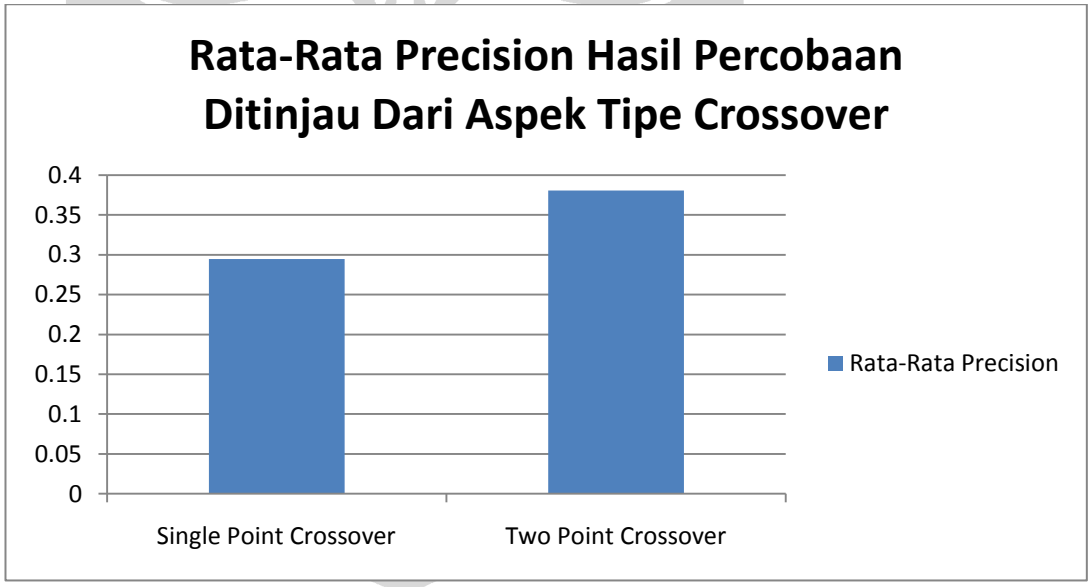
| Tipe Crossover | Rata-Rata Precision | Rata-Rata Recall |
|---|---|---|
| Single Point Crossover | 0.295 | 0.611 |
| Two Point Crossover | 0.380 | 0.833 |



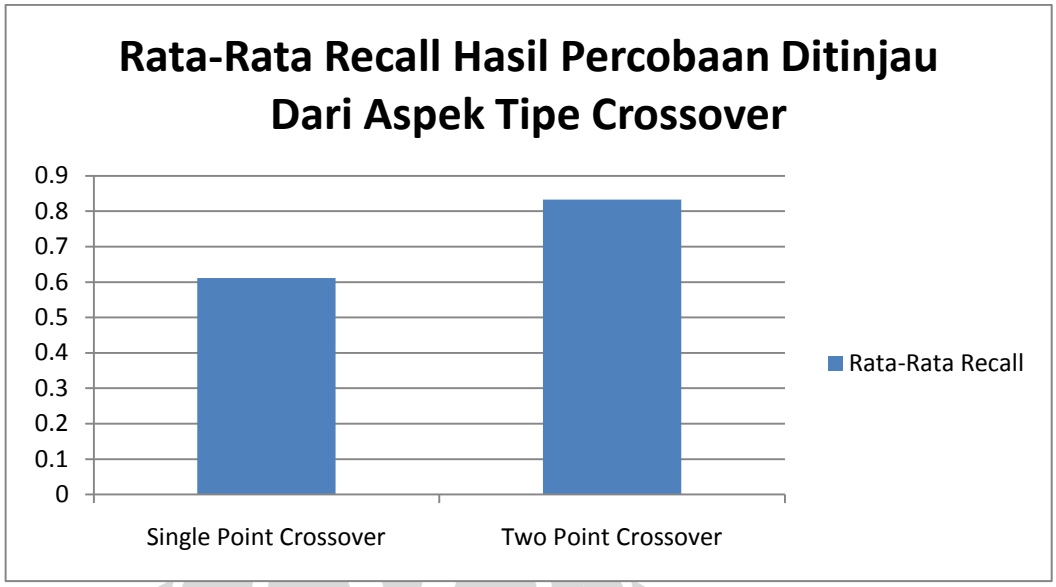

**Gambar 5.11 Grafik nilai rata-rata *precision* hasil percobaan ditinjau dari aspek probabilitas mutasi**




**Gambar 5.12 Grafik nilai rata-rata *recall* hasil percobaan ditinjau dari aspek probabilitas mutasi**




**Gambar 5.13 Grafik nilai rata-rata *precision* hasil percobaan ditinjau dari aspek tipe *crossover***




**Gambar 5.14 Grafik nilai rata-rata *recall* hasil percobaan ditinjau dari aspek tipe *crossover***

Dari data yang disajikan pada Tabel 5.7, Gambar 5.11, dan Gambar 5.12, dapat dilihat bahwa nilai *precision* dan *recall* tertinggi diperoleh dengan menggunakan probabilitas mutasi 0.09, yaitu dengan nilai 0.331 untuk nilai rata-rata *precision* dan 0.833 untuk nilai rata-rata *recall* dari data yang disajikan pada Tabel 5.8, Gambar 5.13, dan Gambar 5.14, dapat dilihat bahwa nilai *precision* dan *recall* tertinggi diperoleh dengan menggunakan tipe *crossover two point crossover*, yaitu dengan nilai 0.380 untuk nilai rata-rata *precision* dan 0.833 untuk nilai rata-rata *recall*. Berdasarkan hasil yang diperoleh maka dapat disimpulkan bahwa tipe *crossover* dan probabilitas mutasi yang memberikan hasil segmentasi terbaik adalah tipe *crossover two point crossover* dan probabilitas mutasi 0.09.

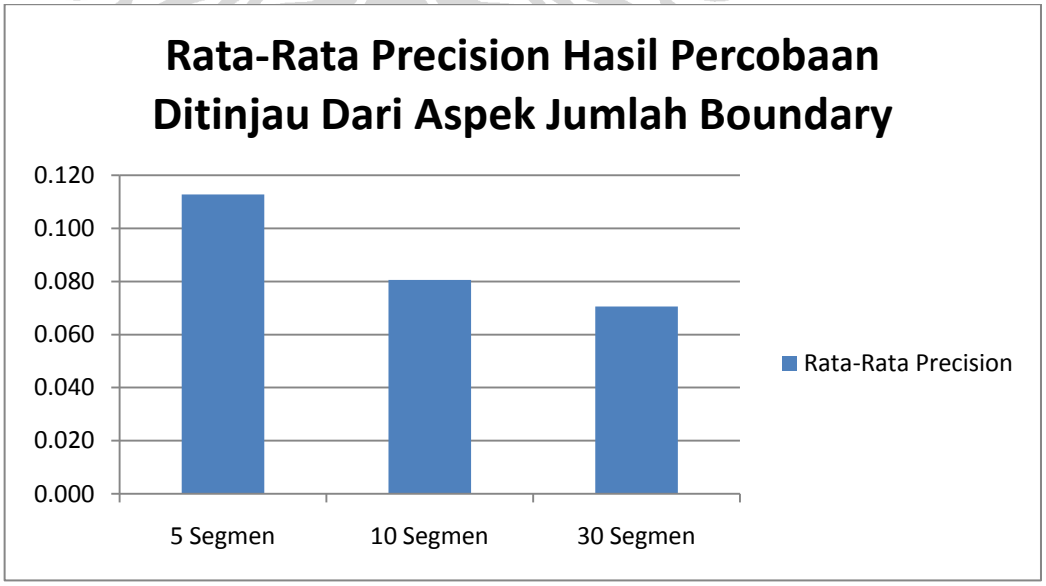
## 5.7 HASIL SEGMENTASI DOKUMEN DITINJAU DARI ASPEK BANYAKNYA JUMLAH SEGMEN

Percobaan ini bertujuan untuk melihat hasil segmentasi dengan *genetic algorithm* menggunakan konfigurasi percobaan terbaik yang telah diuji cobakan pada percobaan-percobaan sebelumnya terhadap jenis dokumen yang berbeda. Konfigurasi percobaan yang digunakan pada percobaan ini adalah *genetic algorithm* dengan *fitness function* SPEA 2 menggunakan aggregasi 1:8, metode penghitungan *similarity* dengan *dice coefficient*, ukuran populasi sebanyak 50 individu, jumlah iterasi sebanyak 1000 iterasi, probabilias mutasi 0.09, dan tipe

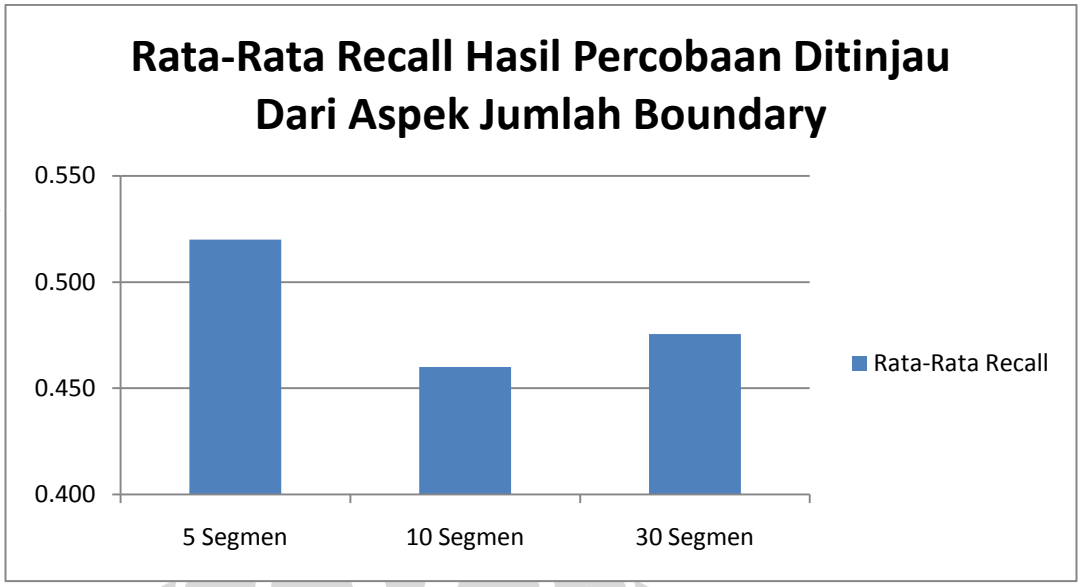
*crossover* menggunakan *two point crossover*. Pada percobaan ini proses segmentasi akan diujikan terhadap dokumen dengan jumlah segmen beragam. Dokumen yang digunakan adalah dokumen dengan 5 segmen (6 dokumen penyusun), 10 segmen (11 dokumen penyusun), dan 30 segmen (31 dokumen penyusun) dengan domain dokumen penyusun adalah tidak mirip (campuran antara artikel media massa dan abstrak tulisan ilmiah). Percobaan ini dilakukan dengan menggunakan 5 buah *test case* berbeda, dan untuk masing-masing *test case* percobaan dilakukan sebanyak 3 kali. Data yang disajikan adalah data rata-rata *precision* dan *recall* hasil segmentasi dari keseluruhan percobaan. Tabel hasil segmentasi terhadap dokumen-dokumen tersebut dari aspek banyaknya jumlah segmen dapat dilihat pada Tabel 5.9 sebagai berikut:

**Tabel 5.9 Tabel nilai rata-rata *precision* dan *recall* hasil percobaan ditinjau dari aspek banyaknya jumlah segmen**

| Jumlah Segmen | Rata-Rata Precision | Rata-Rata Recall |
|---|---|---|
| 5 Segmen | 0.113 | 0.520 |
| 10 Segmen | 0.081 | 0.460 |
| 30 Segmen | 0.071 | 0.476 |




**Gambar 5.15 Grafik nilai rata-rata *precision* hasil percobaan ditinjau dari aspek banyaknya jumlah segmen**




**Gambar 5.16 Grafik nilai rata-rata *recall* hasil percobaan ditinjau dari aspek banyaknya jumlah segmen**

Dari data yang disajikan pada Tabel 5.9, dan Gambar 5.15 dapat dilihat bahwa semakin banyak jumlah segmen pada suatu dokumen, maka precision yang diperoleh akan semakin menurun, hal ini disebabkan karena jumlah segmen yang ditemukan oleh metode ini semakin meningkat sedangkan jumlah segmen benar yang ditemukan cenderung stabil. Dari data yang disajikan pada Tabel 5.9, dan Gambar 5.16 dapat dilihat bahwa nilai *recall* tertinggi diperoleh pada dokumen dengan 5 segmen, sedangkan untuk dokumen dengan 10 dan 30 segmen rata-rata *recall* yang diperoleh cenderung berimbang (perbedaan rata-rata *recall* lebih kurang 0.015), meskipun berdasarkan data percobaan rata-rata *recall* untuk dokumen dengan 30 segmen lebih tinggi dibandingkan dengan rata-rata *recall* untuk dokumen dengan 10 segmen. Hal ini dirasa berkaitan dengan meningkatnya tingkat kesulitan bagi proses segmentasi untuk menentukan letak segmen sebenarnya seiring dengan bertambahnya jumlah komposisi dokumen penyusunnya.

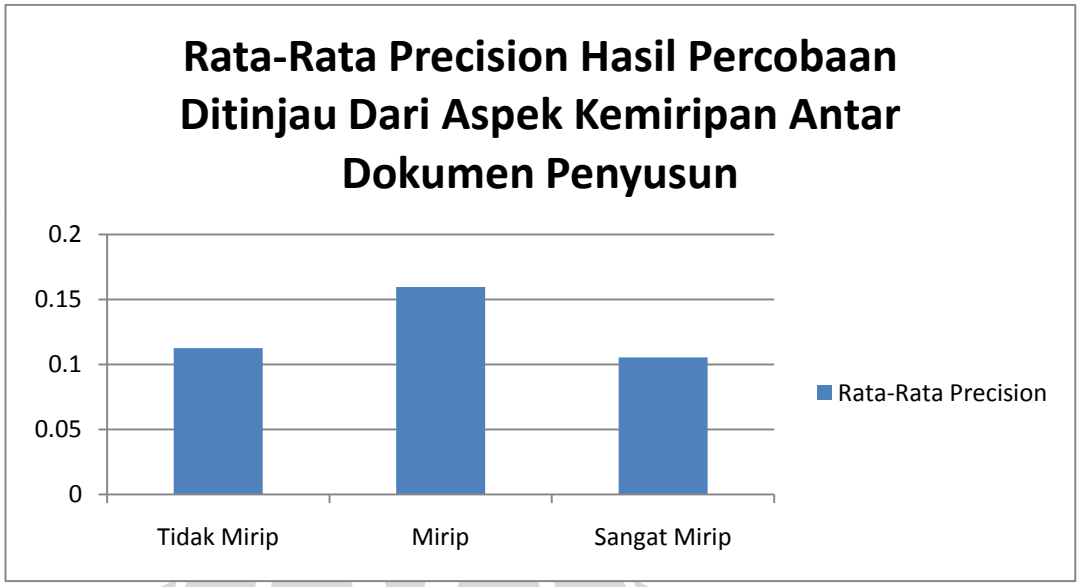
## 5.8 HASIL SEGMENTASI DOKUMEN DITINJAU DARI ASPEK KEMIRIPAN ANTAR DOKUMEN PENYUSUN

Percobaan ini bertujuan untuk melihat hasil segmentasi dengan *genetic algorithm* menggunakan konfigurasi percobaan terbaik yang telah diuji cobakan

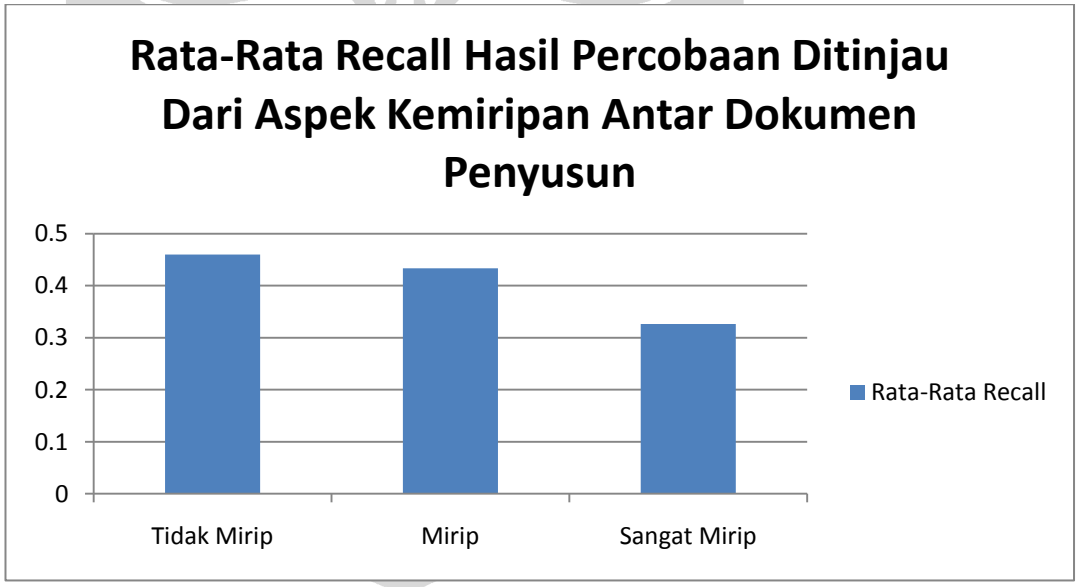
pada percobaan-percobaan sebelumnya terhadap jenis dokumen yang berbeda. Konfigurasi percobaan yang digunakan pada percobaan ini adalah *genetic algorithm* dengan *fitness function* SPEA 2 menggunakan aggregasi 1:8, metode penghitungan *similarity* dengan *dice coefficient*, ukuran populasi sebanyak 50 individu, jumlah iterasi sebanyak 1000 iterasi, probabilias mutasi 0.09, dan tipe *crossover* menggunakan *two point crossover*. Pada percobaan ini proses segmentasi akan diujikan terhadap dokumen dengan variasi tingkat kemiripan antar dokumen penyusunnya. Dokumen pada percobaan ini terdiri dari 3 jenis dokumen yaitu dokumen dengan domain dokumen penyusun tidak mirip (campur), dokumen dengan domain dokumen penyusun mirip (dokumen abstrak tulisan ilmiah), dokumen dengan domain dokumen penyusun sangat mirip (dokumen abstrak tulisan ilmiah dengan topik RPL). Jumlah segmen dokumen pada percobaan kedua tersebut adalah dokumen dengan 10 segmen (11 dokumen penyusun). Percobaan ini dilakukan dengan menggunakan 5 buah *test case* berbeda, dan untuk masing-masing *test case* percobaan dilakukan sebanyak 3 kali. Data yang disajikan adalah data rata-rata *precision* dan *recall* hasil segmentasi dari keseluruhan percobaan. Tabel dan grafik hasil segmentasi terhadap dokumen-dokumen tersebut dari aspek kemiripan antar dokumen penyusun dapat dilihat pada Tabel 5.10 sebagai berikut:

**Tabel 5.10 Tabel nilai rata-rata *precision* dan *recall* hasil percobaan ditinjau dari aspek kemiripan antar dokumen penyusun**

| Tingkat Kemiripan | Rata-Rata Precision | Rata-Rata Recall |
|---|---|---|
| Tidak Mirip | 0.113 | 0.460 |
| Mirip | 0.160 | 0.433 |
| Sangat Mirip | 0.105 | 0.327 |




**Gambar 5.17 Grafik nilai rata-rata *precision* hasil percobaan ditinjau dari aspek kemiripan antar dokumen penyusun**




**Gambar 5.18 Grafik nilai rata-rata *recall* hasil percobaan ditinjau dari aspek kemiripan antar dokumen penyusun**

Berdasarkan data yang disajikan pada Tabel 5.10, dan Gambar 5.17 dapat dilihat bahwa nilai rata-rata precision tertinggi diperoleh pada dokumen dengan domain dokumen penyusun mirip (domain abstrak tulisan ilmiah), sedangkan berdasarkan data yang disajikan pada Tabel 5.10 dan Gambar 5.18, rata-rata *recall* tertinggi diperoleh pada dokumen dengan domain dokumen penyusun tidak mirip (gabungan artikel media massa dan abstrak tulisan ilmiah). Penurunan nilai *recall*

ini sesuai dengan dugaan bahwa dengan semakin miripnya domain dokumen penyusun maka proses segmentasi akan semakin sulit untuk menentukan letak segmen yang sebenarnya karena dengan semakin miripnya domain maka kalimat-kalimat yang digunakan pada dokumen tersebut akan semakin mirip. Domain dokumen mirip memiliki nilai *precision* yang paling tinggi dirasakan berkaitan dengan pemilihan domain dokumen ini berasal dari domain abstrak tulisan ilmiah yang memiliki komposisi kalimat lebih baku dan lebih teratur dibandingkan dengan domain dokumen artikel media massa (terutama untuk domain artikel *travel* dan properti), sehingga meminimalkan jumlah segmen yang ditemukan oleh proses segmentasi. Penggunaan kalimat-kalimat yang baku dapat menghindari adanya kalimat-kalimat yang benar-benar berbeda dibandingkan kalimat-kalimat sebelum dan sesudah kalimat tersebut. Dengan munculnya kalimat-kalimat yang benar-benar berbeda tersebut makan dapat mengurangi nilai *similarity* antar kalimat-kalimat disekitarnya (penjelasan lebih detil dapat dillihat pada subbab 5.10). Untuk domain dokumen sangat mirip memiliki nilai *precision* yang paling rendah dirasakan berkaitan dengan menurunnya jumlah segmen benar yang dapat ditemukan oleh proses segmentasi pada jenis dokumen ini, sehingga turut mempengaruhi nilai *precision* yang dihasilkan.

## 5.9 PERBANDINGAN HASIL SEGMENTASI DENGAN METODE *GENETIC ALGORITHM* DAN HASIL SEGMENTASI DENGAN METODE TEXTTILING

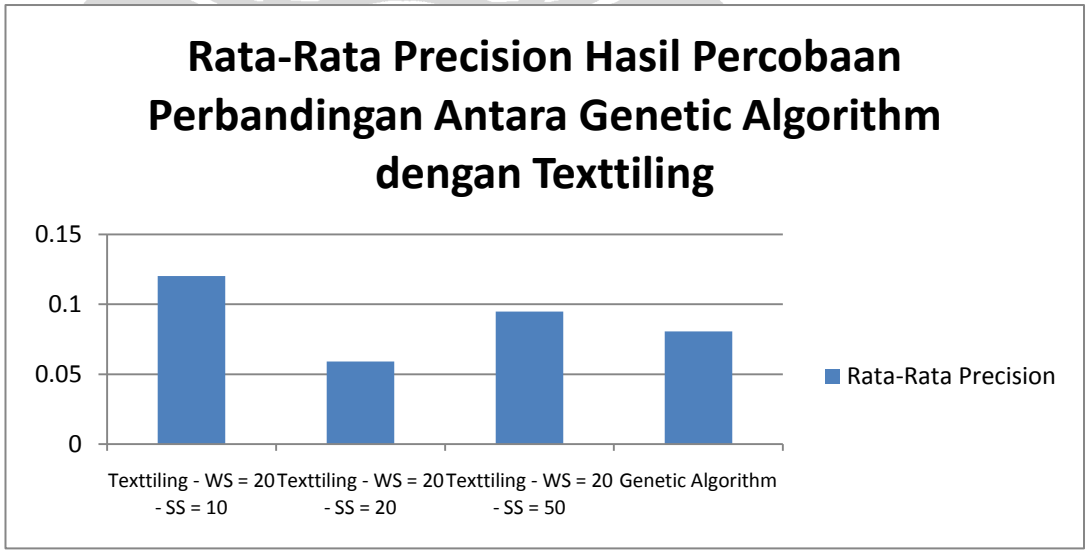
Percobaan ini bertujuan untuk melihat hasil segmentasi dengan *genetic algorithm* menggunakan konfigurasi percobaan terbaik yang telah diuji cobakan pada percobaan-percobaan sebelumnya jika dibandingkan dengan metode segmentasi dokumen lainya. Metode yang digunakan sebagai pembanding adalah metode Texttiling (Hearst, 1997) yang sejak lama merupakan salah satu metode segmentasi dokumen yang cukup baik. Percobaan segmentasi dokumen dengan menggunakan metode Texttiling ini dilakukan pada suatu *website* yang menyediakan fasilitas segmentasi dengan metode Texttiling secara *online* (http://morphadorner.northwestern.edu). *Website* ini menyediakan 2 variabel yang berkaitan dengan metode Texttiling yang dapat diubah-ubah sesuai keinginan *user*,

variabel tersebut ada *window size* dan *segment size.* Pada percobaan yang dilakukan ini *window size* yang dipilih adalah 20 sesuai dengan yang disarankan pada (Manning, 1999), sedangkan *segment size* yang dipilih adalah 10, 20, dan 50. Dokumen yang digunakan pada percobaan ini adalah dokumen dengan 10 segmen (11 dokumen penyusun) dan domain dokumen penyusun adalah tidak mirip (campuran antara artikel media massa dan abstrak tulisan ilmiah). Percobaan ini dilakukan dengan menggunakan 5 buah *test case* berbeda. Data yang disajikan adalah data rata-rata *precision* dan *recall* hasil segmentasi dari keseluruhan percobaan. Berikut adalah Tabel hasil segmentasi terhadap dokumen-dokumen tersebut dengan menggunakan metode segmentasi yang berbeda dapat dilihat pada Tabel 5.11 sebagai berikut:

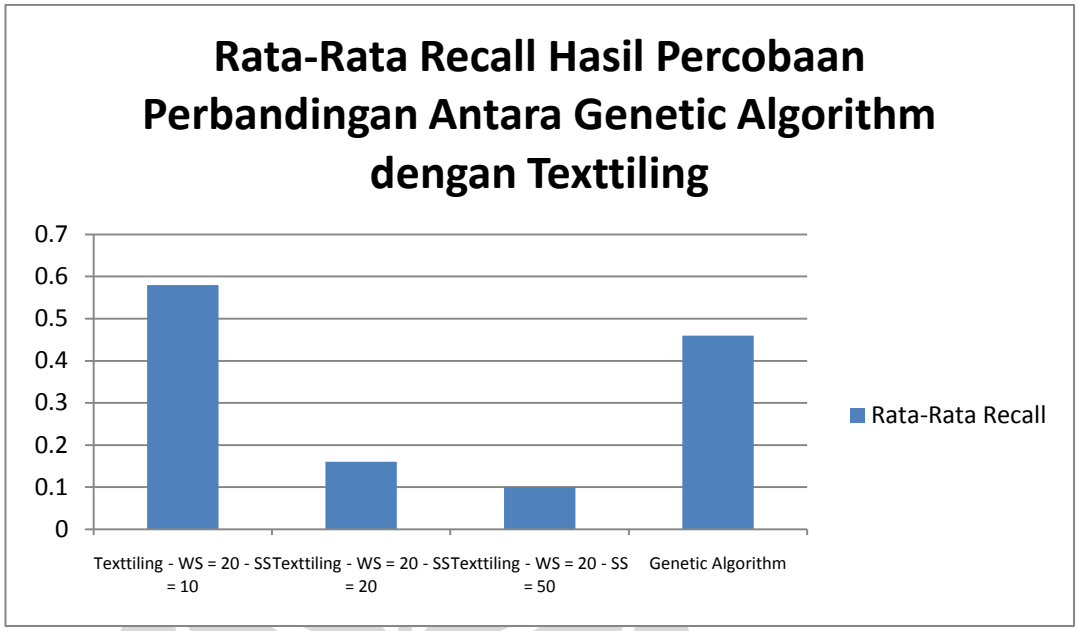
**Tabel 5.11 Tabel nilai rata-rata *precision* dan *recall* hasil percobaan perbandingan antara metode *genetic algorithm* dengan metode Textiling**

| Metode | Rata-Rata Precision | Rata-Rata Recall |
|---|---|---|
| Texttiling - WS = 20 - SS = 10 | 0.120 | 0.580 |
| Texttiling - WS = 20 - SS = 20 | 0.059 | 0.160 |
| Texttiling - WS = 20 - SS = 50 | 0.095 | 0.100 |
| Genetic Algorithm | 0.081 | 0.460 |

*WS = Window Size, SS = Segment Size




**Gambar 5.19 Grafik nilai rata-rata *precision* hasil percobaan perbandingan antara metode *genetic algorithm* dengan metode Textiling**




**Gambar 5.20 Grafik nilai rata-rata *recall* hasil percobaan perbandingan antara metode *genetic algorithm* dengan metode Textiling**

Dari data yang disajikan pada Tabel 5.11, Gambar 5.19, dan Gambar 5.20, dapat dilihat bahwa nilai *precision* dan *recall* tertinggi diperoleh dengan menggunakan pendekatan metode segmentasi Texttiling dengan *window size* 20, dan *segment size* 10, yaitu dengan nilai 0.120 untuk nilai rata-rata *precision* dan 0.580 untuk nilai rata-rata *recall*. Dapat dilihat pula dari hasil segmentasi bahwa hasil yang diberikan oleh metode Texttiling sangat bergantung kepada pemilihan *window size* dan *segment size* yang tepat. Jika pemilihan kedua parameter tersebut tepat, maka akan menghasilkan hasil segmentasi yang lebih baik dibandingkan hasil segmentasi dengan *genetic algorithm* dan sebaliknya jika pemilihan kedua parameter tersebut kurang baik makan hasil yang lebih baik akan dihasilkan oleh *genetic algorithm*. Penggunaan sistem segmentasi dokumen pada *website* http://morphadorner.northwestern.edu masih memerlukan intervensi manual dari pengguna dalam menentukan *window size* dan *segment size*, berbeda dengan *genetic algorithm* yang tidak memerlukan intervensi manual.

## 5.10 ANALISA HASIL

Setelah mengamati dan menganalisa keseluruhan hasil segmentasi yang dilakukan dengan metode *genetic algorithm* ini, diketahui bahwa rentang nilai rata-rata *precision* yang diperoleh adalah berkisar antara 0.08 sampai 0.38, dan rentang nilai rata-rata *recall* yang diperoleh adalah berkisar antara 0.4 sampai 0.9. Berdasarkan rentang nilai tersebut, nilai rata-rata *precision* yang diperoleh cukup rendah dan nilai rata-rata *recall* yang diperoleh cukup baik meskipun nilainya tidak tinggi untuk semua jenis dokumen. Kedua hal tersebut cukup memberikan gambaran bahwa proses segmentasi ini cenderung menemukan segmen cukup banyak sehingga nilai *precision* (perbandingan segmen benar dengan total segmen yang ditemukan) rendah, sedangkan untuk nilai *recall* metode segmentasi ini sudah cukup bisa menentukan letak batas antara dokumen-dokumen penyusunnya meskipun hal tersebut masih bergantung pada jumlah segmen sebenarnya yang perlu ditemukan dan domain dokumen penyusun.

Rendahnya nilai rata-rata *precision* yang diperoleh salah satunya disebabkan karena banyaknya jumlah segmen yang ditemukan oleh proses segmentasi ini. Salah satu penyebab banyaknya *boundary* yang ditemukan oleh proses segmentasi ini adalah karena penggunaan metode penghitungan *similarity* yang hanya merefleksikan tingkat kemiripan berdasarkan kata-kata yang persis sama yang muncul pada kedua kalimat. Permasalahan ini sangat sering ditemukan pada percobaan-percobaan yang telah dilakukan. Hal tersebut terkadang membuat proses segmentasi menentukan beberapa *boundary* pada satu dokumen meskipun segmen hasil segmentasi tersebut masih termasuk kedalam satu dokumen penyusun. Berikut adalah contoh cuplikan dokumen beserta hasil segmentasi dari suatu dokumen (terdiri dari 2 topik penyusun) yang tersegmen karena perbedaan kalimat-kalimat penyusun dokumen tersebut:

cermin guna solusi luas

nah letak partisi lapis cermin meskipun sempit tetap asa lega

efek pantul cermin

sempit cantik

nara safina russia 37823

rena williams united states 37174

ana ivanovic serbia 32895

**Gambar 5.21 *Input* yang digunakan**

cermin guna solusi luas

nah letak partisi lapis cermin meskipun sempit tetap asa lega

efek pantul cermin

sempit cantik

---

nara safina russia 37823

---

rena williams united states 37174

ana ivanovic serbia 32895

**Gambar 5.22 Hasil segmentasi *input* dengan metode SPEA 2 aggregasi 1:8**

cermin guna solusi luas

nah letak partisi lapis cermin meskipun sempit tetap asa lega

efek pantul cermin

sempit cantik

---

nara safina russia 37823

rena williams united states 37174

ana ivanovic serbia 32895

**Gambar 5.23 Letak segmen sebenarnya dari *input* (memisahkan antara topik properti dengan topik olahraga tenis)**

Dari contoh pada Gambar 5.21, Gambar 5.22, dan Gambar 5.23 diatas tampak bahwa penggunaan jenis kata-kata yang berbeda yang terkandung pada suatu dokumen dapat membuat proses segmentasi ini menentukan *boundary* ditempat yang tidak seharunya. Nilai *internal cohesion* dan *dissimilarity* untuk hasil segmentasi pada Gambar 5.22 diatas adalah 0.09363553 untuk *internal cohesion* dan 1.0 untuk *dissimilarity*. Jika segmen diletakan pada posisi sebenarnya (Gambar 5.23) maka *internal cohesion* yang diperoleh adalah 0.08323158, dan *dissimilarity* adalah 1.0. Karena konfigurasi segmen pada Gambar 5.22 memberikan nilai *internal cohesion* yang lebih tinggi dibandingkan dengan konfigurasi seharusnya seperti pada Gambar 5.23 (nilai *dissimilarity* sama), maka metode *genetic algoritm* menentukan konfigurasi segmentasi tersebut adalah yang terbaik.

Menurunnya nilai rata-rata *recall* seiring dengan semakin banyaknya jumlah dokumen penyusun tentunya berkaitan dengan semakin luasnya *search space* yang perlu ditelusuri oleh *genetic algorithm* dengan bertambahnya jumlah dokumen penyusun tersebut. Dengan semakin meningkatnya *search space* ini, *genetic algorithm* belum mampu untuk menemukan konfigurasi yang paling optimal karena tidak diimbangi dengan penyesuaian *genetic operator* (misalnya

penambahan jumlah iterasi atau ukuran populasi). Dengan penyesuaian *genetic operator* tersebut tentunya dapat semakin mengoptimalkan hasil yang diperoleh. Menurunnya nilai rata-rata *recall* seiring dengan semakin miripnya domain dokumen penyusun pun tentunya merupakan hal yang wajar, mengingat pada dokumen-dokumen dengan domain yang sama penggunaan kata-kata/kalimat pada dokumen-dokumen tersebut pun akan semakin mirip sehingga cukup mempersulit proses segmentasi untuk menemukan letak *boundary* yang sebenarnya.

## 5.11 RANGKUMAN HASIL

Pembahasan hasil untuk segmentasi dokumen menggunakan *genetic algorithm* telah dilakukan pada subbab 5.2 sampai subbab 5.6. Subbab ini akan memberikan rangkuman singkat dari hasil-hasil eksperimen yang telah dideskripsikan pada subbab-subbab tersebut. Beberapa hal yang dapat dirangkum adalah:

- Dilihat dari aspek *fitness function*, dibuktikan bahwa *fitness function* dengan pendekatan SPEA 2 memberikan hasil segmentasi yang paling baik dari segi *precision* dan *recall*. Hal tersebut sesuai dengan teori bahwa pendekatan SPEA 2 merupakan pendekatan yang terbaik dalam pencarian solusi *multiobjective problem*.
- Dilihat dari aspek metode penghitungan nilai *similarity*, dibuktikan bahwa metode *dice coefficient* memberikan hasil segmentasi yang paling baik dari segi *precision* dan *recall*. Hal tersebut dirasa berkaitan dengan proses penghitungan nilai *similarity* dengan metode *dice coefficient* yang benar-benar hanya memperhitungkan kemunculan kata—kata yang sama pada kedua kalimat. Dengan proses tersebut maka tingkat kemiripan antar kalimat dapat tergambarkan dengan jelas.
- Dilihat dari aspek ukuran populasi, dibuktikan bahwa ukuran populasi sebanyak 50 individu memberikan hasil segmentasi yang paling baik dari segi *precision* dan *recall*. Hasil ini sesuai dengan dugaan bahwa semakin besar ukuran populasi maka hasil yang diperoleh akan semakin baik. Hal tersebut berkaitan dengan semakin banyak ukuran populasi maka *search*

*space* dari algoritma ini akan bertambah sehingga keragaman akan benar-benar terjamin.

- Dilihat dari aspek jumlah iterasi, dibuktikan bahwa jumlah iterasi sebanyak 1000 evolusi memberikan hasil segmentasi yang paling baik dari segi *precision* dan *recall*. Hasil ini sesuai dengan dugaan bahwa semakin besar jumlah iterasi maka hasil yang diperoleh akan semakin baik. Hal tersebut berkaitan dengan semakin banyak jumlah iterasi maka *search space* dari algoritma ini akan bertambah sehingga keragaman akan benar-benar terjamin.
- Dilihat dari aspek probabilitas mutasi, dibuktikan bahwa probabilitas mutasi 0.09 memberikan hasil segmentasi yang paling baik dari segi *precision* dan *recall*.
- Dilihat dari aspek tipe *crossover*, dibuktikan bahwa tipe *crossover two point crossover* memberikan hasil segmentasi yang paling baik dari segi *precision* dan *recall*.
- Dilihat dari aspek banyaknya jumlah segmen, dibuktikan bahwa semakin banyak jumlah segmen yang perlu ditemukan oleh proses segmentasi akan menurunkan nilai *precision* dan *recall*, namun untuk nilai *recall* pada dokumen dengan jumlah segmen 10 dan 30 relatif seimbang. Hal tersebut dirasa berkaitan dengan meningkatnya tingkat kesulitan bagi proses segmentasi untuk menentukan letak *boundary* sebenarnya seiring dengan bertambahnya jumlah komposisi dokumen penyusunnya.
- Dilihat dari aspek kemiripan domain dokumen penyusun , dibuktikan bahwa nilai *recall* akan semakin menurun seiring dengan semakin miripnya domain dokumen penyusun tersebut dan nilai *precision* yang tertingi diperoleh pada dokumen dengan domain dokumen penyusun yang mirip. Penurunan nilai *recall* ini sesuai dengan dugaan bahwa dengan semakin miripnya domain dokumen penyusun maka proses segmentasi akan semakin sulit untuk menentukan letak *boundary* yang sebenarnya karena dengan semakin miripnya domain maka kalimat-kalimat yang digunakan pada dokumen tersebut akan semakin mirip. Domain dokumen mirip memiliki nilai *precision* yang paling tinggi dirasakan berkaitan

dengan pemilihan domain dokumen ini berasal dari domain abstrak tulisan ilmiah yang memiliki komposisi kalimat lebih baku dan lebih teratur dibandingkan dengan domain dokumen artikel media massa (terutama untuk domain artikel *travel* dan properti), sehingga meminimalkan jumlah segmen yang ditemukan oleh proses segmentasi. Untuk domain dokumen sangat mirip memiliki nilai *precision* yang paling rendah dirasakan berkaitan dengan menurun derastisnya nilai *recall* pada jenis dokumen ini (jumlah segmen benar yang ditemukan oleh proses segmentasi jauh lebih sedikit), sehingga turut mempengaruhi nilai *precision* yang dihasilkan.

- Hasil percobaan membandingkan hasil segmentasi antara metode *genetic algorithm* dengan metode Texttiling menunjukan bahwa hasil segmentasi dengan metode Texttiling menggunakan parameter tertentu akan memberikan hasil yang lebih baik dibandingkan dengan metode *genetic algorithm*.
- Metode Texttiling yang digunakan pada percobaan masih memerlukan intervensi manual untuk menentukan parameter-parameter yang dibutuhkan.