ISSN : 2301-7201

ITSMART

Volume 4. Nomor 2. Desember 2015

jurnal teknologi & informasi

JURNAL ITSMART
Teknik Informatika FMIPA UNS

Penanggung Jawab
Kepala Program Studi Informatika FMIPA UNS

Pemimpin Redaksi
Rini Anggrainingsih, M.T

Sekretaris
Afrizal Doewes, S.Kom, M.eng

Bendahara
Esti Suryani, S.Si, M.Kom

Mitra Bestari
Dr. Ir Abdul Kadir, M.M, M.T (UTM Malaysia)
Dr Lukito Edi Nugroho (UGM)
Dr. Anto Satriyo Nugroho (BPPT)

Dewan Redaksi
Dr. Wiranto
Drs. Bambang Harjito, Ph.D
Heri Prastyawan, Ph.D
Umi Salamah, S.Si, M.Kom
Wiharto, S.T, M.Kom

Redaksi Pelaksana
Ristu Saptono, S.Si, M.T
Abdul Aziz, S.Kom, M.Cs
Sari Widya Sihwi, S.Kom, M.TI
Hasan Dwi Cahyono, S.Kom, M.Kom

Alamat Redaksi :
Program Studi Informatika FMIPA UNS
Jl. Ir. Sutami 36A Kentingan Surakarta
Telp. (0271) 663451, Fax. (0271) 663451
Email : itsmart_uns@yahoo.co.id

ISSN : 2301–7201

DAFTAR ISI

Pembangunan Sistem Informasi Presensi Perkuliahan Menggunakan Basis Data Terdistribusi Dengan Metode Replikasi-*Asynchronous* 48 – 56

Arifin Satria Ajinusa, Antonius Bima Murti W, Abdul Aziz

Mobile Positioning Menggunakan Network Identity Sebagai Sistem Absensi Berbasis Mobile Studi Kasus: Universitas Sebelas Maret 57 – 62

Endar Suprih Wihidayat, Azis Rahmanto, Afrizal Doewes

Optimasi Produksi Menggunakan Algoritma Fuzzy Linear Programming (Studi Kasus : Produksi Tas UKM Cantik Souvenir) 63 – 72

Lia Primadani, Sarngadi Palgunadi, Bambang Harjito

Analisis Perbandingan Metode Vector Space Model dan Weighted Tree Similarity dengan Cosine Similarity pada kasus Pencarian Informasi Pedoman Pengobatan Dasar di Puskesmas 73 – 83

Viko Basmalah Wicaksono, Ristu Saptono, Sari Widya Sihwi

Implementasi *Iterative Dichotomiser 3* Pada Data Kelulusan Mahasiswa S1 di Universitas Sebelas Maret 84 – 91

Thanh Thi Bi Dan, Sari Widya Sihwi, Rini Anggrainingsih

Pemodelan Penjadwalan Multilevel Feedback Queue Menggunakan Dynamic Time Quantum Pada Kasus Pemesanan Makanan di Restoran 92 – 100

Tri Wahyu Prasetyo, Wiharto, Afrizal Doewes

INDEX JUDUL

Pembangunan Sistem Informasi Presensi Perkuliahan Menggunakan Basis Data Terdistribusi Dengan Metode Replikasi-Asynchronous, 48 – *56*

Mobile Positioning Menggunakan Network Identity Sebagai Sistem Absensi Berbasis Mobile Studi Kasus: Universitas Sebelas Maret, *57 – 62*

Optimasi Produksi Menggunakan Algoritma Fuzzy Linear Programming (Studi Kasus : Produksi Tas UKM Cantik Souvenir), *63 – 72*

Analisis Perbandingan Metode Vector Space Model dan Weighted Tree Similarity dengan Cosine Similarity pada kasus Pencarian Informasi Pedoman Pengobatan Dasar di Puskesmas, *73 – 83*

Implementasi Iterative Dichotomiser 3 Pada Data Kelulusan Mahasiswa S1 di Universitas Sebelas Maret, *84 – 91*

Pemodelan Penjadwalan Multilevel Feedback Queue Menggunakan Dynamic Time Quantum Pada Kasus Pemesanan Makanan di Restoran, *92 – 100*

INDEX PENULIS

Abdul Aziz, 48 - 56

Afrizal Doewes, 57 - 62, 92 - 100

Antonius Bima Murti W, 48 - 56

Arifin Satria Ajinusa, 48 - 56

Azis Rahmanto, 57 - 62

Bambang Harjito, 63 - 72

Endar Suprih Wihidayat, 57 - 62

Lia Primadani, 63 - 72

Rini Anggrainingsih, 84 – 91

Ristu Saptono, 77 - 87

Sari Widya Sihwi, 73 - 83, 84 - 91

Sarngadi Palgunadi, 63 - 72

Thanh Thi Bi Dan, 84 - 91

Tri Wahyu Prasetyo, 92 - 100

Viko Basmalah Wicaksono, 73 - 83

Wiharto, 92 - 100

PEMBANGUNAN SISTEM INFORMASI PRESENSI PERKULIAHAN MENGGUNAKAN BASIS DATA TERDISTRIBUSI DENGAN METODE REPLIKASI-*ASYNCHRONOUS*

Arifin Satria Ajinusa
Jurusan Informatika
Universitas Sebelas Maret
Jl. Ir. Sutami 36A Kentingan
Surakarta
Ajinusa92@gmail.com

Antonius Bima Murti Wijaya
Jurusan Informatika
Universitas Sebelas Maret
Jl. Ir. Sutami 36A Kentingan
Surakarta
bimamurti@uns.ac.id

Abdul Aziz
Jurusan Informatika
Universitas Sebelas Maret
Jl. Ir. Sutami 36A Kentingan
Surakarta
Abdul_7773@yahoo.com

Abstrak — Presensi merupakan salah satu aktivitas yang biasa dilakukan di setiap institusi pendidikan dalam mengetahui kehadiran siswa atau mahasiswa. Aktivitas presensi di Universitas Sebelas Maret masih dilakukan secara manual, sehingga menimbulkan ketidak efektifan dalam rekapitulasi oleh operator jurusan. Oleh karena itu, dibutuhkan sebuah sistem presensi yang mampu mengatasi masalah ini. Sistem presensi RFID adalah sebuah sistem otomatis yang digunakan untuk aktivitas presensi suatu entitas yang terdaftar pada suatu organsisasi. Konsep basis data terdistribusi diterapkan dalam pembangunan sistem ini, yang bertujuan untuk meningkatkan kinerja sistem itu sendiri. Dengan menerapkan konsep basis data terdistribusi, ketersediaan data dapat terjamin karena terdapat replika basis data. Metode asynchronous akan dilibatkan dalam proses transaksi antar basis data agar lalu lintas jaringan tidak akan berjalan terlalu padat. Untuk mengatasi masalah kegagalan transaksi digunakan metode rollback pada basis data. Hasil pengujian menunjukkan bahwa proses transaksi pengiriman dan penarikan data telah berhasil dilakukan dengan melakukan simulasi transaksi antar komputer client-server. Selain itu pengujian terhadap penanganan kegagalan ketika proses pengiriman maupun penarikan data, sistem dapat melakukan rollback untuk mengembalikan basis data dalam keadaan semula.

Kata Kunci : Basis data terdistribusi, RFID, replikasi-*asynchronous*.

1. PENDAHULUAN

Presensi merupakan salah satu aktivitas yang biasa dilakukan di setiap institusi pendidikan dalam mengetahui kehadiran siswa atau mahasiswa. Pengajar dapat mengevaluasi kedisiplinan siswa atau mahasiswanya dengan menggunakan presensi. Universitas merupakan sebuah lembaga pendidikan yang memperhatikan kualitasnya melalui daftar kehadiran mahasiswa.

Universitas Sebelas Maret (UNS) adalah salah satu universitas yang berada di Indonesia yang ingin meningkatkan kualitas pendidikannya melalui penataan presensi mahasiswa. Aktivitas presensi di UNS masih dilakukan secara manual, di mana memiliki banyak kekurangan, diantaranya adalah aktivitas rekapitulasi yang dilakukan oleh operator jurusan maupun fakultas memakan waktu cukup lama. Kehilangan data presensi juga mungkin terjadi mengingat data presensi yang diperoleh berasal dari lembaran kertas absensi yang dapat hilang, oleh karena itu tidak cukup aman untuk disimpan.

Sistem Informasi Akademik (SIAKAD) UNS adalah sistem yang mengelola data informasi akademik universitas, salah satunya adalah rekapitulasi data presensi mahasiswa. Berdasarkan hasil wawancara yang telah dilakukan dengan beberapa admin program studi maupun fakultas di antaranya program studi Teknik Arsitektur, program studi Teknik Sipil, program studi Informatika, program studi D3 Teknik Informatika, Fakultas Hukum, Fakultas Ilmu Sosial dan Ilmu Politik, Fakultas Sastra dan Seni Rupa dan Fakultas Pertanian, proses rekap yang dilakukan ke sistem SIAKAD memakan waktu yang cukup lama. Proses pengumpulan data pun sering ditemukan banyak masalah, antara lain terdapat dosen yang tidak mengumpulkan lembar presensi perkuliahan. Hal ini membuat data rekapitulasi presensi mahasiswa tidak lengkap. Bahkan, ada pula beberapa program studi dan fakultas yang tidak menggunakan sistem rekap presensi mahasiswa di SIAKAD di antaranya program studi Teknik Arsitek, program studi Informatika, program studi D3 Teknik Informatika, Fakultas Pertanian, Fakultas Sastra dan Seni Rupa, mengingat proses yang dilakukan cukup lama dan tidak efektif.

Sistem presensi otomatis dengan teknologi *Radio Frequency Identification* (RFID) merupakan salah satu solusi untuk mengatasi masalah ini. Sistem ini dapat digunakan di sekolah, perguruan tinggi, dan tempat kerja. Data presensi diperlukan sebagai bukti kedisiplinan mahasiswa maupun pegawai dalam sebuah organisasi atau perusahaan. [1] Sistem presensi RFID adalah sebuah sistem otomatis yang digunakan untuk aktivitas presensi suatu entitas yang terdaftar pada suatu organsisasi. Setiap pegawai menggunakan sebuah kartu RFID dan reader untuk merekam data ketika pegawai masuk maupun keluar. Kartu terdiri dari sebuah nomor identifikasi unik yang disebut dengan *electronic product code* (EPC). [2]

Replikasi dalam proses komputasi memungkinkan berbagi informasi atau data untuk memastikan konsistensi antar sumber daya informasi yang redundan, untuk meningkatkan kehandalan, toleransi kesalahan maupun aksesibilitas. [3] Replikasi adalah proses membuat dan *maintenance database* duplikat dalam suatu sistem basis data terdistribusi. [4]

Sistem presensi mahasiswa menggunakan teknologi RFID memiliki beberapa kelebihan, di antaranya dapat mengurangi tindak indisipliner yang dilakukan mahasiswa karena terlambat dalam mengikuti perkuliahan maupun melakukan penitipan absen, karena sistem ini akan menampilkan informasi jumlah kehadiran mahasiswa yang mengikuti perkuliahan secara *real time*. Selain itu, proses

rekapitulasi dapat dilakukan dengan lebih cepat dan tertata, data yang dihasilkan lebih akurat dibandingkan presensi yang dilakukan secara manual.

Jika dalam pembangunan sistem ini menerapkan basis data terpusat maka akan berdampak pada beban kerja *server*, apabila *server* utama mati maka aktivitas presensi tidak dapat dilakukan. Selain itu, juga akan berdampak pada aktivitas presensi setiap mahasiswa yang akan mengakibatkan *delay* yang cukup lama. Oleh karena itu, penulis menerapkan konsep basis data terdistribusi dalam pembangunan sistem ini, dimana dalam RFID akan bertindak sebagai *client.* Dalam RFID terdapat basis data yang menampung data sementara selama proses presensi dilakukan, yang kemudian data tersebut akan dikirimkan ke basis data *server*. Untuk menjaga agar lalu lintas jaringan tidak terlalu padat, penelitian ini menerapkan metode replikasi *asynchronous* dalam proses transaksi antar basis data. Replikasi ini dilakukan secara periodik, yaitu setiap hari sistem akan melakukan transaksi data antara *server* dengan RFID. Teknik replikasi yang digunakan adalah *partial replication*. Namun, secara arsitektur menggunakan teknik fragmentasi horizontal yang melibatkan variabel waktu. Terputusnya komunikasi antar basis data sangat mungkin terjadi karena beberapa hal seperti listrik mati atau terputusnya koneksi internet sehingga *server* tidak dapat berkomunikasi dengan RFID. Oleh karena itu, penulis juga menerapkan metode untuk antisipasi kegagalan transaksi dalam pembangunan sistem ini yaitu dengan metode *rollback* basis data.

2. DASAR TEORI

2.1. Sistem Presensi

Presensi mahasiswa adalah masalah penting untuk perguruan tinggi, karena banyak perguruan tinggi mengevaluasi presensi mahasiswa untuk banyak kepentingan seperti nilai mahasiswa. Beberapa perguruan tinggi memilih untuk menggunakan lembar kertas untuk mengontrol kehadiran mahasiswa. [5]

Metode konvensional dalam presensi dengan cara memanggil nama atau tanda tangan dalam kertas sangat mengkonsumsi waktu, kurang aman dan tidak efisien. Sistem presensi RFID memberikan manfaat di antaranya presensi *real time* akan dapat diperoleh secara akurat. [1]

2.2. Basis Data Terdistribusi

Sistem basis data terdistribusi adalah sistem yang memiliki data yang terdistribusi dan ter-replikasi pada beberapa lokasi, tidak seperti sistem basis data terpusat, dimana hanya terdapat satu salinan data yang disimpan. Data mungkin ter-replikasi melalui jaringan menggunakan fragmentasi horizontal dan vertikal. [6]

2.3. Replikasi *(Replication)*

Replikasi berarti bahwa kita menyimpan beberapa salinan dari relasi atau fragmen relasi dalam suatu basis data. Seluruh relasi dapat direplikasi pada satu situs atau lebih. Demikian pula, satu atau lebih fragmen relasi dapat direplikasi pada situs lain. [7]

2.2.1 Tipe-tipe Replikasi

Strategi replikasi dapat diklasifikasikan berdasarkan dua karakteristik dasarnya yaitu Dimana (*Where*) dan Kapan (*When*) [8]:

a. Karakterisitik *When*:

- *Sychronous Replication (Master-Slave replication)*

  Replikasi *Synchronous* bekerja pada prinsip *Two-Phase commit protocol*. Ketika *client* mengirimkan permintaan *update* ke *master database*, master sistem terhubung ke semua sistem lainnya (*slave databases*). Mengunci semua database pada tingkat *record* dan kemudian meng-*update* secara bersamaan. Jika salah satu dari basis data tidak tersedia, data tidak dapat di-*update*. Strategi replikasi *Synchronous* berfokus pada konsistensi data, sehingga membutuhkan ketersediaan semua situs pada saat propagasi *update*. [8]

- *Asynchronous Replication (Store and Forward replication)*

  Metode replikasi yang paling cocok untuk lingkungan di mana perubahan data jarang terjadi. Seluruh basis data *snapshot* dibuat dan dikirimkan untuk pelanggan ketika sinkronisasi terjadi. Replikasi snapshot merupakan cara termudah dalam teknik replikasi, karena tabel ini di-*refresh* setiap kali ketika membutuhkan menyalin semua data. [9]

- *Merge Replication*

  Merge Replication adalah proses mendistribusikan data dari *Publisher* ke *Subscribers*, memungkinkan keduanya untuk melakukan pembaruan data ketika terhubung (*connected)* maupun tidak terhubung *(disconnected)*. [10]

b. Karakterisitik *Where*:

- *Update Everywhere Replication*

  Update Everywhere tidak memiliki simpul (*server*) utama, sebagai gantinya semua node dalam sistem dianggap sama. Itu berarti setiap *node* basis data dalam sistem dapat menangani transaksi update bukan hanya di satu node utama. [3]

- *Primary Copy Replication*

  Dalam metode ini hanya terdapat satu salinan utama (*primary copy*) yang dapat di-update dan salinan lainnya (*secondary* atau *slave*) di-update mengacu pada perubahan yang terjadi di salinan utama (*master*). [8]

2.2.2 *Fault Tolerance*

Toleransi kesalahan menggambarkan sebuah prosedur atau teknik yang menjaga data tetap konsisten ketika terjadi kegagalan.

Failover merupakan proses pemindahan basis data, metadata dan koneksi pengguna dari situs utama yang gagal atau *shutdown* ke situs sekunder sehingga pengguna tetap bisa mengakses data juga menjaga ketersediaan. Terdapat dua metode yang digunakan diantaranya *Active-Active* dan *Active-Passive.* Pada *Active-Passive,* servis disediakan oleh salah satu node yakni node primer atau node sekunder.

Proses *failover* menggunakan teknik *failback* untuk mengembalikan situs utama dalam keadaan konsisten. [8]

2.2.3 Teknik Replikasi

Dalam teknik replikasi, proses penyimpanan data dibagi menjadi dua yakni *Full Replication* dan *Partial Replication*. *Full Replication* adalah teknik replikasi dengan cara mereplikasi seluruh data dari *database master*. Sedangkan *Partial Replication* merupakan teknik replikasi dengan cara mereplikasi bagian tertentu dari *database master*, dan juga dapat mereplikasi bagian yang berbeda untuk pengguna yang berbeda. Dengan menggunakan replika parsial, kita dapat menduplikasi data yang hanya dibutuhkan oleh pengguna. [10]

2.4. *Fragmentation*

Fragmentasi terdiri dari pemecahan relasi ke dalam relasi atau fragmen yang lebih kecil dan menyimpan fragmen-fragmen tersebut, memungkinkan pada situs yang berbeda. Dalam Fragmentasi Horizontal, setiap fragmen terdiri dari sebuah sekumpulan baris dari relasi asli. Dalam Fragmentasi Vertikal, setiap fragmen terdiri dari sekumpulan kolom dari relasi asli. [11]

2.1. *Systems Development Life Cycle* (SDLC)

SDLC adalah proses mempelajari bagaimana sebuah sistem informasi dapat mendukung kebutuhan bisnis, mendesain sistem, membangun sistem, dan mengimplementasikannya kepada pengguna. [12]

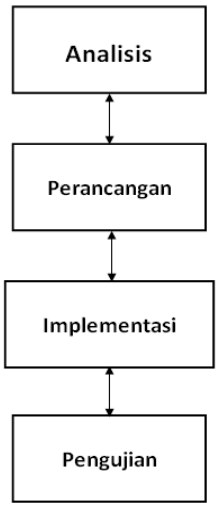
2.5. RFID

Radio Frequency Identification (RFID) adalah sistem identifikasi yang menggunakan radio frekuensi untuk mengidentifikasi objek dan lokasi. [13]

RFID merupakan salah satu jenis teknologi *Automatic Identification and Data Capture* (AIDC) yang cepat dan handal dalam mengidentifikasi suatu benda atau objek. Terdapat dua komponen utama dalam RFID, yakni RFID *Reader* yang mentransmisikan dan menerima sinyal, dan RFID Tag yang melekat pada objek. [1]

3. METODOLOGI PENELITIAN

Langkah-langkah yang dilakukan dalam penelitian ini ditunjukkan oleh Gambar 1.



Gambar 1. Alur Penelitian

3.1. Analisis Proses Bisnis

3.1.1 Studi Literatur

Pada penelitian ini diawali dengan studi literatur dengan mengumpulkan referensi literatur terkait sebagai dasar penelitian. Referensi diperoleh dari jurnal penelitian, buku dan situs internet terkait penelitian ini. Referensi yang dikumpulkan di antaranya mengenai teknologi RFID dan metode dalam replikasi *database*.

3.1.2 Tahap Pengumpulan Data

Pengumpulan data dilakukan sebagai lanjutan dari studi pustaka. Data berfungsi sebagai objek yang diolah dan diproses untuk diimplementasikan pada sistem. Wawancara dilakukan terkait *requirement gathering* yaitu berupa kebutuhan-kebutuhan fungsional dan non fungsional. Pengambilan data dalam penelitian ini menggunakan data yang bersifat primer dan sekunder sebagai bahan penelitian.

3.2. Perancangan Sistem

Perancangan sistem berupa pemetaan dari kebutuhan fungsional yang telah teranalisa ke dalam suatu diagram. Dalam proses perancangan sistem terdapat beberapa diagram yang digunakan yakni *Sequence Diagram, Activity Diagram, Class Diagram* dan perancangan arsitektur basis data terdistribusi. Sistem dirancang dengan menggunakan pendekatan *Object Oriented Programming (OOP)* yang akan memberikan kemudahan ketika dilakukan *maintenance*.

3.3. Implementasi

Implementasi dilakukan dengan membangun sistem sesuai dengan perancangan yang telah dilakukan. Ruang lingkup perangkat keras dan perangkat lunak yang digunakan untuk proses pengembangan sistem ini antara lain sebagai berikut:

a. Perangkat Keras
- RFID UHF Reader
- RFID UHF Tag
- Raspberry Pi
- Micro SD 10 Gb
- UTP *Cable*, RJ45

b. Perangkat Lunak
- *Progamming Package* :
  - Bahasa Pemrograman : PHP 5.3, Java
  - *Web Server* : Apache 2.0
  - *Database Server* : Microsoft SQL *Server* 2008
  - *Database Manager* : Microsoft SQL Management Studio 10
- *Editing Tool* : Netbeans IDE 7.4
- *Supporting Tools* :
  - MySQL –Front version: 5.3 (Build:4.123)
  - Microsoft Word 2007
  - Microsoft Office Visio 2007
  - StarUML version: 5.0.2.1570
  - Adobe Photoshop CS3

3.4. Pengujian

Setelah sistem selesai dibangun, dilakukan pengujian sistem yang bertujuan untuk mengetahui kesesuaian fungsional sistem. Dalam pengujian ini dilakukan dengan metode *Black Box Testing* terutama untuk fungsi-fungsi yang bersifat *critical*.

4. HASIL DAN PEMBAHASAN

4.1. Analisis Proses Bisnis

Dalam implementasi sistem ini akan melibatkan beberapa pihak di antaranya sistem RFID, dosen, admin jurusan, admin *server,* dan waktu transaksi otomatis. Setiap pihak memiliki perannya masing-masing dalam jalannya sistem ini.

Dalam pelaksanaan presensi perkuliahan, mahasiswa dapat melakukan presensi dengan menggukan kartu mahasiswa sebagai media yang akan diidentifikasi oleh RFID. Setiap kartu mewakili setiap mahasiswa, sehingga kartu tersebut memiliki nomor unik untuk membedakan satu sama lain. Mahasiswa juga dapat meminjam kartu ke admin jurusan jika tidak membawa kartu miliknya. Data-data presensi mahasiswa ini akan ditampung sementara dan disimpan dalam basis data oleh sistem RFID.

Sebelum mahasiswa dapat melakukan aktivitas presensi, dosen akan melakukan *check-in* ruangan terlebih dahulu. Ketika dosen melakukan *check-in*, akan diberikan waktu 15 menit untuk mahasiswa melakukan aktivitas presensi melalui RFID. Ketika mahasiswa datang terlambat, maka status presensi mahasiswa akan dianggap alpha. Selain itu, dosen juga dapat melihat rekap data presensi perkuliahan dan data presensi di RFID melalui website. Data presensi yang terdapat di RFID akan muncul ketika dosen telah melakukan *check-in* ruang perkuliahan.

Pengelolaan data presensi mahasiswa dapat dilakukan oleh admin jurusan. Ketika dosen tidak dapat melakukan aktivitas *check-in* kelas, dosen dapat meminta admin jurusan untuk melakukan *check-in* melalui website. Selain itu, admin jurusan juga dapat mengelola data presensi mahasiswa seperti mencetak laporan presensi mahasiswa dan mengubah status mahasiswa ketika terdapat mahasiswa yang ijin maupun sakit sehingga tidak dapat mengikuti perkuliahan. Rekap data presensi mahasiswa juga dapat dilihat melalui website. Admin jurusan dapat mengelola data RFID setiap ruangan di jurusan tersebut, seperti dengan mengatur ip address RFID pada setiap ruangan. Admin jurusan dapat mengaktifkan status ajaran pada jurusan masing-masing, yang berfungsi untuk memberikan status aktif pada status ajaran dan meng-*import* data Kartu Rencana Studi mahasiswa dari SIAKAD dimana aktivitas ini hanya dilakukan sekali dalam 1 semester. Kemudian, admin jurusan juga dapat melakukan transaksi data secara manual terhadap transaksi yang gagal dilakukan ketika dilakukan secara otomatis oleh sistem. Aktivitas ini hanya dapat dilakukan ketika admin jurusan telah mengaktifkan status ajaran pada jurusan tersebut. Admin jurusan juga dapat mengelola data peminjaman kartu oleh mahasiswa.

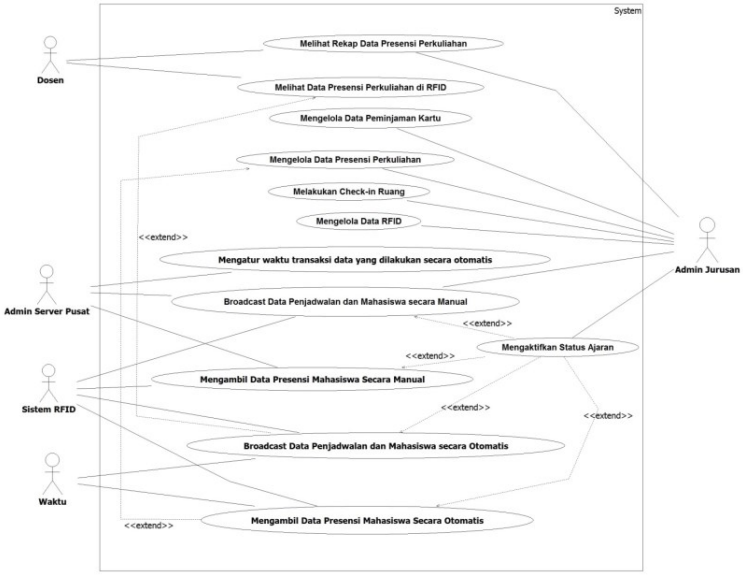
Admin *server* pusat memiliki peran penting dalam mengawasi transaksi data ke setiap RFID. Di antaranya dengan melihat laporan transaksi yang telah dilakukan, dan juga dapat melakukan transaksi ulang secara manual terhadap transaksi yang gagal dilakukan. Proses transaksi hanya dapat dilakukan ketika status ajaran di setiap jurusan telah dimulai oleh admin jurusan.

Transaksi data otomatis akan dilakukan pada waktu yang telah ditentukan. Aktivitas yang dilakukan dalam transaksi otomatis adalah pengiriman data mata kuliah dan mahasiswa ke RFID dan penarikan data presensi mahasiswa dari RFID.

4.2. Perancangan Sistem

4.2.1 *Use Case Diagram*

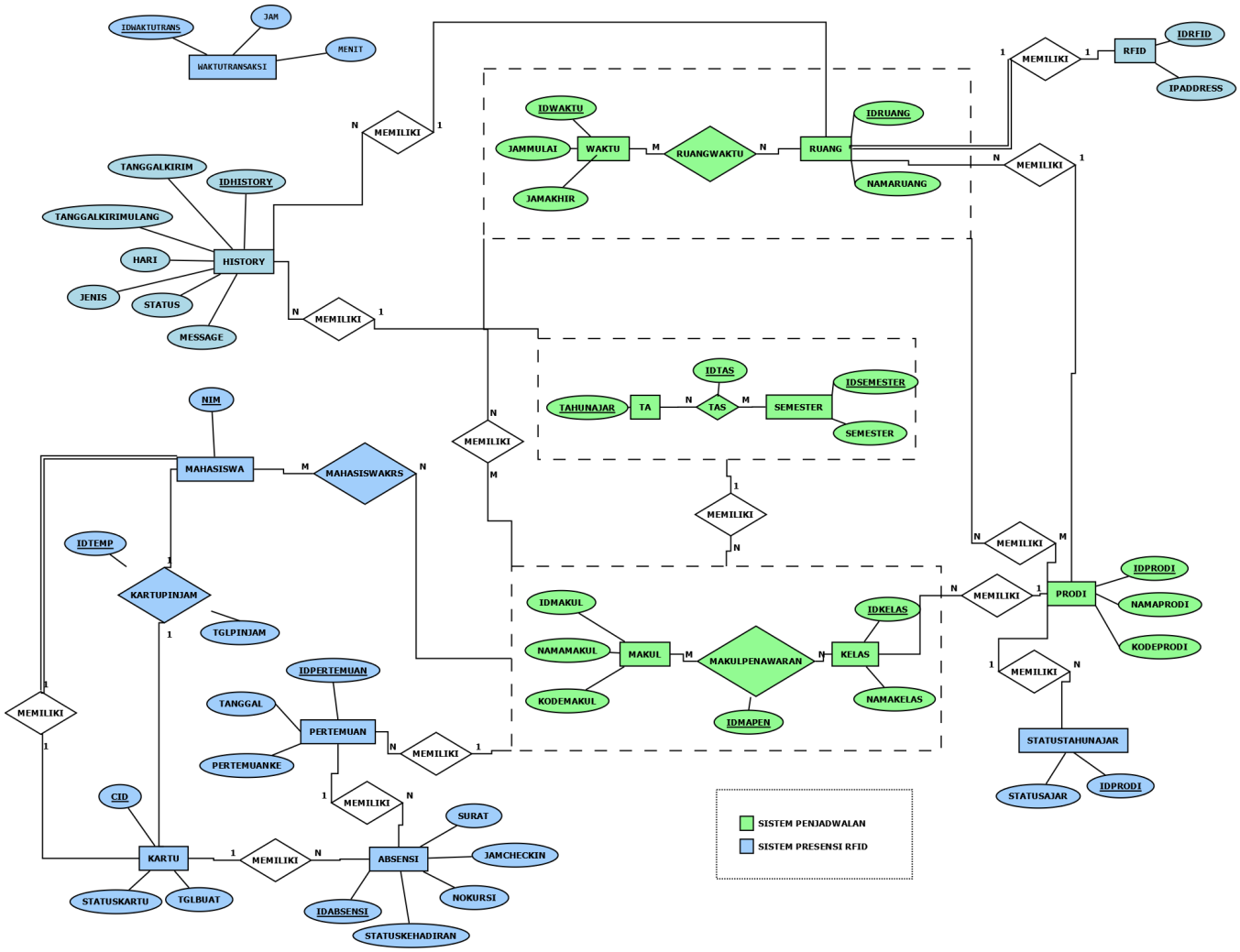
Pada Gambar 2 menunjukkan *Use Case Diagram* Sistem Informasi Presensi Perkuliahan.



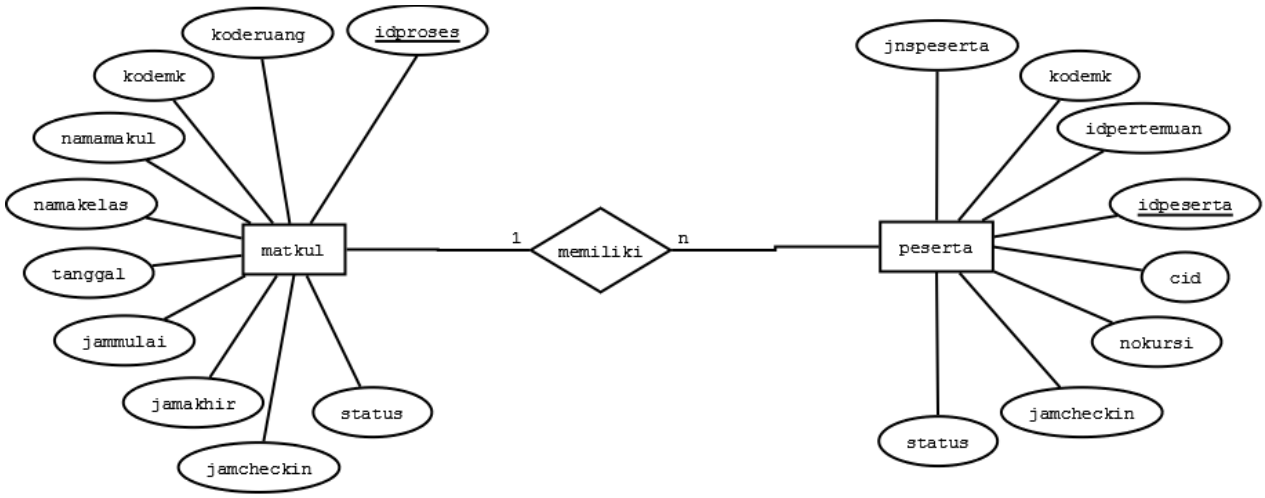
Gambar 2. *Use Case Diagram*

4.2.2 *Entity Relationship Diagram* (ERD)

ERD *server* Sistem Informasi Presensi Perkuliahan ditunjukkan pada Gambar 3 dan ERD Raspberry Sistem Informasi Presensi Perkuliahan ditunjukkan pada Gambar 4.

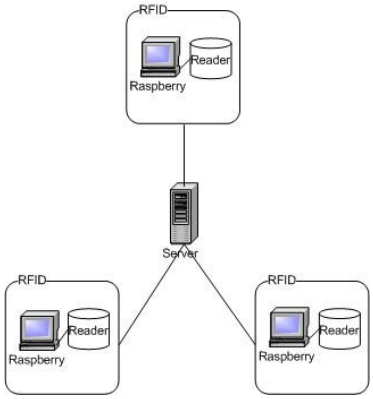


Gambar 3. ERD *Server*



Gambar 4. ERD Raspberry

4.2.3 Arsitektur Sistem

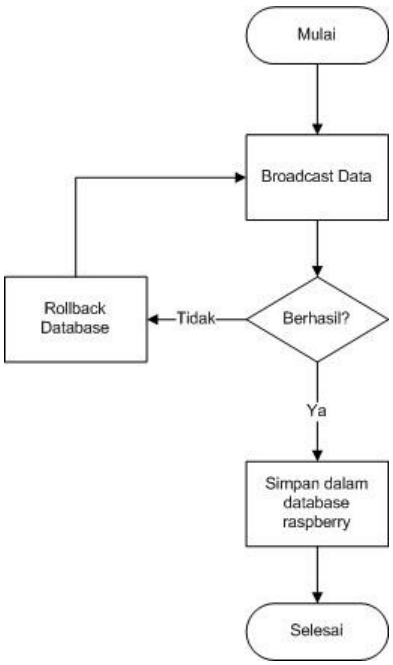


Gambar 5. Arsitektur Sistem.

Pada Gambar 5 menunjukkan arsitektur pembangunan sistem ini dimana terdapat 1 buah *server* yang berfungsi untuk mengatur proses transaksi dengan basis data RFID di setiap ruangan dan sebagai repositori utama dalam penyimpanan data mahasiswa, mata kuliah, jadwal, dan presensi mahasiswa.

4.2.3.1. Flowchart Pengiriman Data

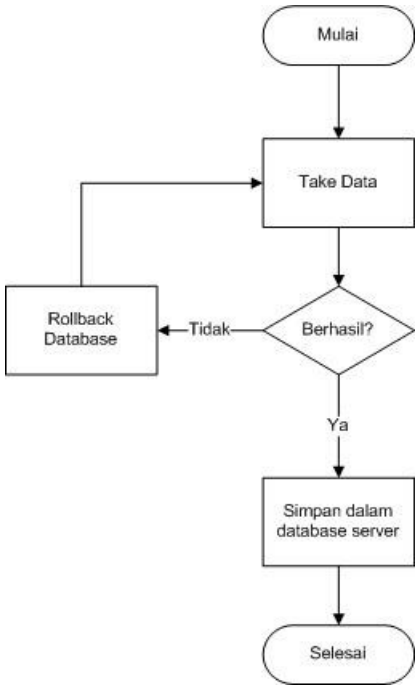
Dalam proses pengiriman data ke RFID seperti yang ditunjukkan pada Gambar 6 dilakukan dengan metode replikasi *asynchronous.* Data yang dikirimkan adalah data mata kuliah dan data mahasiswa yang mengambil mata kuliah. Data akan dikirimkan pada waktu tertentu dan sesuai dengan kelas maupun ruangan yang telah ditetapkan dalam penjadwalan. Ketika dalam proses transaksi terjadi kegagalan, maka basis data akan melakukan *rollback.* Artinya akan mengembalikan basis data dalam keadaan semula. Jika berhasil maka data akan disimpan ke dalam basis data RFID.



Gambar 6. *Flowchart* Pengiriman Data.

4.2.3.2. Flowchart Penarikan Data

Sama dengan proses pengiriman data ke RFID, dalam proses penarikan data dari RFID ke *server* utama seperti yang ditunjukkan pada Gambar 7 dilakukan dengan metode replikasi *asynchronous.* Data akan ditarik pada waktu tertentu. Ketika dalam proses transaksi terjadi kegagalan, maka basis data akan melakukan *rollback.* Artinya akan mengembalikan basis data dalam keadaan semula. Jika berhasil maka data presensi akan disimpan ke dalam basis data *server*.



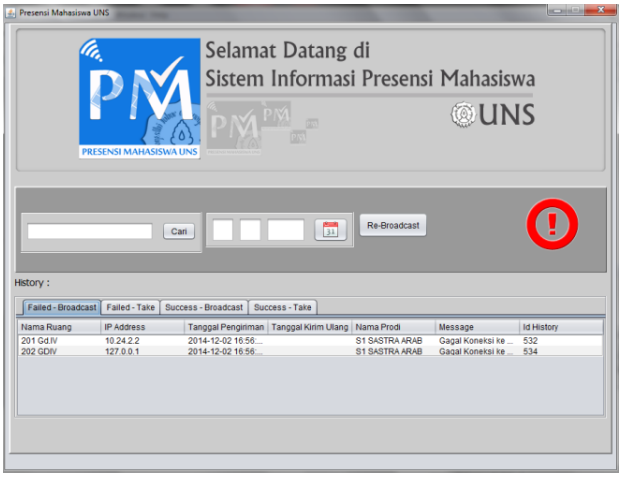
Gambar 7. *Flowchart* Penarikan Data.

4.2.3.3. Implementasi Sistem

Sistem dibuat dengan 2 bahasa pemrograman yaitu Java dan PHP. Dalam versi *desktop,* sistem di-*install* di *server* utama yang dapat dikelola oleh operator *server* utama, sedangkan dalam versi *website* dapat dikelola oleh operator jurusan.

Aplikasi *desktop* berfungsi untuk melakukan transaksi data mahasiswa dan mata kuliah secara otomatis maupun manual. Data yang telah dilakukan transaksi akan disimpan dalam *history* transaksi. *History* terdiri dari 4 macam yaitu *history* gagal mengirim data, gagal menarik data, berhasil mengirim data, dan berhasil menarik data.

Pada proses transaksi pengiriman data, data-data yang gagal dilakukan transaksi akan tersimpan dalam *history* seperti yang ditunjukkan pada Gambar 8. Transaksi gagal mengirim data dapat dilihat pada tab "Failed-Broadcast". Dari data *history* tersebut, operator dapat melakukan transaksi ulang secara manual dengan cara menekan tombol "Re-Broadcast".



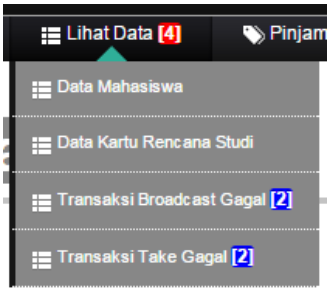
Gambar 8. Tampilan Aplikasi Sistem Presensi *Broadcast* Data versi *Desktop*.

Seperti dalam proses transaksi pengiriman data ke RFID, dalam proses transaksi penarikan data dari RFID, data-data yang gagal dilakukan transaksi akan tersimpan dalam *history* seperti yang ditunjukkan pada Gambar 9. Transaksi yang gagal ketika proses menarik data yang dapat dilihat pada tab "Failed-Take". Dari data *history* tersebut, operator dapat melakukan transaksi ulang secara manual dengan cara menekan tombol "Re-Take".

Gambar 9. Tampilan Aplikasi Sistem Presensi *Take* Data versi *Desktop.*

Sama dengan versi *desktop*, aplikasi versi *web* juga memiliki fungsi untuk melakukan transaksi secara manual terhadap transaksi yang gagal dilakukan ketika proses transaksi dilakukan secara otomatis oleh sistem.

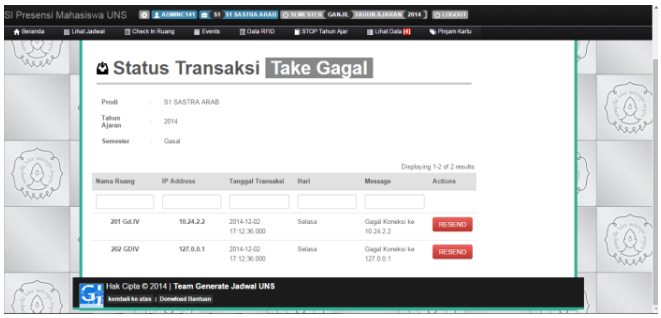
Jika terdapat transaksi yang terjadi kegagalan, maka sistem akan memberikan pemberitahuan terhadap jumlah transaksi yang gagal dilakukan seperti pada Gambar 10**.**



Gambar 10. Tampilan Pemberitahuan Transaksi Gagal

Pada sistem ini user dapat menekan tombol "Resend" untuk melakukan transaksi ulang seperti yang ditunjukkan pada Gambar 11 dan Gambar 12.

Gambar 4.11 Tampilan Aplikasi Sistem Presensi *Broadcast* Data versi *Web.*



Gambar 4.12 Tampilan Aplikasi Sistem Presensi *Take* Data versi *Web.*

4.2.3.4. Hasil Pengujian

Dalam penelitian ini dilakukan pengujian transaksi dan pengujian penanganan kegagalan dengan melakukan simulasi transaksi antara 3 komputer dan 1 RFID, dimana 1 komputer berperan sebagai *server* sedangkan 2 komputer dan 1 RFID berperan sebagai client, sehingga dapat diketahui apakah sistem sudah berjalan sesuai dengan rumusan masalah penelitian. Pengujian dilakukan menggunakan *blackbox testing* seperti yang ditunjukkan pada Tabel 4.1.

Tabel 1. Tabel Pengujian Sistem

No	Perlakuan	Input	Harapan	Hasil
1	Pengujian transaksi dengan melakukan pengiriman data makul	Data mata kuliah dan mahasiswa untuk hari Senin	Transaksi pengiriman data mahasiswa dan mata kuliah	Data mahasiswa dan mata kuliah berhasil di simpan

Tabel 1 Lanjutan. Tabel Pengujian Sistem

No	Perlakuan	Input	Harapan	Hasil
	dan mahasiswa ke masing-masing RFID dengan ipaddress 10.24.2.18, 10.24.2.19, dan 10.24.2.250		berhasil dilakukan	dalam basis data masing-masing rfid dengan ipaddress 10.24.2.18, 10.24.2.19, dan 10.24.2.250
2	Pengujian transaksi dengan melakukan penarikan data presensi mahasiswa dari setiap RFID dengan ipaddress 10.24.2.18, 10.24.2.19, dan 10.24.2.250	Data presensi mahasiswa pada hari Senin dari setiap RFID	Transaksi penarikan data presensi mahasiswa berhasil dilakukan	Data presensi mahasiswa berhasil di simpan dalam basis data *server* dari masing-masing rfid dengan ipaddress 10.24.2.18, 10.24.2.19, dan 10.24.2.250
3	Pengujian terhadap penanganan kegagalan dengan melakukan pengiriman data mahasiswa dan mata kuliah ke rfid dengan ipaddress 10.24.2.250	Melakukan pengiriman data mahasiswa dan mata kuliah ke rfid dengan ipaddress 10.24.2.250	Basis data dapat dikembalikan dalam keadaan semula ketika gagal melakukan pengiriman data.	Basis data berhasil melakukan *rollback*
4	Pengujian terhadap penanganan kegagalan dengan melakukan penarikan data presensi dari rfid dengan ipaddress 10.24.2.250	Melakukan penarikan data presensi mahasiswa dari rfid dengan ipaddress 10.24.2.250	Basis data dapat dikembalikan dalam keadaan semula ketika gagal melakukan penarikan data.	Basis data berhasil melakukan *rollback*

5. KESIMPULAN DAN SARAN

Hasil dari penelitian ini adalah terbangunnya sistem presensi perkuliahan otomatis menggunakan teknologi RFID dengan replikasi-asynchronous di Universitas Sebelas Maret. Untuk mengatasi masalah kegagalan selama proses transaksi, penulis menerapkan metode rollback dalam transaksi basis data. Dalam pengembangan sistem ini menggunakan konsep Software Development Life Cycle (SDLC) yang terdiri dari beberapa tahapan yaitu analisis, perancangan, implementasi, dan pengujian. Pada tahap analisis terdiri dari tahap pengumpulan data dan membuat use case diagram, use case specification, dan Entity Relationship Diagram. Pada tahap perancangan terdiri dari pembuatan Sequence Diagram, Activity Diagram, Class Diagram dan perancangan arsitektur basis data terdistribusi. Pada tahap implementasi dilakukan dengan membangun sistem sesuai dengan perancangan yang telah dilakukan. Pada tahapan terakhir dilakukan pengujian sistem yang bertujuan untuk mengetahui kesesuaian fungsional sistem yaitu dengan metode Black Box Testing. Hasil dari pengujian dengan Black Box Testing adalah setiap fungsional sistem telah berjalan sesuai dengan apa yang diharapkan. Terutama pada pengujian transaksi data menunjukkan bahwa proses transaksi pengiriman dan penarikan data telah berhasil dilakukan dengan melakukan simulasi transaksi antar komputer client-server. Selain itu pengujian terhadap penanganan kegagalan ketika proses pengiriman maupun penarikan data, sistem dapat melakukan rollback untuk mengembalikan basis data dalam keadaan semula.

Adapun saran yang dipertimbangkan untuk pengembangan penelitian ini adalah sistem dikembangkan untuk menangani keamanan dalam proses transaksi baik pada saat melakukan presensi maupun saat transaksi pengiriman data.

6. DAFTAR PUSTAKA

[1] C.S.Karthikeyan and S. Murugeswari, "Anytime Anyplace-Remote Monitoring of Students Attendace Based On RFID and GSM Network," *International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering*, pp. 6154-6159, 2013.

[2] O. G. Chiagozie and O. G. Nwaji, "Radio Frequency Identification (RFID) Based Attendance System with Automatic Door Unit," *Academic Research International*, vol. 2, no. 2, pp. 168-183, March 2012.

[3] S. Sapate and M. Ramteke, "Survey on Comparative Analysis of Database Replication Techniques," *International Journal of IT, Engineering and Applied Sciences Research (IJIEASR)*, vol. 2, no. 3, pp. 72-80, March 2013.

[4] M. M. Oo et al., "Fault Tolerance by Replication of Distributed Database in P2P System using Agent Approach," *International Journal of Computers*, vol. 4, no. 1, pp. 9-18, 2010.

[5] Ghodekar, V. et al., "Automated Attendance system with RFID through SMART CARD," *International Journal of Engineering Research & Technology (IJERT)*, vol. 2, no. 10, pp. 2724-2728, October 2013.

[6] G Alkhatib and R. S. Labban, "Transaction Management in Distributed Database Systems: the Case of Oracle's Two-Phase Commit," *Journal of Information Systems Education*, vol. 13, no. 2, pp. 95-103.

[7] R. Ramakrishnan and J. Gehrke, *Database Management System - Third Edition*, 3rd ed.: McGraw-Hill Education, 2003.

[8] S. A. Moiz et al., "Database Replication: A Survey of Open Source and Commercial Tools," *International Journal of Computer Applications (0975 – 8887)*, vol. 13, no. 6, pp. 1-8, January 2013.

[9] A. Hussain and M. N. A. Khan, "Discovering Database Replication Techniques in RDBMS," *International Journal of Database Theory and Application*, vol. 7, no. 1, pp. 93-102, 2014.

[10] M. C. Mazilu, "Database Replication," *Database Systems Journal*, vol. 1, no. 2, pp. 33-38, 2010.

[11] Raghu Ramakrishnan and Johannes Gehrke, *Database Management System - Third Edition*, 3rd ed.: McGraw-Hill Education, 2003.

[12] Alan Dennis, Barbara Haley Wixom, and Roberta M Roth, *Systems Analysis & Design*, 4th ed., Beth Lang Golub, Ed.: John Wiley & Sons, 2009.

[13] Gangi Raghu Ram, N Rajesh Babu, N P Sudhakar, Balanagu Raviteja, and K Rammohanarao, "Tracking Objects Using RFID and Wireless Sensor Networks," *International Journal of Engineering Science & Advanced Technology*, vol. 2, no. 3, pp. 513 – 517, May-June 2012.

Mobile Positioning Menggunakan *Network Identity* Sebagai Sistem Absensi Berbasis *Mobile* Studi Kasus: Universitas Sebelas Maret

Endar Suprih Wihidayat,S.T.,M.Eng.
Jurusan D3 Teknik Informatika
Fakultas MIPA
Universitas Sebelas Maret
endars@staff.uns.ac.id

Azis Rahmanto
Jurusan Informatika
Fakultas MIPA
Universitas Sebelas Maret
azis@student.uns.ac.id

Afrizal Doewes S.Kom., M.Sc.
Jurusan Informatika
Fakultas MIPA
Universitas Sebelas Maret
afrizal.doewes@gmail.com

ABSTRAK

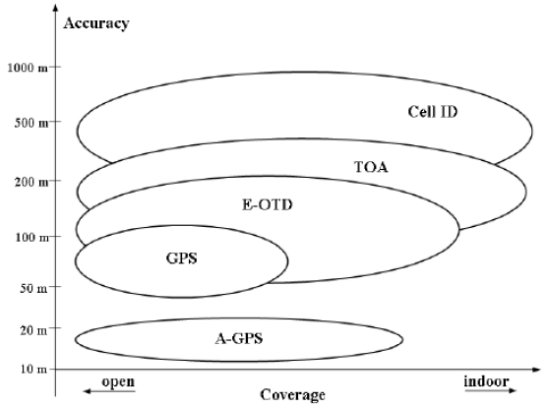
Mobile positioning merupakan suatu teknologi untuk mengetahui posisi perangkat mobile di suatu area geografis. *Mobile Positioning* dapat menggunakan berbagai cara seperti GPS, *Assisted* GPS (A-GPS), jaringan GSM ataupun jaringan Wi-Fi, dan dimanfaatkan untuk berbagai hal, paling umum digunakan sebagai alat navigasi. Pada paper ini akan ditunjukkan manfaat *Mobile Positioning* menggunakan *Cell Id* jaringan GSM / 3G yang dikombinasikan dengan Wi-Fi untuk sistem absensi di Universitas Sebelas Maret. Hasil penelitian menunjukkan *Mobile Positioning* yang menggunakan kombinasi *Cell Id* dan Wi-Fi memiliki kepresisian yang cukup untuk proses validasi bahwa karyawan UNS sudah berada di area kerja.

Kata Kunci: ***Mobile Positioning, Wi-Fi positioning, GSM Cell Id, sistem absensi, webservice***

1. PENDAHULUAN

Mobile Positioning dilakukan dengan beragam cara. Masing-masing cara memiliki kelebihan dan kekurangan. Kelebihan dan kekurangan tersebut bisa diukur dengan beberapa parameter seperti akurasi, coverage, biaya akses dan penyediaan alat. *Mobile Positioning* yang paling populer adalah *Global Positioning System* (GPS). GPS menyediakan akurasi yang tinggi, *coverage* yang luas, di seluruh dunia, akses gratis dan modul GPS yang murah. GPS memiliki kekurangan seperti hanya tersedia di wilayah terbuka, tidak bekerja dengan baik di dalam gedung dan menguras banyak energi [1]. Untuk keperluan navigasi presisi menjadi hal penting, namun pada paper ini, target yang diinginkan *coverage* sempit, kepresisian yang sedang dan sumberdaya energi paling minimal.

Penggunaan teknologi *Mobile Positioning* berdasarkan akurasi dan bisa dilihat di Gambar 1 [2]. Dalam Gambar 1 bisa diketahui kelebihan dari GPS adalah akurasi, dan kekurangannya adalah coverage yang kurang baik di area indoor. Sedangkan kelebihan dari *Cell Id* adalah coverage yang baik sampai dengan indoor building namun jelek di akurasinya. *Assisted* GPS (A-GPS) memiliki akurasi terbaik dan cakupan coverage yang juga bagus, namun tidak bekerja dengan baik di indoor building dan membutuhkan upgrade di *mobile station* (MS) /*handphone*, dan *Base Transceiver Station* (BTS). *Enhanced Observed Time Difference* (E-OTD) dan *Time of Arrival* (TOA) merupakan teknologi positioning yang menggunakan jaringan GSM. Namun kedua teknologi ini hanya bisa diakses oleh operator telepon, sehingga tidak tersedia secara bebas. Terakhir adalah *Cell Id*. *Cell Id* memiliki akurasi rendah namun bisa menjangkau hingga ke dalam gedung. *Cell Id* merupakan identitas *cell* dari sinyal GSM / WCDMA yang bisa diakses secara bebas. MS selalu meng-*update Cell Id* terus-menerus selama *background*.



Gambar 1. Akurasi Vs *Coverage*

Penggunaan *Cell Id* sebagai *Mobile Positioning* jauh lebih hemat energi, yaitu mencapai 90% [1]. Hemat energi ini menjadi pertimbangan utama karena aplikasi yang dibuat, direncanakan untuk selalu menyala selama karyawan bekerja. Kunci hemat energi ini dikarenakan perangkat mobile selalu berinteraksi dengan BTS pada jaringan 2G [3] atau NodeB pada jaringan 3G [4]. Jadi aplikasi yang akan dibuat hanya mengambil data yang sesungguhnya sudah tersedia. Untuk menambah kepresisian maka ditambahkan data *MAC Address* dari Wi-Fi. Namun demikian Wi-Fi ini hanya akan dinyalakan pada saat akan absen masuk. Pada saat Wi-Fi diaktifkan, perangkat mobile juga akan berinteraksi dengan semua *Access Point* (AP) di sekitarnya [5].

Tunjangan Kinerja di Universitas Sebelas Maret (UNS) dibayarkan berdasarkan kehadiran/presensi dan penilaian prestasi kerja karyawan yang diatur dalam Peraturan Rektor No.5931/UN27/KP/2012 yang didasarkan pada Permendikbud No.107 Tahun 2013 yang mengatur tentang jumlah jam kerja yang

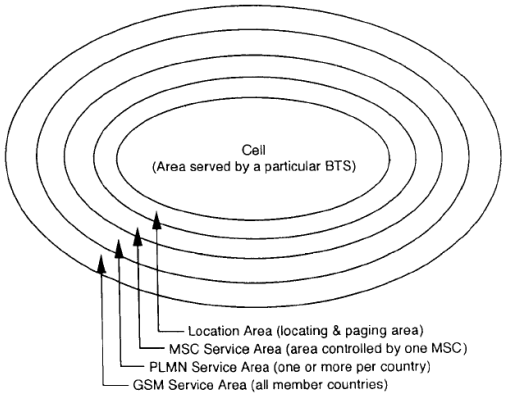
harus dipenuhi setiap karyawan. UNS telah menerapkan sistem absensi menggunakan pemindai sidik jari (*Fingerprint*). UNS membagi unit kerjanya dalam Fakultas dan Biro-biro. Setiap karyawan memiliki hanya bisa melakukan absensi berdasarkan unit kerjanya. Sistem *Fingerprint* sudah sangat baik, namun demikian terdapat beberapa kekurangan. Pertama adalah keterbatasan pemindai *Fingerprint* yang berupa keterbatasan informasi, keterbatasan representasi, dan keterbatasan invariance [6]. Kedua lingkungan kerja yang luas memerlukan infrastrukur pendukung yang kompleks. *Fingerprint* tidak berdiri sendiri melainkan harus terhubung ke jaringan intranet. Kedua, karyawan yang sedang mendapatkan tugas di luar unit kerjanya harus kembali ke unit kerjanya hanya untuk melakukan absensi karena masih terikat dengan sistem absensi di masing-masing unit kerja. Ketiga, bila dilakukan integrasi sistem absensi sidik jari, dengan jumlah karyawan UNS yang mencapai 2548 orang [7], akan menjadikan proses matching semakin lama, *False Match Rate* (FMR) semakin tinggi, yang akhirnya menyebabkan kegagalan absen semakin tinggi dan kenyamanan karyawan berkurang.

Penelitian ini membahas aplikasi sistem absensi dengan penanda *Cell Id* dan *MAC Address* menggunakan perangkat mobile menggunakan metode pengembangan *software prototyping*. Aplikasi yang ditanamkan dalam perangkat mobile ini dinamakan Mobile Attendance System (MobAS). MobAS akan mendeteksi *Cell Id* dan *MAC Address* sebagai validasi posisi karyawan yang sudah berada di area kerjanya. MobAS kemudian akan melakukan koneksi dengan sistem absensi di server untuk mengirimkan data jam masuk dan jam keluar karyawan.

2. TINJAUAN PUSTAKA

2.1 *Cell Id*

Cell merupakan cakupan area terkecil dari sebuah jaringan seluler. Dari Gambar 2 [3] bisa dijelaskan luas cakupan dari jaringan seluler. Dimulai dari GSM, dengan cakupan seluruh dunia, diwakili dengan kode *Mobile Country Code* (MCC), Public Land Mobile Network (PLMN) yang merupakan operator seluler, diwakili oleh kode *Mobile Network Code* (MNC). Dibawah PLMN adalah *Mobile Switching Control* (MSC), *Location Area* dan *Cell* yang diwakili oleh sebuah *Cell Id*. Setiap *Cell Id* mewakili secara spesifik suatu *cell*. Mengidentifikasi suatu lokasi dilakukan dengan mengidentifikasi *Cell Id* dimana perangkat mobile berkomunikasi pada *Cell Id* tersebut. Setiap *Cell* memiliki luas *coverage* berbeda.



Gambar 2. Cakupan area jaringan seluler

Jangkauan *Cell Id* di perkotaan atau urban padat lebih sempit jangkauannya daripada di pedesaan. Hal ini berakibat penentuan lokasi menggunakan *Cell Id* di perkotaan lebih akurat daripada di pedesaan. Luas cakupan *Cell Id* sesuai dengan Gambar 1 berkisar dari 100 meter sampai dengan 1 km untuk BTS di *outdoor* dan puluhan meter untuk BTS *indoor*.

2.2 *MAC Address*

Medium Access Control (MAC) *Address* atau *Physical Address* adalah setara dengan nomor seri Kartu Jaringan (*Network Interface* Card/NIC). Setiap alamat MAC adalah unik yang panjangnya terdiri dari 48 bit. 24 bit pertama dari *MAC Address* merepresentasikan vendor pembuat kartu tersebut dan 24 bit sisanya merepresentasikan nomor kartu tersebut. Pada lapisan OSI (Open System Interconnection) *MAC Address* menempati lapisan Data Link Layer (layer kedua setelah Physical Layer). Akses point Wi-Fi yang merupakan perangkat jaringan memiliki *MAC Address* yang bersifat unik dan permanen.

2.3 *Mobile Positioning*

Cell Id mewakili suatu lokasi geografis Base Transfer Station (BTS) dari jaringan seluler. Jika sebuah perangkat mobile tersambung ke *Cell Id*, maka posisi perangkat mobile itu dapat diperkirakan sejauh jangkauan pancaran dari sebuah *cell*. Sebuah perangkat mobile akan tersambung ke satu *Cell Id* terbaik, dan akan menerima daftar *Cell Id* lainnya yang masih dalam jangkauan, yang disebut sebagai *neighbor cells*. Semakin jauh perangkat mobile dengan lokasi *Cell Id*, semakin kecil kekuatan sinyalnya. Oleh karena itu, penentuan posisi perangkat mobile dapat dilakukan dengan *Cell Id* yang tersambung dan *neighbour*-nya dengan mempertimbangkan kekuatan sinyalnya. Begitu juga dengan *MAC Address* Wi-Fi, *Mobile Positioning* menggunakan Wi-Fi Wi-Fi biasa digunakan untuk melakukan *Mobile Positioning* dalam kondisi di dalam ruangan. Beberapa *access point* yang posisinya sudah terpetakan sebelumnya, akan ditangkap perangkat mobile sehingga dapat ditentukan posisi perangkat mobile.

2.4 Penelitian pendukung

Terdapat beberapa penelitian sebelumnya yang mendukung penelitian ini. Penelitian yang berjudul *Energy-Efficient Positioning for Smartphones using Cell Id Sequence Matching* [1] membahas tentang penentuan posisi smartphone menggunakan *Cell Id* Sequence Matching. Metode yang digunakan adalah *Cell Id* Aided Possitoning System (CAPS). Metode ini didesain berdasarkan kebiasaan rute perjalanan user. Metode ini menghasilkan estimasi posisi yang akurat dengan tingkat penggunaan energi yang kecil. Penggunaan energi lebih hemat lebih dari 90% dari penggunaan GPS sebagai penentu posisi, dan tingkat kesalahan kurang dari 20% dari skema berdasarkan tower *Cell Id*.

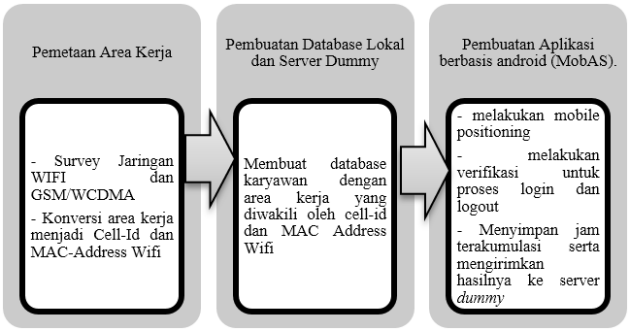
Penelitian kedua berjudul *No Need to War-Drive: Unsupervised Indoor Localization* [8]. Penelitian ini membahas posisi smartphone dalam ruangan (indoor). Penelitian ini menghasilkan sebuah metode yang dinamakan unLoc. UnLoc merupakan gabungan dari tiga ide, yaitu dead-reckoning, urban sensing, dan Wi-Fi-based *partitioning*. UnLoc akan melacak jejak pengguna dalam ruangan. Sebagai contoh melacak jejak pengguna dalam sebuah mall. Ketika pengguna memasuki sebuah mall, aplikasi UnLoc dalam smartphone akan berjalan dan membaca beberapa sensor untuk diolah. Hal-hal yang disimpan untuk diolah antara lain menggunakan accelerometer, kompas, gyroscope, dan akses poin Wi-Fi.

Pada penelitian lain, *Robust Tracking in Cellular Networks Using HMM Filters and Cell Id Measurements* [9], membahas tentang

pelacakan jaringan seluler menggunakan Hidden Markov Model (HMM) Filter yang berdasarkan pada *Cell Id*. Penggunaan HMM digunakan untuk perhitungan yang lebih mudah. Penilaian akurasi didasarkan dengan tingkat akurasi yang tinggi yaitu *Fingerprint* dan pendekatan menggunakan *Cell Id*. Hasil akurasi yang didapatkan dari pendekatan ini melebihi akurasi yang didapatkan dari pendekatan *Cell Id*.

3. METODOLOGI

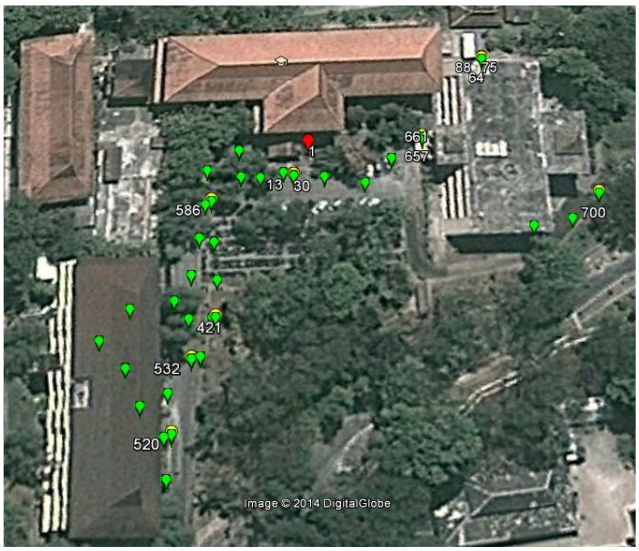
Tujuan dari penelitian ini adalah membuat prototipe aplikasi berbasis Android yang akan berfungsi menjadi input sistem absensi dan memberikan akses informasi jumlah jam kerja kepada karyawan. Untuk mewujudkan tujuan tersebut, dilaksanakan 3 langkah metodologi yang dilaksanakan seperti ditunjukkan dalam Gambar 3.



Gambar 3. Metodologi Pembuatan MobAS

4. PEMBAHASAN

4.1 Pemetaan Area Kerja



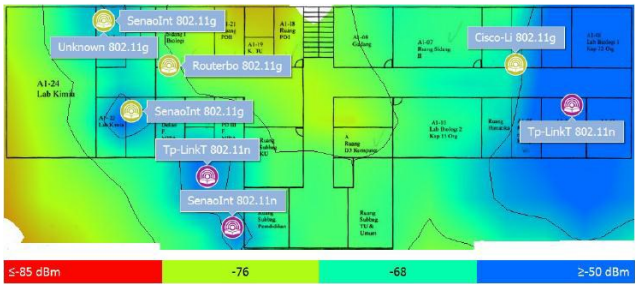
Gambar 4 Hasil survey jaringan 2G/3G di Fakultas MIPA menggunakan MobiLog®

Dalam penelitian ini dibutuhkan data *Cell Id* yang melingkupi area kampus UNS dan Mac Adress Wi-Fi yang berada pada setiap gedung unit kerja. Dalam survey 2G/3G digunakan aplikasi MobiLog®. Aplikasi ini merekap beberapa parameter yang memiliki referensi geografis. Karena memiliki referensi geografis, hasil survey bisa di buka di Google Maps atau Google Earth. Dalam penelitian ini untuk memudahkan pengamatan digunakan aplikasi Google Earth. Pada Gambar 4 ditampilkan hasil survey untuk fakultas MIPA. Dari survey tersebut diambil data yang diperlukan yaitu identitas jaringan atau *Cell Id* seperti ditunjukkan dalam Tabel 1.

Tabel 1. *Cell Id* di lingkungan UNS

No	LAC	CID
1	3521	38329
2	3521	38328
3	3521	38327
4	3521	38326
5	3521	38325
6	3521	38324
7	3521	38322
8	3521	38321
9	3521	29505
10	3521	29502
11	3521	38124
12	3521	38121

Pada tahap ini dilakukan pemetaan kerja dari hasil survey 2G/3G yang telah dilaksanakan. Pada Gambar 5, diperlihatkan contoh hasil survey di Fakultas MIPA. Dari pemetaan Gambar 5, maka diperoleh *MAC Address* seperti yang tersaji dalam Tabel 2.



Gambar 5 Hasil survey Wi-Fi Gedung A Fakultas MIPA

Tabel 2. *MAC Address* beberapa unit kerja

No	UNIT KERJA	*MAC ADDRESS*
1	Perpustakaan	4c:e6:76:43:12:b9
2	Bapsi	d8:5d:4c:eb:8b:18, bc:f6:85:cc:d3:c2
3	Fakultas Ekonomi	d8:5d:4c:a4:86:82, e8:94:f6:5e:64:66
4	Fakultas Hukum	98:fc:11:bf:11:02
5	Fakultas MIPA	d8:5d:4c:ea:31:52, 64:70:02:8a:75:7e

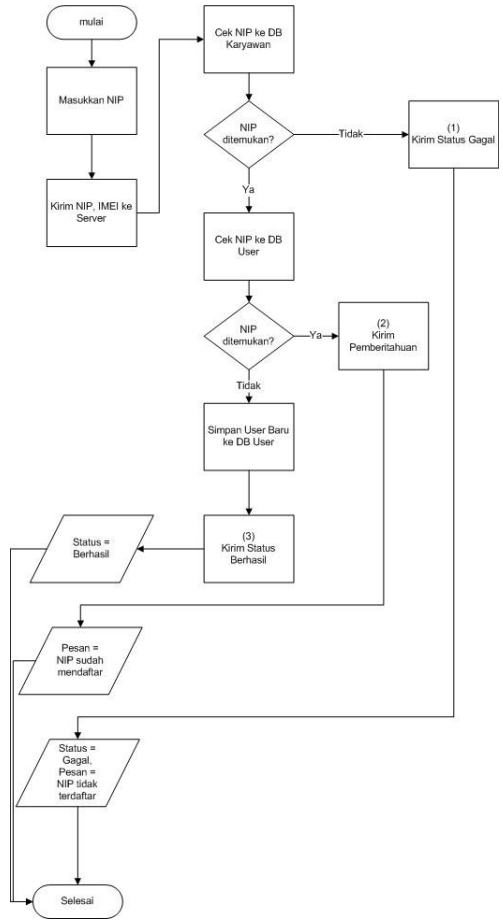
4.2 Aplikasi MobAS

Aplikasi ini memuat tiga proses utama, yaitu registrasi, login, dan logout.

4.2.1 Proses Registrasi

Registrasi digunakan saat aplikasi pertama kali dibuka, yaitu untuk mendaftarkan identitas pemilik ke sistem MobAS. Identitas pemilik ini berupa data diri karyawan dan identitas jaringan yang bersesuaian dengan karyawan tersebut berupa identitas jaringan 2G/3G yaitu *Cell Id* dan jaringan Wi-Fi yaitu *MAC Address*. Identitas jaringan yang tersimpan dalam MobAS ini digunakan

untuk dicocokan dengan identitas jaringan yang didapatkan secara real time dalam proses login dan logout. Cara kerja registrasi ini dapat dilihat dalam flowchart registrasi Gambar 6.



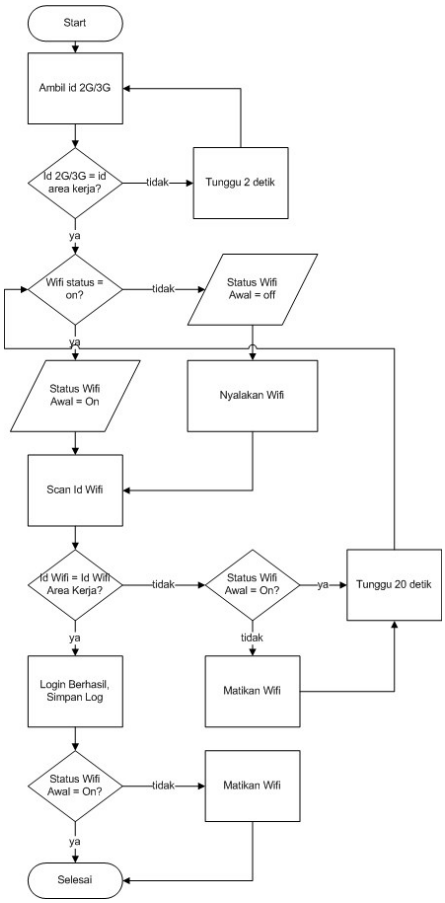
Gambar 6. Flowchart registrasi

4.2.2 Proses Login

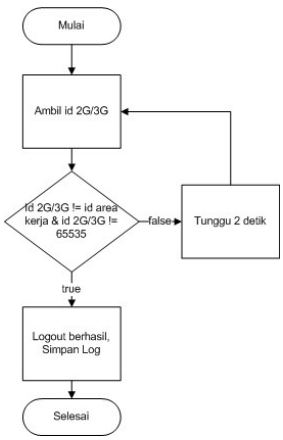
Proses kedua adalah proses login. Proses login adalah proses ketika MobAS berada di dalam kondisi area kerja karyawan. Proses login ini dapat dikatakan juga sebagai absen masuk atau absen pagi. Proses login membutuhkan pencocokan identitas jaringan 2G/3G (*Cell Id*) dan identitas jaringan Wi-Fi (*MAC Address*) antara identitas jaringan yang tersimpan di dalam database MobAS dengan identitas jaringan yang didapatkan secara real time. Jika pencocokan *Cell Id* dan *MAC Address* ini berhasil, maka akan disimpan waktunya dalam database MobAS. Proses login ini dapat digambarkan dengan flowchart seperti pada Gambar 7 proses login.

4.2.3 Proses Logout

Proses ketiga adalah logout. Proses logout adalah proses yang terjadi setelah MobAS dalam kondisi login kemudian keluar dari area kerja karyawan sehingga MobAS tidak mendapatkan *Cell Id* yang cocok. Proses ini dikatakan sebagai absen keluar atau absen sore. Proses logout ini mencocokan *Cell Id* yang didapatkan MobAS dan *Cell Id* yang didaftarkan. Jika pencocokan *Cell Id* ini bernilai *false*, maka proses logout berhasil dan waktu logout ini akan disimpan dalam database MobAS. Proses logout ini dijelaskan dalam *flowchart* seperti pada Gambar 8 proses logout.



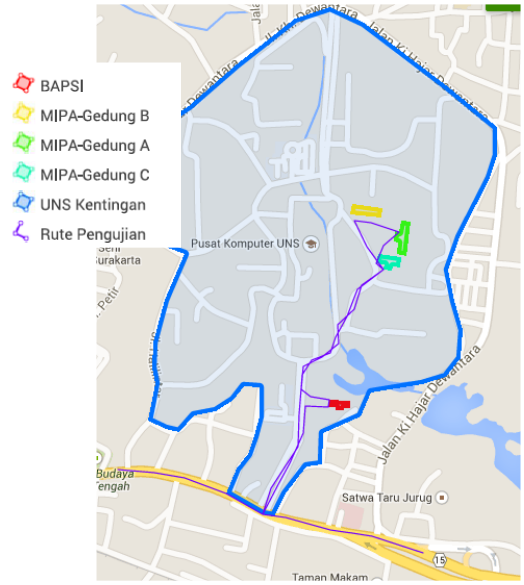
Gambar 7. Proses login



Gambar 8. Proses logout

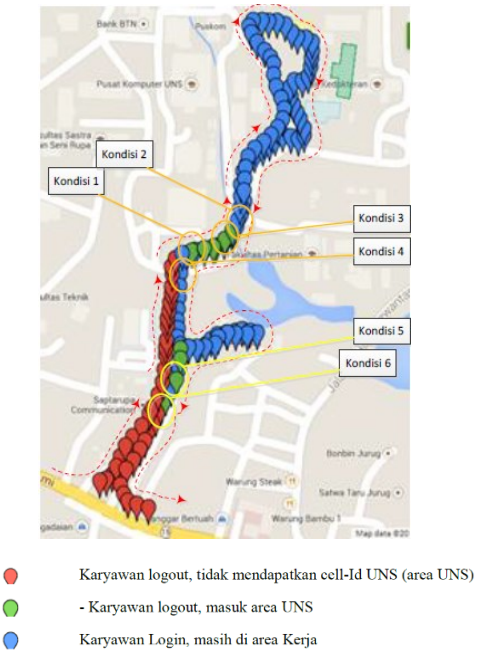
4.3 Hasil Uji Coba

Pengujian MobAS ini bertujuan menguji fungsionalitas dari aplikasi. Apakah sudah memenuhi target awal penelitian yaitu memberikan kemampuan karyawan untuk absen dengan cukup memasuki area kerjanya. Untuk pengujian MobAS, digunakan 2 sampel area kerja, Fakultas MIPA dan Biro BAPSI. Bisa dilihat dalam Gambar 9 yang merupakan posisi dari Gedung F.MIPA dan BAPSI dalam area kerja UNS.



Gambar 9 Posisi Gedung MIPA dan BAPSI

Uji coba ini menggunakan dua user, dengan ketentuan seperti dalam Tabel 3. Uji coba ini dilakukan dengan melakukan pengecekan kondisi login atau logout di lokasi-lokasi tertentu, seperti pada Tabel 4.



Gambar 10. Hasil pengujian

Dengan rute pengujian seperti yang direncanakan, hasil pengujian bisa dilihat di Gambar 10.

Tabel 3 User dan ketentuan unit kerjanya

No	Nama	Unit Kerja
1	User 1	BAPSI, FMIPA
2	User 2	Perpustakaan

Tabel 4 Hasil Pengujian

Lokasi	User 1	User 2
Berada di luar area kampus	Logout	Logout
Berada di area kantor BAPSI	Login	Logout
Berada di area Perpustakaan	Login	Login
Berada di luar area kampus	Logout	Logout

4.4 Akurasi *Cell Id*

Cell Id digunakan sebagai penanda bahwa karyawan sedang berada di lingkungan UNS. *Cell Id* yang bersesuaian dengan lingkup UNS telah dijelaskan pada poin 4.1 Pemetaan Area Kerja. Percobaan ini dilakukan dengan menghitung jarak antara posisi MobAS saat mendapatkan *Cell Id* (saat masuk lingkungan UNS) dengan gerbang dan jarak antara posisi MobAS saat melepaskan *Cell Id* (saat keluar lingkungan UNS) dengan gerbang. Jarak inilah yang dimaksud dengan ketepatan *Cell Id*, dimana nilai yang lebih kecil berarti lebih baik karena berarti semakin mendekati *border* lingkungan UNS. Hasil percobaan ini dapat dilihat dalam Tabel 5.

Tabel 5 Akurasi *Cell Id* dan gerbang kampus

No	Gerbang Depan		Gerbang Belakang	
	Masuk (meter)	Keluar (meter)	Masuk (meter)	Keluar (meter)
1	347	110	215	129
2	187	97.6	341	125
3	48.8	60.1	208	142
4	245	13.4	112	401
5	260	96.4	207	102
Rata-rata	217.56	75.5	216.6	179.8
Nilai tertinggi	347	110	341	401
Nilai terendah	48.8	13.4	112	102
Akurasi	14.25		198.2	

Dari Tabel 4 akurasi *Cell Id* di atas, dapat disimpulkan adalah akurasi dari gerbang depan saat kondisi masuk dan keluar adalah 146.25 meter. Sedangkan untuk gerbang belakang adalah 198.2 meter.

4.5 Akurasi Wi-Fi

Wi-Fi digunakan sebagai penanda bahwa MobAS sedang dalam area lingkungan kerja atau dengan kata lain, MobAS berhasil login. Percobaan ini mengukur akurasi jarak posisi MobAS berhasil login dengan batas lokasi area kerja.

Tabel 6. Akurasi Wi-Fi dengan area kerja

No	Perpus (meter)	Mipa (meter)	Bapsi (meter)
1	21.3	28.4	6.5
2	25.6	36.9	16.9
3	57.8	32	27.6
Rata-rata akurasi	34.9	32.4	17

Percobaan ini dilakukan di tiga unit kerja, yaitu Perpustakaan, Fakultas MIPA, dan BAPSI. Hasil dari percobaan akurasi Wi-Fi ini adalah rata-rata 34,9 meter untuk area kerja perpustakaan, 32,4 meter untuk area kerja Fakultas MIPA, dan 17 meter untuk area kerja BAPSI. Hasilnya dapat dilihat dalam Tabel 6.

5. KESIMPULAN DAN SARAN

5.1 Kesimpulan

Kesimpulan dari penelitian ini adalah:

1. Penelitian ini telah menghasilkan sistem absensi mobile dengan penanda jaringan *Cell Id* dan *MAC Address*.
2. Fungsi pendeteksi posisi dari aplikasi telah berjalan dengan baik, hasilnya sesuai dengan id jaringan yang dideteksi.
3. Tingkat akurasi sesuai dengan cakupan area dari *cell* BTS / NodeB dan cakupan dari akses point Wi-Fi. Wi-Fi yang cakupan pancaran lebih sempit maka menghasilkan akurasi penentuan posisi yang lebih presisi.
4. Rata-rata akurasi *Cell Id* untuk gerbang depan adalah 146.25 meter dan untuk gerbang belakang adlah 198.2 meter.
5. Rata-rata akurasi Wi-Fi dengan area kerja Perpustakaan pusat adalah 34.9 meter, Fakultas MIPA 32.4 meter, dan BAPSI 17 meter.

5.2 Saran

Saran untuk pengembangan penelitian selanjutnya adalah:

1. Melakukan penelitian tentang Wi-Fi dan kekuatan sinyalnya di seiap masing-masing area unit kerja untuk meningkatkan tingkat kepresisian saat login.
2. Melakukan penelitian *Cell Id* dan kekuatan sinyalnya untuk menentukan border di lingkungan UNS saat logout.
3. Penanganan perpindahan *Cell Id* yang tidak terdaftar di database, saat kondisi sudah login dan masih berada di lingkungan UNS.

6. DAFTAR PUSTAKA

[1] Jeongyeup Paek, Kyu-Han Kim, Jatinder P. Singh, and Ramesh Govindan, 2011, Energy-efficient positioning for smartphones using *Cell Id* sequence matching. In Proceedings of the 9th international conference on Mobile systems, applications, and services (MobiSys '11), ACM, New York, NY, USA.

[2] Trevisani, E.; Vitaletti, A., 2004, *Cell Id* location technique, limits and benefits: an experimental study, Mobile Computing Systems and Applications, WMCSA, Sixth IEEE Workshop on , vol., no., pp.51,60

[3] Asha Mehrotra, 1997, GSM System Engineering, Artech House Inc, London

[4] Christophe Chevallier, Christopher Brunner, Andre Garavaglia, Kevin P. Murray, Kenneth R. Baker, 2006, WCDMA (UMTS) Deployment Handbook John Wiley & Sons Ltd, West Sussex PO19 8SQ, England

[5] IEEE Computer Society, 2012, Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, New York

[6] [Davide Maltoni, Dario Maio, Anil K. Jain, and Salil Prabhakar, 2009, Handbook of *Fingerprint* Recognition (2nd ed.), Springer Publishing Company, Inc, New Jersey

[7] Jamal Wiwoho, 2013, Presentasi Koordinasi Tim Penataan Pegawai UNS,http://jamalwiwoho.com/wp/content/uploads/2013/05/Koordinasi-Tindaklanjut-Penataan-Pegawai.pdf, diakses tanggal 24 Februari 2013

[8] He Wang, Souvik Sen, Ahmed Elgohary, Moustafa Farid, Moustafa Youssef, and Romit Roy Choudhury, 2012, No need to war-drive: unsupervised indoor localization, In Proceedings of the 10th international conference on Mobile systems, applications, and services (MobiSys '12). ACM, New York, NY, USA, 197-210

[9] [9] Bshara, M.; Orguner, U.; Gustafsson, F.; Van Biesen, L., 2011, Robust Tracking in Cellular Networks Using HMM Filters and *Cell Id* Measurements, Vehicular Technology, IEEE Transactions on , vol.60, no.3, pp.1016,1024.

Optimasi Produksi Menggunakan Algoritma Fuzzy Linear Programming (Studi Kasus : Produksi Tas UKM Cantik Souvenir)

Lia Primadani
Jurusan Informatika
Universitas Sebelas Maret
Jl. Ir. Sutami No.36A, Jebres, Surakarta
primdanilia@student.uns.ac.id

YS.Palgunadi
Jurusan Informatika
Universitas Sebelas Maret
Jl. Ir. Sutami No.36A, Jebres, Surakarta
palgunadi@uns.ac.id

Bambang Harjito
Jurusan Informatika
Universitas Sebelas Maret
Jl. Ir. Sutami No.36A, Jebres, Surakarta
bambang.harjito@staff.uns.ac.id

ABSTRAK

Effisiensi produksi dan pengoptimalan kinerja dalam suatu perusahanan dapat dicapai dengan menggunakan model optimasi. Model optimasi dapat dituliskan dalam sebuah fungsi persamaan dan pertidaksamaan yang dikenal dengan nama Program Linier. Program Linier dapat dikombinasikan dengan nilai *fuzzy* untuk menyesuaikan model permasalahan produksi yang sangat bergantung pada permintaan pasar yang berubah-ubah. Pemodelan ini dikenal dengan nama *Fuzzy Linear Programming*.

Menggunakan *Fuzzy Linear Programming*, permasalahan produksi UKM Cantik Souvenir dianalisis dan dipecahkan dengan bantuan program simulasi. Penyelesaian *Fuzzy Linear Programming* menggunakan metode Simpleks *fuzzy*. Penyelesain dengan metode Simpleks *fuzzy* terdiri dari beberapa langkah yaitu menghitung batas bawah dan batas atas optimal dengan metode simpleks maksimasi, memodifikasi persamaan awal dengan menambahkan variabel λ, dan menyelesaikannya dengan metode simpleks dua fase. Hasil penyelesaian dengan program simulasi menunjukkan jumlah perkiraan profit menggunakan persamaan Program Linier Rp 8.740.375,00 dan menggunakan persamaan *Fuzzy Linear Programming* sebesar Rp 9.510.003, 00 dengan nilai fungsi keanggotaan *fuzzy* sebesar 0,5.

Keywords

Model Matematika,Fuzzy Linear Programming, Jadwal Produksi Induk, Metode Simpleks Fuzzy Linear Programming.

1. PENDAHULUAN

Optimasi adalah sarana untuk mengekpresikan model yang bertujuan untuk memecahkan masalah dengan cara terbaik [1]. Model optimasi yang ada digunakan untuk menyelesaikan berbagai permasalahan dalam pemerintahan, bisnis, teknik ekonomi, ilmu-ilmu fisika dan sosial yang terkait dengan adanya keterbatasan pengalokasian sumber daya [2]. Salah satu contoh pemanfaatan analisa optimasi dalam bisnis adalah untuk melakukan penentuan jumlah produksi paling optimal dengan persedian bahan baku yang terbatas. Pengoptimalan dapat dilakukan dengan menggunakan Jadwal Produksi Induk dalam model matematis [3],

Penjadwalan induk Produksi dimanfaat untuk menganalisa permasalahan atau kendala yang dihadapi selama proses produksi berlangsung. Kendala proses produksi diantaranya adalah kapasitas produksi, jam kerja, dan jumlah permintaan yang bersifat tidak stabil [4]. Permasalahan seperti ini sering kita jumpai pada UKM (Usaha Kecil Menengah) yang masih mengadopsi sistem pengerjaan tradisional dimana semua proses dikerjakan secara manual mengandalkan tenaga manusia sehingga hasil atau *ouput* produksi terkadang tidak mampu menyesuaikan permintaan pasar. Oleh sebab itu diperlukan satu pengendalian dan perencaanaan produksi yang dapat mengendalikan *input* dan *output* produksi sesuai dengan permintaan pasar [3].

Perencanaan dapat dilakukan dengan membuat sebuah model yang merepresentasikan setiap permasalahan yang ada dengan tujuan untuk memudahkan dalam proses analisa yaitu dengan menggunakan model matematis [5]. Permasalahan diubah dalam model optimasi berupa persamaan linier yang lebih dikenal dengan permasamaan Program Linier.

Pada penelitian sebelumnya [6], Program Linier dikombinasikan dengan fungsi keanggotaan *fuzzy trapezoidal* dan diselesaikan menggunakan bantuan metode simpleks serta fungsi ranking. Fungsi ranking tersebut digunakan untuk mengurutkan bilangan *fuzzy* sehingga diperoleh bilangan real dan dapat dibandingkan, kemudian diterapkan dalam penentuan jumlah produksi dua jenis obat yaitu fluon dan fluin. Pemanfaatan fungsi keanggotaan *trapezoidal* yang sama pada penelitian sebelumnya dibandingkan dengan penggunaan fungsi keangootaan dengan kurva S termodifikasi untuk melakukan analisa penilian kinerja karyawan [7] merujuk pada penelitian [8] diperoleh hasil penilaian kinerja dengan penggunaan fungsi keangggotaan *trapezoidal* pada Program Linier lebih baik dibandingkan dengan fungsi kenggotaan kurva S. Namun dalam penelitian-penelitian yang ada, belum ditemui bentuk pengubahan persamaan Program Linier ke dalam bentuk persamaan *Fuzzy Linear Programming* serta hasil perhitungan menggunakan metode Simpleks *Fuzzy Linear Programming*. Oleh Sebab itu dalam penelitian ini, akan dibahas tentang pembentukan model persamaan *Fuzzy Linear Programming* dengan contoh studi kasus produksi tas di UKM Cantik Souvenir dan pencarian hasil penyelesaiaan menggunakan sebuah program simulasi yang dibangun dengan bahasa pemrograman PHP.

2. PENELITIAN TERKAIT

Algoritma *Fuzzy Linear Programming* sebelumnya pernah digunakan pada beberapa penelitian seperti yang dijelaskan pada tabel 1.

Tabel 1. Penelitian Terkait

No	Penulis	Tujuan	Metode	Kelebihan	Kelemahan
1	Purba Rivelson, 2012 [1]	Mengetahui peranan fuzzy dalam menyelesaikan program linier *fuzzy* dan membandingkan program linier *fuzzy* dan program linier biasa	Simpleks dua fase	Dapat mengetahui model *defuzzifikasi* pada permasalahan *Fuzzy Linear Programming*	Variabel keputusan dan batasan tidak relevan.
2	Iveline Anne Marei, Eriyanto, Yandra Arkeman, & Dadan Umar Diahani, 2011 [3]	Pemanfaatan Jadwal Produksi Induk untuk Perencanaan produksi	*Fuzzy Linier Programming* dengan fungsi keanggotaan *fuzzy kurva–S* yang dimodifikasi	penggunaan dua fungsi tujuan sekaligus dalam satu kasus, dapat membuat jadwal produksi induk dalam model matematis *Fuzzy Linear Programming*	- Model persamaan *Fuzzy Linear Programming* tidak ada - Perolehan nilai *fuzzy* tidak ada
3	Effati S. & Abbasiyan H., 2009 [5]	Modifikasi metode *subgradient,* penentuan daerah layak, memodifikasi metode untuk penentuan daerah yang layak dengan memperkenalkan metode baru yaitu fuzzy decisive set method	Fuzzy Linier Programming, Augmented Lagrangian penalty method, fuzzy decisive set method	Dapat mengetahui model Fuzzy pada nilai koefisien ruas kanan	Lebih berfokus pada menanikkan nilai fungsi keanggotaan fuzzy bukan pada pencarian hasil optimal variabel tujuan.
4	Astuti Irma Suryani, Lilik Linawati, Hanna A. Parhusip ,2014 [8]	Penggunaan *Fuzzy Linear Programming* dengan fungsi keanggotaan kurva-S pada Penilaian Kinerja Karyawan	FLP dengan fungsi kenggotaan *fuzzy* kurva-S termodifikasi	Berfokus pada perbanding penggunaan fungsi keanggotaan *fuzzy* pada PL, mana yang feasible ketika ditambahkan pada PL	penggunaan fungsi kenggotaan *fuzzy* kurva S tidak cocok untuk menyelesaikan kasus penilaiaan seleksi tenaga kerja
5	Barkha Sharma & Rajendra Dubey, 2012 [9]	Menjelaskan tentang penyelesaain permasalahan program linier dalam bentuk *fully fuzzy* menggunakan simulasi perhitungan sederhana	Fungsi *ranking*, metode simpleks	Fungsi *ranking* digunakan untuk menentukan urutan dari bilangan *fuzzy,* metode simpleks Big M, adanya penjelasan tentang cara penyelesaian masalah *fully fuzzy* dengan metode big M	Penyelesaian dengan elimanasi *gauss* tidak cocok untuk menyelesaiakan kasus dengan jumlah batasan dan variabel tujuan dalam jumlah banyak

Berdasarkan penelitian terkait pada tabel 1, dalam penelitian ini akan membahas bagaimana membuat sebuah model optimasi produksi tas pada UKM Cantik Souvenir dengan mengimplementasikan Algoritma *Fuzzy Linear Programming* dalam sebuah program simulasi yang dibangun menggunakan bahasa pemrograman PHP.

3. *Fuzzy Linear Programming* pada Produksi Induk

Model matematika *Fuzzy Linear Programming* pada Produksi Induk merupakan sebuah metode yang digunakan untuk memodelkan hasil analisa Jadwal Induk Produksi dalam sebuah fungsi pertidaksamaan dan persamaan Program Linier dengan variabel *fuzzy*.

Hasil analisa dari Jadwal Produksi Induk berupa perencanaan jumlah produksi dari analisis faktor produksi, diantaranya adalah jumlah permintaan, *status inventory*, pengalokasi sumber daya, pembelian yang dikeluarkan, dan ketersediaan *stock*. Kemudian dari hasil analisa jadwal produksi induk tersebut dilakukan pembentukan model ke dalam sebuah persamaan dan pertidaksamaan matematika dengan menggabungkan konsep logika *fuzzy* pada permasalahan Program Linier yang dikenal dengan *Fuzzy Linear Programming*.

Fuzzy Linear Programming merupakan penggambungan sebuah konsep logika *fuzzy* dalam sebuah permasalahan Program Linear. Konsep *fuzzy* dalam Program Linear pertama kali dikemukakan oleh Tanaka et al, dalam sebuah kerangka pengambilan keputusan *fuzzy* oleh Bellman dan Zadeh. Sedangkan formulasi pertama dari FLP dikemukakan oleh Zimmermann [9].

Berikut adalah bentuk matematika dari Kasus Program Linier yang telah diberikan nilai *fuzzy* pada bagian sumber daya atau koefisien ruas kanan dan ruas kiri fungsi tujuan:

Fungsi tujuan $\tilde{Z} = maximize \sum_{j=1}^{n} C_j x_j$

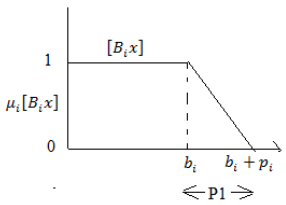
Kendala/batasan: $\sum_{j=1}^{n} a_{ij} x_j \leq \tilde{b}_i, 1 \leq i \leq m$

$x_j \geq 0, 1 \leq j \leq n$ ..... (1)

Dimana x_j adalah vektor variabel keputusan, a_{ij} adalah penggunaan sumberdaya masing-masing item, dan $\tilde{b}_i$ adalah nilai *fuzzy* (mewakili jumlah sumber daya yang tersedia) dengan fungsi keanggotaan linier sebagai berikut:

$$\mu_{\tilde{b}_i} \begin{cases} 1, x < b_i \\ \frac{b_i + p_i - x}{p_i}, b_i \leq x \leq b_i + p_i, \\ 0, x \geq b_i + p_i, \end{cases} \quad \text{..... (2)}$$

Jika digambarkan dalam sebuah grafik membentuk gambar grafik seperti gambar 1.



Gambar 1. Fungsi Kenggotaan *Fuzzy Trapezoidal* [1]

Dimana p_i menyatakan jumlah penambahan sumber daya yang diberikan untuk menaikkan nilai profit produksi [5] dengan jumlah sesedikit mungkin. Nilai $p_i > 0$ untuk $i = 1, \dots, m$ dan dapat diperoleh dengan menggunakan menghitung simpangan atau rentang perubahan nilai dalam penggunaan bahan (sumber daya) dalam satu kali produksi sebagai berikut:

$$p_i = t_i \times b_i \quad \text{..... (3)}$$

Dimana b_i merupakan sumber daya dan t_i adalah prosentase perubahan penggunaan bahan dalam satu kali produksi.

$$t_i = \frac{\sqrt{\frac{\Sigma(x_i - \bar{x})^2}{n-1}}}{\bar{x}} \times 100\% \quad \text{..... (4)}$$

X_i= data sampel indeks ke – i , i =1,2,3,...,n

$\overline{X}$ = nilai rata – rata untuk seluruh data sampel. Nilai rata-rata diperoleh dengan $\frac{\sum x_i}{n}$

$n =$ jumlah data sampel yang diambil.

Kemudian untuk mencari penyelesaiaan optimal diperlukan dua tahapan, yaitu menghitung batas bawah nilai optimal (Z_l) dan batas atas nilai atas optimal (Z_u). Nilai optimal Z_l dan Z_u dapat dicari menggunakan penyelesaian program linier dengan mengasumsikan permasalahan yang ada sebagai berikut:

$$Z_l = Maximize \sum_{j=1}^{n} C_j x_j$$

$$\text{Kendala: } \sum_{j=1}^{n} a_{ij} x_j \le b_i, 1 \le i \le m \quad \text{..... (5)}$$
$$x_j \ge 0, 1 \le j \le n$$

dan

$$Z_u = Maximize \sum_{j=1}^{n} C_j x_j$$

$$\text{Kendala: } \sum_{j=1}^{n} a_{ij} x_j \le b_i + p_i, 1 \le i \le m \quad \text{..... (6)}$$
$$x_j \ge 0, 1 \le j \le n$$

Nilai dari fungsi tujuan berada diantara Z_l dan Z_u yang mana nilai dari koefisien ruas kanan berada diantara b_i dan $b_i + p_i$. Kemudian nilai dari keanggotaan *fuzzy* optimal adalah G, yang merupakan subset dari R^n yang didefinisikan sebagai berikut:

$$\mu_G(x) \begin{cases} 0, & \sum_{j=1}^{n} c_j x_j < Z_l \\ \frac{\sum_{j=1}^{n} c_j x_j}{Z_u - Z_l}, & Z_l \le \sum_{j=1}^{n} c_j x_j < Z_u \\ 1, & \sum_{j=1}^{n} c_j x_j \ge Z_u \end{cases} \quad \text{..... (7)}$$

Kemudian untuk *fuzzy set* i pada batasan c_i, yang merupakan subset dari R^n yang didefinisikan sebagai berikut :

$$\mu_{c_i}(x) \begin{cases} 0, & b_i < \sum_{j=1}^{n} a_{ij} x_j \\ \frac{b_i - \sum_{j=1}^{n} a_{ij} x_j}{p_i}, & \sum_{j=1}^{n} a_{ij} x_j \le b_i < \sum_{j=1}^{n} a_{ij} x_j + p_i \\ 1, & b_i \ge \sum_{j=1}^{n} a_{ij} x_j + p_i \end{cases} \quad \text{..... (8)}$$

Fungsi tujuan *Fuzzy* [6] didefinisikan sebagai berikut :

$$\mu_D(x) = \min\left(\mu_G(x), min_i(\mu_{c_i}(x))\right) \quad \text{..... (9)}$$

Persamaan tersebut dapat diubah kebentuk penyelesaian untuk kasus optimasi sebagai berikut:

$$max_{x \ge 0} \mu_D(x) = max_{x \ge 0} \min\left(\mu_G(x), min_i(\mu_{c_i}(x))\right) \quad \text{..... (10)}$$

Sehingga persamaan 1 dapat diubah kebentuk penyelesaian optimal sebagai berikut:

Fungsi Tujuan : Maximize λ

Batasan: $\mu_G(x) \ge \lambda$

$\mu_{c_i}(x) \ge \lambda, 1 \le i \le m$

$x \ge 0$

$0 \le \lambda \le 1$ ..... (11)

Persamaan 7, 8 dan 11 dapat didefinisikan sebagai berikut:

$$\lambda(Z_u - Z_l) - \sum_{j=1}^{n} C_j x_j + Z_l \le 0$$

$$\sum_{j=1}^{n} a_{ij} x_j + \lambda p_i \le b_i + p_i, \text{dimana } 1 \le i \le m$$

$$x \ge 0, 0 \le \lambda \le 1 \quad \text{..... (12)}$$

4. Metode Simpleks untuk Fuzzy Linear Programming

Metode Simpleks *Fuzzy Linear Programming* merupakan metode gabungan simpleks dan Simpleks dua fase untuk menyelesaikan permasalahan *Fuzzy Linear Programming* menggunakan sebuah tabel yang dikenal dengan *tablo simplex*. Bentuk tabel ekuivalen dengan bentuk persamaan aljabar, sehingga memudahkan dalam proses perhitungan atau penyelesaian dalam bentuk komputasi [2] yang dijelaskan pada tabel 2.

Tabel 2. Tablo Simpleks

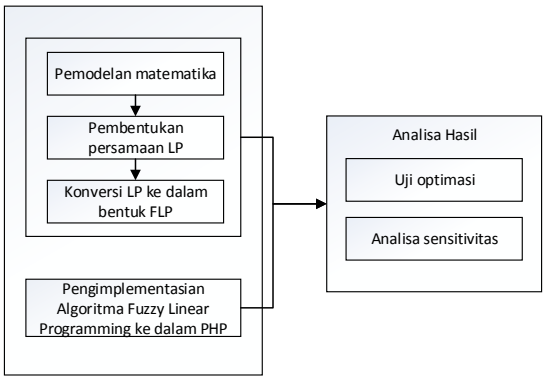
Variabel Dasar	Koefisien Dari							RHS
	Z	x_1	...	x_n	S_1	...	S_n	

Adapun alur penyelesaian permasalahan *Fuzzy Linear Programming* adalah sebagai berikut [11]:

1. *Fuzzifikasi* , mengubah persamaan PL ke bentuk FLP
2. *Defuzzifikasi* , mencari nilai batas atas optimal dan batas bawah optimal. Menggunakan dua metode yaitu simpleks dan simpleks dua fase.

5. METODE PENELITIAN

Penelitian ini diselesaikan dengan beberapa tahapan seperti yang dijelaskan diagram alur pada gambar 2.



Gambar 2. Metodologi Penelitian

5.1 Model Matematika dan Parameternya

Model matematika dari persamaan PL ditunjukkan pada persamaan 13.

Fungsi tujuan : $Z = Maximize \sum_{j=1}^{n} C_j x_j$

Kendala: $\sum_{j=1}^{n} a_{ij}\, x_j \leq b_i,\ 1 \leq i \leq m$ ..... (13)

$x_j \geq 0, 1 \leq j \leq n$

Model persamaan 13 disusun dengan menggunakan beberapa parameter input, diantaranya adalah:

i. Variabel tujuan (Obj)

Dilambangkan dengan x_i seperti yang ada pada tabel 3.

Tabel 3. Variabel Tujuan

Obj	Keterangan
x_1	jumlah produksi optimum tas dengan kode ELL
x_2	jumlah produksi optimum tas dengan kode DO
x_3	jumlah produksi optimum tas dengan kode DAM
x_4	jumlah produksi optimum tas dengan kode DD
x_5	jumlah produksi optimum tas dengan kode DS
x_6	jumlah produksi optimum tas dengan kode RJ1
x_7	jumlah produksi optimum tas dengan kode RJ2
x_8	jumlah produksi optimum tas dengan kode AW

ii. Tujuan, masksimasi profit produksi.

iii. Penentuan fungsi batasan, dijabarkan pada tabel 4 dan 5

Tabel 4. Fungsi Batasan

Batasan	Nomer	Keterangan
Lama Waktu Produksi	1	Penggambaran dan pemotongan pola
	2	Penyablonan
	3	Penjahitan
	4	Pemasangan kancing knop
	5	Pemasangan tali dan stopper
	6	Pengguntingan pinggir
	7	Perapian benang dan pengepakan
Kapasitas produksi per-bulan	8	Jumlah produksi maksimum ELL
	9	Jumlah produksi maksimum DO
	10	Jumlah produksi maksimum DAM
	11	Jumlah produksi maksimum DD
	12	Jumlah produksi maksimum DS
	13	Jumlah produksi maksimum RJ1
	14	Jumlah produksi maksimum RJ2
	15	Jumlah produksi maksimum AW
Penggunaan Bahan	16	Penggunaan kain spunbond polos
	17	Penggunaan kain spunbond motif
	18	Penggunaan kain D600
	19	Penggunaan perekat atau kaitan
	20	Penggunaan Puring
	21	Penggunaan tali
	22	Penggunaan plastik transparan
	23	Penggunaan kancing stopper
	24	Penggunaan kancing knop

5.2 Pengubahan Bentuk Persamaan PL ke FLP

Pengubahan bentuk persamaan PL ke bentuk FLP dilakukan dengan manambahkan fungsi keanggotaan fuzzy. Sehingga diperoleh untuk ruas kanan mendekati/lebih dari/sama dengan b_i seperti yang ditunjukkan pada persamaan 1.

5.3 Pengimplementasian Algoritma *Fuzzy Linear Programming*

Pengimplementasian Algoritma *Fuzzy Linear Programming* menggunakan bahasa pemrograman PHP yang dijelaskan pada *pseudocode* gambar 3, 4, dan 5.

5.3.1 *Metode simpleks*

Pseudocode metode simplek dijelaskan pada gambar 3.

```
input : ruas_kiri, jml_var=8, jml_cons=24
output : z,x1,x2,x3,x4,x5,x6,x7,x8
proses :
1.  $var← 8
2.  $cons←24
3.  initial array $ruas_kiri
4.  $ruas_kiri[0][0]←1
5.  for m=1 to m<=8
    5.1    $ruas_kiri[$m][0]←0
    5.2    endfor
6.  for m=1 to m=24
    6.1    $n=$m+8;
    6.2    $ruas_kiri[0][$n]←0
    6.3    endfor
7.  for 1 to $var
    7.1   $ruas_kiri[0][$m]←$profit[$m]*(-1)
    7.2   endfor
8.  function simplex_maksimasi($ruas_kiri,$var,
    $cons)
9.  for 1 to $var
    11.1   $sort=select minimum value of
           $ruas_kiri[0][$m]
    11.2   endfor
    11.3   find column of $sort and set value as
           $column_pivot
10. for 0 to 24
    12.1   if $column_pivot[$l]<=0
    12.2   set value $rasio[$l] = 'error'
    12.3   else
    12.4   $rasio[$l]←$ruas_kiri[$l]['nk']/$colu
           mn_pivot[$l]
    12.5   endif
    12.6   endfor
11. for 0 to $cons
    13.1   $rasio=select min value of $rasio[$l]
    13.2   endfor
    13.3   find row of $rasio and set value as
           $row_pivot
12. $cek←$ruas_kiri[0][$column]
13. while ($cek<0)do
    15.1   $temp_ruas_kir←$ruas_kiri
    15.2   for 1 to $cons
    15.3   if ($temp_ruas_kiri[$row]=$row_pivot)
    15.4   update $row=$row_pivot/$column_pivot
    15.5   else
    15.6   update $row = value of $row[$l]-
           ($temp_ruas_kiri[$column]*temp_ruas_ki
           ri[$row]
    15.7   endif
    15.8   endfor
14.  back to step 12
```

Gambar 3. *Pseudocode Program* metode Simpleks (lanjutan)

Berikut langkah-langkah untuk mencari nilai batas atas dan batas bawah dengan metode simpleks yang dituliskan dalam bentuk *code* gambar 3:

a. Membuat matriks ruas kiri dengan dimensi *cons*var*

b. Menentukan variabel input, ditentukan dengan mengecek nilai matriks ruas kiri baris pada baris 1. Pilih kolom dengan nilai matriks ruas kiri[0][1] sampai ruas kiri[0][8] dengan nilai paling kecil (negatif terbesar) selanjutnya disebut dengan *column pivot*

c. Menghitung nilai rasio yaitu dengan membagi nilai ruas kiri[1][8] sampai ruas kiri [8][8] dengan *column pivot* . Rasio

dengan nilai positif terkecil selanjutnya tandai baris dengan rasio terkecil sebagai *row_pivot*. Selanjutnya menentukan *key*. *Key* diambil dari nilai irisan matriks *column pivot* dengan *row pivot.*

d. Memperbarui nilai *row*, jika *row=row pivot*, *update row* dengan *value of row/value of key*. Selain itu *update row* dengan nilai *row* dari kolom dikurangi (nilai *row* lama*angka *column pivot* di baris yang sama).
e. Lakukan hal yang sama sampai dengan nilai koeffisien ruas kiri[0][1] sampai ruas kiri[0][8] lebih dari atau sama dengan 0

5.3.2 Metode Simplek Dua Fase

Metode ini dibagi dalam dua fase, yaitu fase minimasi dan maksimasi. Berikut algoritma untuk perhitungan minimasi dijelaskan pada gambar 4.

```
Input:ruas_kiri,   x1,x2,x3,x4,x5,x6,x7,x8,var=8,
cons=24
Output :lamda
Proses :
##fase 1 => minimasi
1.  function simplex_minimasi($ruas_kiri, $var,
    $cons)
2.  $two_var=8+1=9
3.  $two_cons=24+1=25
4.  for 1 to 9
5.  $sort=  select  max  value  of  array
    $ruas_kiri[0][$m]
6.  endfor
7.  Find column of $sort and set value as as
    column_pivot
8.  for 0 to $cons do
     8.1  if $column_pivot[$l]<=0
     8.2  set value $rasio[$l] = 'error'
     8.3  else
     8.4  $rasio[$l]←$ruas_kiri[$l]['nk']/$colum
          n_pivot[$l]
     8.5  endif
     8.6  endfor
9.  for 0 to $cons
     9.1  $rasio=select min value of $rasio[$l]
     9.2  endfor
     9.3  Find row of $rasio and set valus as
          $row_pivot
10. $cek←$ruas_kiri[0][$column]
11. while ($cek>0)do
     11.1 $temp_ruas_kir←$ruas_kiri
     11.2 for 1 to $cons
     11.3 if ($temp_ruas_kiri[$row]=$row_pivot)
     11.4 update $row=$row_pivot/$column_pivot
     11.5 else
     11.6 update  $row  =  value  of  $row[$l]-
          ($temp_ruas_kiri[$column]*temp_ruas_kir
          i[$row]
     11.7 endif
     11.8 endfor
12. back to step 4
## fase 2 => maksimasi
13. call  function  simplex_maksimasi($ruas_kiri,
    $two var, $two cons)
```

Gambar 4. *Pseudocode* Program untuk Minimasi

Pada fase satu akan dilakukan perhitungan penyelesaiaan dalam bentuk minimasi. Tujuan dari fase satu adalah menghilangkan *artificial variable*. Secara umum langkah untuk penyelesaiaan bentuk minimasi sebagai berikut:

a. Melakukan inisiasi nilai *var* dan *cons* dimana nilai *two_cons two_var* merupakan nilai *cons* dan *var* ditambah dengan 1.
b. Medefinisikan *array* ruas kiri dengan dimensi *two_cons*two_var*
c. Selanjutnya menentukan variabel input dengan mencari nilai absolut positif terbesar dari arrau ruas kiri [0][1] sampai ruas kiri[0][8], selanjutnya simpan nilai dari ruas kiri kolom dengan nilai absolut terbesar sebagai *column pivot*.
d. Menentukan *row pivot* dengan mencari baris ruas kiri dengan nilai *rasio* terkecil. *Rasio* terkecil diperoleh dengan membandingkan nilai dari *column pivot* dengan ruas kiri[1][8] sampai ruas kiri[8][8]
e. Memperbarui nilai *row*, jika *row=row pivot*, update *row* dengan value of *row/* value of *key*. Selain itu *update row* dengan nilai row dari kolom – (nilai row lama*angka colum pivot di baris yang sama).
f. Lakukan hal yang sama untuk nilai koeffisien ruas kiri[0][1] sampai ruas kiri[0][8] kurang dari atau sama dengan 0
g. Kemudian melanjutkan pada fase 2 yaitu fase maksimasi untuk mencari nilai fungsi keanggotaan *fuzzy* dengan memanggil fungsi simplex_maksimasi.

Selanjutnya menghitung nilai profit produksi. Perhitungan profit produksi dijelaskan dengan *pseudocode* pada gambar 5.

```
Input : nilai_lamda, ruas_kiri
Output :Z
Proses :
function hitung_profit($ruas_kiri, $nilai_lamda)
initial $Z=0
for $m=1 to $m<=8 do
  $nilai=$ruas_kiri[1][$m]*$nilai_lamda[$m];
  $Z=sum total value of($nilai)
endfor
```

Gambar 5. Code Program Perhitungan Total Profit

Perhitungan total profit (Z) produksi menggunakan langkah-langka sebagai berikut:

a. Menginisiasi nilai dari Z dengan nilai 0
b. Mengalikan nilai dari matriks ruas kiri[1][1] sampai ruas_kiri[1][8] dengan matriks nilai_lamda[1] sampai nilai_lamda[8] secara bergantian
c. Menjumlahkan seluruh hasil perkalian pada langkah b

5.4 Analisa Hasil

Pada tahapan analisa hasil dilakukan uji optimasi dan analisa sensitivitas. Uji optimal digunakan untuk mengetahui hasil penyelesaian persamaan pada model persamaan PL dan FLP optimal atau tidak. Penyelesaian dikatakan optimal menurut [11] dan [2] adalah:

5.4.1 Persamaan dengan Penyelesaian Metode Simpleks

a. bentuk penyelesaiaan minimasi dikatakan optimal jika dan hanya jika nilai dari seluruh koeffisien ruas kiri fungsi tujuan kurang dari atau sama dengan 0.
b. bentuk maksimasi dikatakan optimal jika dan hanya jika nilai dari seluruh koeffisien ruas kiri fungsi tujuan lebih dari atau sama dengan 0

5.4.2 Persamaan dengan Penyelesaian Simpleks Dua Fase

a. Pada fase 1 dikatakan optimal jika nilai dari R atau *artificial variable* sama dengan 0 dan seluruh nilai ruas kiri pada fungsi tujuan sama dengan 0
b. Pada fase 2 sama dengan syarat optimal pada penyelesaian dengan metode Simpleks

5.4.3 Analisa Sensitivitas

Analisa sensitivitas digunakan untuk mengetahui faktor-faktor dari produksi yang paling berpengaruh terhadap pennyelesaiaan optimal, sehingga diperoleh faktor atau kendala yang mengikat kenaikan profit produksi.

6. HASIL DAN PEMBAHASAN

Pada tahapan ini akan disajikan hasil penyelesaiaan dan langkah-langkah memperoleh penyelesaiaan model persamaan FLP serta diikuti analisa hasil yang terdiri dari uji optimasi dan analisa sensitivitas.

6.1 Mencari Nilai Koefisien Fungsi Tujuan dan Batasan

Terdapat 8 variabel tujuan yaitu x_1 sampai x_8 dengan keterangan seperti yang ditunjukkan pada tabel 3. Kemudian menyusun persamaan fungsi tujuan. Pada tahap berikutnya akan dilakukan perhitungan profit produksi untuk setiap item. Nilai profit tersebut selanjutnya akan digunakan sebagai koefisien ruas kiri dari fungsi tujuan (c_j). Berikut contoh perhitungan profit produksi untuk tas dengan kode ELL:

$$bp_{ELL} = \frac{(420000\times1)+(26000\times1)+(150\times1000)+(250\times1000)+(25000\times1)}{1000}$$

$$bp_{ELL} = \frac{871000}{1000} = 871$$

$$profit_{ELL} = harga\ jual - bp_{ELL} \quad \text{..... (14)}$$

$$profit_{ELL} = 1709 - 871 = 838$$

Menggunakan cara yang sama dilakukan perhitungan profit produksi untuk tas dengan kode DO, DAM, DD, DS, RJ1, RJ2, dan AW seperti yang ditunjukkan pada tabel 5.

Tabel 5. Perhitungan Profit Produksi

Kode	ELL	DO	DAM	DD	DS	RJ1	RJ2	AW
Profit	838	1980	750	1682	2392	1626	1590	2805

Kemudian dilakukan pembentukan fungsi batasan terhadap hasil analisa jadwal induk produksi yang dijelaskan pada tabel 5. Selanjutnya pada tabel 6 tersedia data jumlah permintaan untuk bulan Januari–Mei yang digunakan untuk menentukan nilai maksimum dan minimum produksi, sehingga dari data tersebut dapat ditentukan berapa kapasitas maksimal produksi dalam satu kali produksi. Kemudian menyusun dalam bentuk persamaan batasan nomer 8 sampai nomer 15 (pada tabel 4).

Tabel 6. Data Permintaan Bulan Januari-Mei

BULAN	JENIS TAS							
	ELL	DO	DAM	DD	DS	RJ1	RJ2	AW
Januari	1200	623	400	300	157	500	800	0
Februari	1000	500	300	200	150	500	800	0
Maret	1000	500	300	200	150	500	800	0
April	1500	696	313	137	171	500	600	550
Mei	1200	448	400	122	148	0	0	450

Selanjutnya dilakukan perhitungan kebutuhan waktu untuk setiap proses produksi. Sebagai contoh perhitungan untuk proses penggambaran dan pemotongan pola untuk tas kode ELL:

$$proses1_{ELL} = \frac{total\ produksi}{waktu\ produksi\ (menit)} \quad \text{..... (15)}$$

$$proses1_{ELL} = \frac{1000}{4{,}5*60} = 3{,}7037 \approx 3{,}70$$

Menggunakan cara yang sama untuk proses 2 sampai dengan proses 7 dengan hasil perhitungan seperti yang ditunjukkan pada tabel 7.

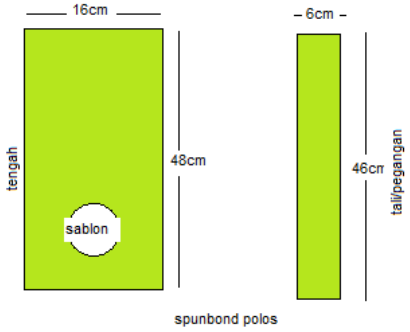
Tabel 7. Perhitungan Waktu Produksi

Proses	JENIS TAS							
	ELL	DO	DAM	DD	DS	RJ1	RJ2	AW
1	3,70	0,97	2	0,95	0,83	1,67	1,43	1,51
2	2,38	1,67	0	1,67	1,67	0	0	2,38
3	0,36	2	0,32	0,18	0,19	0,15	0,19	0,22
4	0	0	0	1,67	0	1,67	0	0
5	0	0,53	0	0	5	0	0	0
6	0	0	0	0	0	1	1,67	0
7	0,73	1,96	1,06	1,87	1,29	1,29	1,96	1,29

Data perhitungan pada tabel 7 digunakan untuk menyusun persamaan fungsi batasan nomer 1 sampai 7. Kemudian dilakukan perhitungan penggunaan bahan baku. Sebagai contoh adalah penggunaan kain spunbond polos untuk tas dengan kode ELL.

a. Bentuk Pola

Gambar 7 menunjukkan bentuk pola tas kode ELL.



Gambar 6. Pola Tas Jenis ELL

b. Perhitungan pola

$$bahansp_{ELL} = \frac{kebutuhan\ pola}{luasan\ tersedia} \quad \text{..... (16)}$$

$$bahansp_{ELL} = \frac{(48*6)+(48*16)}{1600000} = 0{,}00066$$

Karena hasil pembagian menunjukkan nilai mendekati null maka dikalikan dengan 10^5 begitu pula dengan nilai dari koefisien ruas kanan. Lakukan hal yang sama untuk seluruh jenis seperti yang ditunjukkan pada tabel 8.

Tabel 8. Penggunaan Bahan Baku

Bahan baku	JENIS TAS							
	ELL	DO	DAM	DD	DS	RJ1	RJ2	AW
1	66	110,8	7,75	29,25	0	70,96	128,9	0
2	0	9,67	5,33	0	0	1,83	1,23	0
3	0	0	0	12	9,43	0	0	0
4	8	0	0	0	10,67	0	0	0
5	0	1,04	0	0	0	0	0	0
6	0	2,5	0	0	0	2	0	0
7	0	0	0	0	0	8,76	20,44	0
8	0	1	0	0	0	1	0	0
9	0	0	0	1	0	0	1	0

Selanjutnya pada tabel 9 menjelaskan tentang jumlah bahan baku pada bulan Juni. Ketersediaan bahan baku tersebut digunakan sebagai nilai *Righthand Side Ranges*(RHS) pada fungsi batasan nomer 16 sampai 24.

Tabel 9. Persediaan Bahan Baku Bulan Juni

Bulan	Jumlah Persediaan Bahan								
	sp	sm	D600	puring	tali	knop	st	pl	pr
Juni	8	2	1,5	1,13	50	21	3	1	2

Keterangan:

sp :spunbond polos pl : plastik st : kancing stopper
sm : spunbond motif pr : perekat

6.2 Menyusun Persamaan PL

Pada Tahapan ini dilakukan proses penyusunan bentuk persamaan Program Linier sebagai berikut:

a. Pembentukan fungsi tujuan.

Pembentukan fungsi tujuan menggunakan tabel 4 yang disesuaikan dengan bentuk persamaan program linier pada bagian 3, sehingga diperoleh bentuk persamaan 17 dan 18:

$$Max\ Z = 838x_1 + 1980x_2 + 750x_3 + 1682x_4 + 2392x_5 + 1626x_6 + 1590x_7 + 2805x_8$$ ..... (17)

b. Pembentukan fungsi batasan

Menyusun tabel 5,6,dan 7 ke dalam bentuk persamaan 13, sehingga diperoleh bentuk persamaan fungsi batasan sebagai berikut :

1) $37.04x_1 + 9.71x_2 + 20x_3 + 9.52x_4 + 8.33x_5 + 16,67x_6 + 14,29x_7 + 15,15x_8 \leq 80660$
2) $23.81x_1 + 16.67x_2 + 16.67x_4 + 16.67x_5 + 23,81x_8 \leq 80660$
3) $3.63x_1 + 2x_2 + 3.17x_3 + 1.87x_4 + 1.93x_5 + 1,52x_6 + 1,97x_7 + 2,17x_8 \leq 80660$
4) $1.67x_4 + 1,67x_6 \leq 2144$
5) $5,33x_2 + 5x_5 \leq 34460$
6) $1x_6 + 1,67x_7 \leq 5000$
7) $7.27x_1 + 19.56x_2 + 10.56x_3 + 18.67x_4 + 12.89x_5 + 12.89x_6 + 19,56x_7 + 12.89x_8 \leq 80660$
8) $x_1 \leq 1500$
9) $x_2 \leq 696$
10) $x_3 \leq 400$
11) $x_4 \leq 300$
12) $x_5 \leq 171$
13) $x_6 \leq 1000$
14) $x_7 \leq 1000$
15) $x_8 \leq 1000$
16) $66x_1 + 110.75x_2 + 7.75x_3 + 29.25x_4 + 70,97x_6 + 128,88x_7 + 200x_8 \leq 800000$
17) $9.67x_2 + 5.33x_3 + 1,83x_6 + 1,23x_7 \leq 20000$
18) $12x_4 + 9.43x_5 \leq 5000$
19) $8x_1 + 10.67x_3 \leq 20000$
20) $1.04x_2 \leq 1000$
21) $2.5x_2 + 2x_6 \leq 3000$
22) $8,76x_7 + 20,44x_8 \leq 100000$
23) $x_2 + x_6 \leq 1500$
24) $x_4 + x_7 \leq 3024$ ..... (18)

6.3 Persamaan FLP

Mengubah bentuk persamaan 17 dan 18 dalam bentuk persamaan 1, sehingga diperoleh bentuk persamaan 19 dan 20 sebagai berikut:

a. Fungsi tujuan:

$$Max\ \tilde{Z} = 838x_1 + 1980x_2 + 750x_3 + 1682x_4 + 2392x_5 + 1626x_6 + 1590x_7 + 2805x_8$$ ..... (19)

b. Fungsi batasan:

1) $37.04x_1 + 9.71x_2 + 20x_3 + 9.52x_4 + 8.33x_5 + 16,67x_6 + 14,29x_7 + 15,15x_8 \leq \widetilde{80660}$
2) $23.81x_1 + 16.67x_2 + 16.67x_4 + 16.67x_5 + 23,81x_8 \leq \widetilde{80660}$
3) $3.63x_1 + 2x_2 + 3.17x_3 + 1.87x_4 + 1.93x_5 + 1,52x_6 + 1,97x_7 + 2,17x_8 \leq \widetilde{80660}$
4) $1.67x_4 + 1,67x_6 \leq \widetilde{2144}$
5) $5,33x_2 + 5x_5 \leq \widetilde{34460}$
6) $1x_6 + 1,67x_7 \leq \widetilde{5000}$
7) $7.27x_1 + 19.56x_2 + 10.56x_3 + 18.67x_4 + 12.89x_5 + 12.89x_6 + 19,56x_7 + 12.89x_8 \leq \widetilde{80660}$
8) $x_1 \leq \widetilde{1500}$
9) $x_2 \leq \widetilde{696}$
10) $x_3 \leq \widetilde{400}$
11) $x_4 \leq \widetilde{300}$
12) $x_5 \leq \widetilde{171}$
13) $x_6 \leq \widetilde{1000}$
14) $x_7 \leq \widetilde{1000}$
15) $x_8 \leq \widetilde{1000}$
16) $66x_1 + 110.75x_2 + 7.75x_3 + 29.25x_4 + 70,97x_6 + 128,88x_7 + 200x_8 \leq \widetilde{800000}$
17) $9.67x_2 + 5.33x_3 + 1,83x_6 + 1,23x_7 \leq \widetilde{20000}$
18) $12x_4 + 9.43x_5 \leq \widetilde{5000}$
19) $8x_1 + 10.67x_3 \leq \widetilde{20000}$
20) $1.04x_2 \leq \widetilde{1000}$
21) $2.5x_2 + 2x_6 \leq \widetilde{3000}$
22) $8,76x_7 + 20,44x_8 \leq \widetilde{100000}$
23) $x_2 + x_6 \leq \widetilde{1500}$
24) $x_4 + x_7 \leq \widetilde{3024}$ ..... (20)

Kemudian menghitung penambahan atau pengurangan sumberdaya yang tersedia pada koefisien ruas kanan fungsi batasan untuk menentukan nilai parameter *fuzzy*.

i. Penambahan-pengurangan bahan

$$t_i = \frac{\sqrt{\frac{(294)^2 + (-220)^2 + (-136)^2 + (880)^2 + (-818)^2}{5-1}}}{3586} \times 100\%$$

$$t_i = 0,1763 \times 100\% = 17,63\%$$

ii. Parameter *fuzzy*

Pada tabel 10 akan dijelaskan tentang perhitungan nilai parameter *fuzzy* dari koefisien ruas kanan.

Tabel 10. Parameter *Fuzzy*

Batasan	b_{ij}	p_{ij} (t_i*b_{ij})	$b_{ij} + p_{ij}$
1	80660	12718	93378
2	80660	12718	93378
3	80660	12718	93378
4	2144	338	2482
5	34460	5433	39893
6	50000	788	5788
7	80660	12718	93378
8	1500	237	1737
9	696	110	806
10	400	63	463
11	300	47	347
12	171	27	198
13	1000	158	1158
14	1000	158	1158
15	1000	158	1158
16	800000	141026	941026
17	20000	3153	23153
18	5000	788	5788
19	20000	3153	23153
20	1000	158	1158
21	3000	473	3473
22	100000	15767	115767
23	1500	237	1737
24	3024	477	3501

Proses defuzzifikasi lMencari nilai batas bawah optimal dan batas atas optimal dengan program simulasi.

1) Batas bawah optimal (Z_l)

$$Z_l = \max 838x_1 + 1980x_2 + 750x_3 + 1682x_4 + 2392x_5 + 1626x_6 + 1590 +2805x_8$$

dengan batasan:

$$37.04x_1 + 9.71x_2 + 20x_3 + 9.52x_4 + 8.33x_5 + 16{,}67x_6 + 14{,}29x_7 + 15{,}15x_8 \leq 80660$$
$$23.81x_1 + 16.67x_2 + 16.67x_4 + 16.67x_5 + 23{,}81x_8 \leq 80660$$
$$\vdots$$
$$x_4 + x_7 \leq 3024 \quad \text{..... (21)}$$

2) Batas atas optimal (Z_u)

$$Z_u = \max 838x_1 + 1980x_2 + 750x_3 + 1682x_4 + 2392x_5 + 1626x_6 + 1590 +2805x_8$$

dengan batasan:

$$37.04x_1 + 9.71x_2 + 20x_3 + 9.52x_4 + 8.33x_5 + 16{,}67x_6 + 14{,}29x_7 + 15{,}15x_8 \leq 94879$$
$$23.81x_1 + 16.67x_2 + 16.67x_4 + 16.67x_5 + 23{,}81x_8 \leq 94879$$
$$\vdots$$
$$x_4 + x_7 \leq 3557 \quad \text{..... (22)}$$

Diperoleh hasil penyelesaian subproblem menggunakan program simulasi seperti yang ditunjukkan pada Tabel 11 di bawah ini.

Tabel 11. Penyelesaian Subproblem

Variabel	Lower	Upper
Z	8740375,9	10279653,26
X1	555,9	654,11
X2	696	819
X3	400	471

Tabel 11. Penyelesaian Subproblem (Lanjutan)

Variabel	Lower	Upper
X4	288,42	339,33
X5	171	201
X6	804	945
X7	1000	1176
X8	1000	1176

a. Nilai fungsi keanggotaan *fuzzy*

Nilai fungsi keanggotan *fuzzy* diperoleh dengan memodifikasi persamaan PL yaitu menambahkan variabel baru yang mewakili nilai fungsi keanggotaan *fuzzy* (λ). Tujuan dari pencarian nilai keanggotaan *fuzzy* adalah untuk menguji syarat kelayakan dari penambahan angka *fuzzy* dalam sebuah persamaan PL dengan bentuk persamaan 22.

Fungsi tujuan : Max λ

Fungsi batasan :

1) $-1575277\lambda + 1980x_2 + 750x_3 + 1682x_4 + 2392x_5 + 1626x_6 + 1590x_7 + 2805x_8 \geq 8704376$
2) $14219\lambda + 37.04x_1 + 9.71x_2 + 20x_3 + 9.52x_4 + 8.33x_5 + 16{,}67x_6 + 14{,}29x_7 + 15{,}15x_8 \leq 98479$
3) $14219\lambda + 23.81x_1 + 16.67x_2 + 16.67x_4 + 16.67x_5 + 23{,}81x_8 \leq 98479$
4) $14219\lambda + 3.63x_1 + 2x_2 + 3.17x_3 + 1.87x_4 + 1.93x_5 + 1{,}52x_6 + 1{,}97x_7 + 2{,}17x_8 \leq 98479$
5) $378\lambda + 1.67x_4 + 1{,}67x_6 \leq 2522$
6) $6075\lambda + 5{,}33x_2 + 5x_5 \leq 40535$
7) $881\lambda + 1x_6 + 1{,}67x_7 \leq 5881$
8) $14219\lambda + 7.27x_1 + 19.56x_2 + 10.56x_3 + 18.67x_4 + 12.89x_5 + 12.89x_6 + 19{,}56x_7 + 12.89x_8 \leq 98479$
9) $264\lambda + x_1 \leq 1764$
10) $123\lambda + x_2 \leq 819$
11) $71\lambda + x_3 \leq 471$
12) $53\lambda + x_4 \leq 353$
13) $30\lambda + x_5 \leq 201$
14) $176\lambda + x_6 \leq 1176$
15) $176\lambda + x_7 \leq 1176$
16) $176\lambda + x_8 \leq 1176$
17) $141026\lambda + 66x_1 + 110.75x_2 + 7.75x_3 + 29.25x_4 + 70{,}97x_6 + 128{,}88x_7 + 200x_8 \leq 941026$
18) $3526\lambda + 9.67x_2 + 5.33x_3 + 1{,}83x_6 + 1{,}23x_7 \leq 23526$
19) $5881\lambda + 12x_4 + 9.43x_5 \leq 5881$
20) $3526\lambda + 8x_1 + 10.67x_3 \leq 23526$
21) $176\lambda + 1.04x_2 \leq 1176$
22) $529\lambda + 2.5x_2 + 2x_6 \leq 3529$
23) $17628\lambda + 8{,}76x_7 + 20{,}44x_8 \leq 117628$
24) $264\lambda + x_2 + x_6 \leq 1764$
25) $533\lambda + x_4 + x_7 \leq 3557$ ..... (23)

Persamaan 22 dapat diselesaikan dengan metode simpleks dua fase dan diperoleh hasil penyelesaian menggunakan program simulasi seperti yang ditunjukkan pada Tabel 12.

Tabel 12. Penyelesaian Bentuk FLP

Nilai Koefisien							
X1	X2	X3	X4	X5	X6	X7	X8
605	757	435	313	186	874	1088	1088
Z = 9510003							

6.4 Analisa Uji Optimal

Tabel 13 menjelaskan tentang hasil uji optimasi persamaan Program Linier dan ada tabel 14 menujukkan hasil perhitungan optimal dari persamaan PL yang termodifikasi (yang telah ditambahka nilai fungsi keanggotaan *fuzzy*).

Tabel 13. Uji Optimasi Hasil Penyelesaiaan Subproblem

i	in	out	C								Δy	Z	
			1	2	3	4	5	6	7	8		lower	upper
1	-	-	-838	-1980	-750	-1682	-2392	-1626	-1590	-2805	0	0	0
2	c8	s15	-838	-1980	-750	-1682	-2392	-1626	-1590	0	2465,27	2805000	3298680
3	c5	s12	-838	-1980	-750	-1682	0	-1626	-1590	0	1102,189	3214032	3779472
4	c2	s9	-838	0	-750	-1682	0	-1626	-1590	0	512,54	4592112	5401092
5	c4	s18	-838	0	-750	0	0	-1626	-1590	0	123,18	5077229	5971850
6	c6	s23	-838	0	-750	0	0	0	-1590	0	1263,62	6384533	7508420,67
7	c7	s14	-838	0	-750	0	0	0	0	0	1272,919	7974533	9378260,67
8	c1	s1	0	-297,027	0	0	0	0	0	0	22,64864	8621565	10139753,53
9	c3	s10	0	0	0	0	0	0	0	0	297,027	8740376	10279653,26

Tabel 14. Uji Optimasi Penyelesaiaan FLP

fase	iterasi	*in*	*out*	C									Z
				λ	1	2	3	4	5	6	7	8	
1	1	0	0	-1539277	838	1980	750	1682	2392	1626	1590	2805	8740375,9
1	2	c9	s16	-2032957	838	1980	750	1682	2392	1626	1590	0	5441695,9
1	3	c6	s13	-2104717	838	1980	750	1682	0	1626	1590	0	4960903,9
1	4	c3	s10	-2348257	838	0	750	1682	0	1626	1590	0	3339283
1	5	c5	s19	-243389,8	838	0	750	0	0	1626	1590	0	2768525,2
1	6	c7	s24	-2663164,8	838	0	750	0	0	0	1590	0	1231955,2
1	7	c8	R1	0	0	0	0	0	0	0	0	0	0
2	8	-	-	-1	0	0	0	0	0	0	0	0	0
2	9	c1	s15	1	0	-0,0002847	0	-0,0002548	0	0	0	0	0,2167
2	10	c2	s24	1	0	0	0	-9,7148	0	0	0	0	0,45769
2	11	c4	s11	1	0	0	0	0	0	0	0	0	0,5

Pada tabel hasil uji 13 dan 14 dapat kita ketahui bahwa hasil penyelesaiaan dari persamaan PL dan FLP memenuhi syarat optimal seperti yang dijelaskan pada bagian 5.5.

6.5 Analisis Sensitivitas

Analisis sensitivitas digunakan untuk mengetahui faktor-faktor yang paling berpengaruh pada proses produksi (mencari *binding constraint dan unbinding constraint*) ditunjukkan pada Tabel 15.

Tabel 15. Analisa Sensitivitas Fungsi Batasan

Batasan	RHS	Surplus	Min RHS	Max RHS	Dual Prices
1.	80660	0	60091,75	115591,8	17
2.	80660	26387,664	54272,34	*infinity*	0
3.	980660	72136,89	8523,11	*infinity*	0
4.	2144	1051,58	1092,42	*infinity*	0

Tabel 15. Analisa Sensitivitas Fungsi Batasan (Lanjutan)

Batasan	RHS	Surplus	Min RHS	Max RHS	Dual Prices
5.	34460	30125	4335	*infinity*	0
6.	5000	3196	1804	*infinity*	0
7.	80660	11653,21	69006,79	*infinity*	0
8.	15000	994,1	55,9	*infinity*	0
9.	696	0	500	1000	463
10.	400	0	0	1428	410
11.	300	11,58	288,42	*Infinity*	0
12.	171	0	155,56	555,56	1253
13.	1000	196	804	Infinit	0
14.	1000	0	0	1712,68	1353
15.	1000	0	0	2271,88	2550

Tabel 15. Analisa Sensitivitas Fungsi Batasan (Lanjutan)

Batasan	RHS	Surplus	Min RHS	Max RHS	Dual Prices
16.	800000	291306	508693,39	*Infinity*	0
17.	20000	9932	10068	*Infinity*	0
18.	5000	0	1539	5139	111,4
19.	20000	13842,811	6157	*Infinity*	0
20.	1000	304	696	*Infinity*	0
21.	3000	0	3000	*Infinity*	682
22.	100000	73568	26432	*Infinity*	0
23.	1500	0	696	1696	0
24.	3024	1735,9	1288	*infinity*	0

Tabel 16. Analisisis Sensitivitas Fungsi Tujuan

Obj Coeff	Min Obj Coeff	Max Obj Coeff	Reduced Cost
838	0	1387,5	0
1980	1467,46	*infinity*	0
750	452,97	*infinity*	0
1682	203,84	3151,59	0
2392	1289,81	*Infinity*	0
1626	362,38	2138,54	0
1590	317,08	*infinity*	0
2805	339,73	*infinity*	0

Nilai 0 dari variabel Surplus pada tabel 16 menunjukkan bahwa batasan (*constraint)* tersebut merupakan *binding constraint* atau batasan yang mengikat profit Z. Sehingga jika nilai dari sumberdaya atau bahan baku dari batasan tersebut dinaikkan akan meningkatkan profit produksi sejumlah nilai dari *Dual Prices*.

7. KESIMPULAN DAN SARAN

Persamaan Fuzzy Proglin dapat diselesaikan menggunbakan metode Simpleks Fuzzy, bereda dengan persamaan Proglin yang diselesaikan dengan metode Simpleks. Penyelesaian dengan metode Simpleks Fuzzy merupakan penggabungan penyelesaian fuzzy dengan metode simpleks biasa diikut dengan metode simpleks dua fase. Secara singkat penyelesaian dari metode Simpleks Fuzzy sebagai berikut:

1. Mencari nilai batas atas dan batas bawah optimal dengan metode simpleks fuzzy
2. Menghitung parameter fuzzy kemudian menyusun bentuk persamaan baru untuk menghitung nilai fungsi keanggotaan fuzzy dan menyelesaikannya dengan metode simpleks dua fase.

Menggunakan program simulasi diperoleh hasil perkiraan nilai profit produksi optimal untuk Program Linier sebesar Rp Rp 8.740.375,00 dan menggunakan persamaan *Fuzzy Linear Programming* sebesar Rp 9.510.003,00 dengan nilai fungsi keanggotaan fuzzy sebesar 0,5. Nilai 0,5 dari fungsi keanggotaan *fuzzy trapezoidal* menunjukkan bahwa penambahan nilai *fuzzy* pada ruas kanan dari permasalahan PL dapat diterapkan untuk menyesaiakan permasalahan produksi UKM Cantik Souvenir. Nilai 0,5 dari fungsi keanggotaan *fuzzy* memenuhi syarat veasible penggunaan nilai *fuzzy* dalam persamaan PL yaitu berada direntang 0 sampai dengan 1.

Pada penelitian selanjutnya akan lebih baik lagi jika ditambahkan nilai koefisien *fuzzy* pada fungsi tujuan dan menyelesaikan perhitungan optimasi dengan *fungsi ranking*.

8. DAFTAR PUSTAKA

[1] R. Purba, "Penerapan Logika Fuzzy Pada Program Linear," in *Kontribusi Pendidikan Matematika dan Matematika dalam Membangun Karakter Guru dan Siswa*, 2012, pp. 101–114.

[2] F. S.Hillier and G. J. Lieberman, *Introduction to Operation Research*, 5th ed. McGraw-Hill, 1990.

[3] I. A. Marie, Y. Arkeman, and D. U. Daihani, "Penentuan Jumlah Produksi Menggunakan Model Fuzzy Multi Objective Linear Programming Pada Industri Pangan (Studi Kasus Pada Industri Roti PT NIC)," pp. 38–46, 2011.

[4] D. Cahaya N., I. Santoso, and M. Effendi, "Perencanaan Produksi Keripik Kentang Menggunakan Metode Fuzzy Linear Programming (FLP) (Studi Kasus di UKM Agronas Gizi Food Kota Batu)," no. December 2014, pp. 1–7, 2014.

5] S. Effati. and H. Abbasiyan., "Solving Fuzzy Linear Programming Problems with Linear Membership Functions," *Appl. Appl. Math. An Int. J.*, vol. 26, no. 2, pp. 375 – 396, 2002.

[6] A. T. Afriani, N. Kusumastuti, and B. Prihandono, "Metode simpleks fuzzy untuk permasalahan pemrograman linear dengan variabel trapezoidal fuzzy," vol. 01, no. 1, pp. 23–30, 2012.

[7] A. I. Suryani, L. Linawati, and H. A. Parhusip, "Fuzzy Linear Programming dengan Fungsi Keanggotaan Kurva-S untuk Penilaian Kinerja Karyawan," *Pros. Semin. Nas. Sains dan Pendidik. Sains VIII UKSW*, pp. 431–436, 2014.

[8] A. I. Suryani, L. Linawati, and H. A. Parhusip, "Analisis Penilaian Kinerja Karyawan Menggunakan Fuzzy Linear Programming (FLP)," *Pros. Semin. Nas. Penelitian, Pendidik. dan Penerapan MIPA UNY,* 2013.

[9] B. Sharma and R. Dubey, "Optimum Solution of Fuzzy Linear Programming," vol. 3, no. 7, pp. 268–276, 2012.

[10] A. Indrasari and J. Gunawan, "Menentukan Jumlah Produksi Menggunakan Logika Fuzzy Linier Programming Pada Industri Roti (Studi Kasus Pada PO. Mungil)," *J. Ilm. Tek. Ind. dan Inf.*, vol. 3, no. 1, pp. 19–27, 2014.

[11] W. L. Winston, *Operations Research Applications and Algorithms*, 3rd ed. United States of America, 1994.

Analisis Perbandingan Metode *Vector Space Model* dan *Weighted Tree Similarity* dengan *Cosine Similarity* pada kasus Pencarian Informasi Pedoman Pengobatan Dasar di Puskesmas

Viko Basmalah Wicaksono
Jurusan Informatika
Universitas Sebelas Maret
Jl. Ir. Sutami 36A Kentingan
Surakarta

viko.jrs@gmail.com

Ristu Saptono
Jurusan Informatika
Universitas Sebelas Maret
Jl. Ir. Sutami 36A Kentingan
Surakarta

ristu.uns@gmail.com

Sari Widya Sihwi
Jurusan Informatika
Universitas Sebelas Maret
Jl. Ir. Sutami 36A Kentingan
Surakarta

sari.widya.sihwi@gmail.com

ABSTRAK

Sistem pencarian merupakan salah satu solusi yang dapat membantu dalam mendapatkan informasi yang diinginkan. Dengan sistem pencarian, proses pencarian informasi akan menjadi lebih efisien. Sistem pencarian informasi pada *ebook* pedoman pengobatan di puskesmas sangat dibutuhkan karena terdapat banyak data penyakit di dalamnya. Dalam mengembangkan sistem pencarian pada pedoman pengobatan di puskesmas, dapat memanfaatkan metode *Vector Space Model* (VSM) atau *Weighted Tree Similarity* (WTS). Penelitian ini membandingkan metode VSM dengan WTS untuk mendapatkan metode terbaik. Selain itu, ditambahkan algoritma *Hamming Distance* untuk mengetahui pengaruh eksekusi waktu sistem.

Penelitian ini menunjukkan bahwa WTS lebih baik dibandingkan VSM, hal ini dapat dilihat pada hasil pengujian, nilai *precision* pada WTS lebih baik dibandingkan VSM. Karena pada metode pencarian yang efektif adalah yang memberikan nilai ketepatan(*precision*) terbaik, meskipun nilai *recall* lebih rendah. Pada pengujian sistem, VSM menunjukkan hasil nilai rata – rata *precision* sebesar 44,82983 % dan *recall* sebesar 99,08165 %. Sedangkan pada WTS nilai rata – rata *precision* sebesar 52,17332% dan *recall* sebesar 98,61761%. Kemudian pada pengujian pakar menunjukkan *precision* WTS dengan rata – rata sebesar 46,675% dan *recall* sebesar 73,6111%. Sedangkan nilai *precision* VSM sebesar 33,6737% dan nilai *recall* sebesar 86,8056%.

Algoritma *Hamming Distance* sangat membantu dalam mempercepat eksekusi sistem. Pengaruh penggunaan algoritma *Hamming Distance* pada VSM memberikan hasil denganrata – rata waktu pengujian adalah 4,512 detik, sedangkan tanpa *Hamming Distance* adalah 9,185 detik. Kemudian Pada metode WTS dengan *Hamming Distance* memberikan hasil rata – rata dengan waktu pengujian adalah 6,042 detik, sedangkan tanpa *Hamming Distance* adalah 14,421 detik.

Kata kunci : *Pedoman Pengobatan Dasar Puskesmas, Sistem pencarian, Vector Space Model, Weighted Tree Similarity*

1. PENDAHULUAN

Dengan banyaknya informasi yang ada di dalam buku Pedoman Pengobatan Dasar di Puskesmas, maka untuk mendapatkan informasi yang dicari dibutuhkan waktu yang relatif lama apabila dilakukan secara manual. Terdapat 109 data penyakit yang masing – masing data memiliki banyak informasi di dalamnya. Apalagi jika terdapat perubahan dan penambahan data, maka informasi yang terbaru akan berbeda dengan informasi yang didapatkan sebelumnya. Dari penjelasan tersebut, permasalahan yang Pada adalah bagaimana cara mendapatkan informasi secara cepat dan akurat. Salah satu solusi dari permasalahan tersebut adalah dengan membuat sistem pencarian.

Sistem pencarian (*Search Engine*) merupakan sistem yang dapat digunakan untuk menemukan informasi yang relevan dengan kebutuhan dari penggunanya secara otomatis dari suatu koleksi informasi [1] . Sistem ini akan menerima masukan berupa kata kunci dari informasi yang akan dicari. Dengan waktu yang relatif lebih singkat akan memberikan hasil beberapa dokumen atau informasi yang relevan terhadap kata kunci yang dimasukkan pengguna.

Dalam sistem temu kembali diperlukan metode yang cocok diterapkan dalam sistem, salah satunya adalah dengan Metode VSM (*Vector Space Model*). VSM adalah suatu metode untuk merepresentasikan sistem temu kembali ke dalam vektor dan memperhitungkan fungsi *similarity* dalam proses pencocokan beberapa vektor. Suatu sistem temu kembali terdiri atas dua bagian, yaitu penyimpanan dokumen dan pemrosesan *query*, baik *query* maupun dokumen-dokumen yang disimpan dinyatakan dalam bentuk vektor [2]. Metode VSM dipilih karena cara kerja model ini efisien, mudah dalam representasi, dan dapat diimplementasikan pada *document-matching* [3].

Selain menggunakan VSM, metode lain untuk pencarian semantik dapat menggunakan WTS (*Weigthed Tree Similarity*). Pencarian semantik dengan *Weighted Tree Similarity* sebelumnya digunakan pada penelitian [5]. *Weighted Tree Similarity* adalah algoritma yang digunakan untuk mengukur tingkat kemiripan dua buah *tree* yang berbobot [4]. Pada *Weighted Tree Similarity*, perhitungan *similarity* pada sub*tree* dihitung dengan metode *Cosine Similarity*. Metode ini telah diterapkan untuk mencocokkan transaksi *e-business*, pencarian obyek pembelajaran, *virtual market* untuk jaringan listrik, transaksi kendaran roda

empat, estimasi biaya proyek, dan pencarian informasi yang tepat untuk *handheld device* [5].

Dalam mempercepat proses pencarian menggunakan metode VSM maupun WTS, digunakan cara menghitung hubungan antar dokumen. Salah satu metode untuk mengetahui hubungan antar dokumen adalah algoritma *Hamming distance*. Pada penelitian [6], penggunaan algoritma *Hamming Distance* diterapkan pada kalimat dalam proses pencarian data dengan bentuk biner agar proses menjadi lebih cepat.

Berdasarkan rincian diatas penulis ingin melakukan analisa perbandingan metode *Vector Space Model* dan *Weighted Tree Similarity* dalam menentukan hasil akurasi sistem pencarian Informasi Pedoman Penyakit Dasar di Puskesmas. Selain itu, dianalisa pengaruh penggunaan algoritma *Hamming Distance* dalam waktu eksekusi pencarian.

2. LANDASAN TEORI

2.1. *Text Preproccessing*

Tahapan *Preproccesing* merupakan tahap persiapan yang dilakukan untuk menyiapkan dokumen sebelum diolah. Penggunaan *text preproccesing* dilakukan karena dokumen teks tidak dapat diproses langsung oleh algoritma pencarian, sehingga diperlukan proses untuk menghasilkan data numerik yang akan digunakan dalam perhitungan. Tahapan *Text Preproccesing* pada penelitian ini meliputi [7]:

a. Penghapusan format dan *markup* dalam dokumen
b. *Tokenizing*
c. *Filtering*
d. *Stemming*

2.2. *Cosine Similarity*

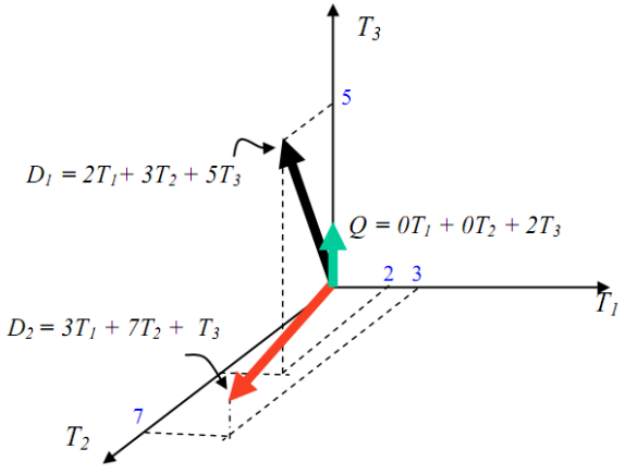
Metode ini digunakan untuk menghitung nilai cosinus sudut antara dua *vector* dan mengukur kemiripan antar dua dokumen [10]. Metode *Cosine Similarity* menggambarkan suatu kesamaan antara vektor *query* dan vektor dokumen dengan dilihat dari sudut yang paling kecil. Perhitungan *Cosine Similarity* dirumuskan pada persamaan (1) berikut ini.

$$sim(d_j, q) = \frac{\vec{d_j} \bullet \vec{q}}{|\vec{d_j}| \bullet |\vec{q}|} = \frac{\sum_{i=1}^{t}(w_{ij} \bullet w_{iq})}{\sqrt{\sum_{i=1}^{t} w_{ij}^2} \bullet \sqrt{\sum_{i=1}^{t} w_{iq}^2}} \quad (1)$$

Dimana:
$\vec{q}$= bobot *query*
$|\vec{q}|$ = panjang *query*
$\vec{d_j}$ = bobot dokumen
$|\vec{d_f}|$= panjang dokumen

2.3. *Vector Space Model* (VSM)

Vector Space Model (VSM) merupakan model yang digunakan untuk mengukur kemiripan antara dokumen dan *query* yang mewakili setiap dokumen dalam sebuah koleksi sebagai sebuah titik dalam ruang [8]. Dalam metode *Vector Space Model* dihitung *weighted* dari setiap *term* yang terdapat dalam semua dokumen dan *query* dari *user*. *Term* adalah kata atau kumpulan kata yang merupakan ekspresi verbal dari suatu pengertian. Penentuan relevansi dokumen dengan *query* dipandang sebagai pengukuran kesamaan antara vektor dokumen dengan vektor *query*. Contoh representasi relevansi antara dokumen dan *query* dapat digambarkan pada Gambar 1. Q merupakan *query* pembanding, D_1 dan D_2 adalah dua dokumen yang akan dibandingkan, sedangkan T_1, T_2 dan T_3 adalah tiga *term* pada dokumen tersebut.



Gambar 1. Representasi dokumen dan *query* pada ruang vektor [8]

Pada perhitungan VSM digunakan pembobotan TF-IDF dan perhitungan nilai *similarity* dengan menggunakan *Cosine Similarity*. Metode TF-IDF adalah cara untuk memberikan bobot hubungan suatu *term* terhadap dokumen. Metode ini menggabungkan dua konsep perhitungan bobot yaitu frekuensi kemunculan kata dalam suatu dokumen dan *inverse* dari frekuensi yang mengandung kata tersebut [9]. Persamaan dalam perhitungan TF-IDF terdapat pada rumusan (2) dan (3) sebagai berikut:

$$W_{(t,d)} = TF_{(t,d)} x\ IDF \quad (2)$$

$$W_{(t,d)} = TF_{(t,d)} x Log \frac{D}{DF_t} \quad (3)$$

Dimana :
W(t,d) : bobot *term* t pada dokumen d
TF (t,d) : total kemunculan *term* t pada dokumen d
D : totalseluruh dokumen
DF_t : total dokumen yang memiliki *term* t

2.3. *Weighted Tree Similarity* (WTS)

Weighted Tree Similarity merupakan salah satu algoritma untuk menentukan tingkat kesamaan dua objek yang disajikan dalam bentuk *tree* [11]. Pada *Weighted Tree Similarity* dokumen yang sudah melalui proses *pre proccessing* direpresentasikan kedalam bentuk *tree*. *Tree* dalam *Weighted Tree* memiliki konsep node atau titik yang berlabel, *arc* atau cabang berlabel, dan *arc* berbobot [5].

Proses pembobotan pada *term* dilakukan dengan memperhatikan TF(*Term Frequency*) dalam sebuah dokumen [5]. Perhitungan pembobotan TF dilakukan dengan persamaan berikut :

$$W = TF/TF_{total} \quad (4)$$

Dimana :
TF = Frekuensi suatu kata
TF_{total} = total frekuensi pada seluruh kata

Dalam perhitungan *similarity* pada *Weighted Tree Similarity*, bobot pada semua bobot cabang berpengaruh pada kemiripan total [4]. Perhitungan dilakukan dengan menghitung rata-rata persamaan aritmatika. Perumusan perhitungan kemiripan *tree* terdapat dalam persamaan (5), yaitu sebagai berikut:

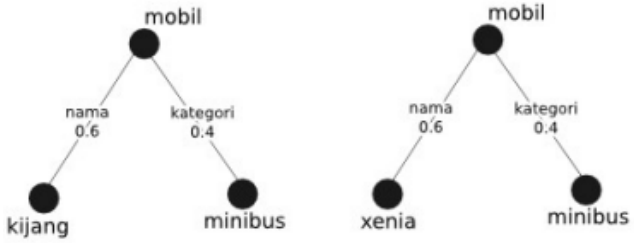
$$\boldsymbol{Sim_{tot}} = \sum(\boldsymbol{S_i} * \boldsymbol{W_i}) \quad (5)$$

Dimana

w_i = bobot cabang *tree* pertama

S_i = kemiripan cabang

Nilai rata-rata bobot hasil perhitungan dikalikan dengan nilai kemiripan cabang (S_i) yang dihasilkan dari perhitungan similarity. Perhitungan *Weighted Tree Similarity* dimulai dengan memberi nilai 1 jika *leaf node*-nya sama dan diberi nilai 0 jika *leaf node*-nya berbeda.



Gambar 2. Contoh perhitungan dasar kemiripan *tree* [4]

Dari Gambar 2 di atas dapat dirumuskan sebagai berikut:

$$\boldsymbol{Sim_{tot}} = (\boldsymbol{Sim}\,[\boldsymbol{kijang}, \boldsymbol{xenia}] * \boldsymbol{0,6}) + (\boldsymbol{Sim}\,[\boldsymbol{minibus}, \boldsymbol{minibus}] * \boldsymbol{0,4})$$

$$\boldsymbol{Sim_{tot}} = (\boldsymbol{0} * \boldsymbol{0.6}) + (\boldsymbol{1} * \boldsymbol{0.4})$$

$$\boldsymbol{Sim_{tot}} = \boldsymbol{0.4}$$

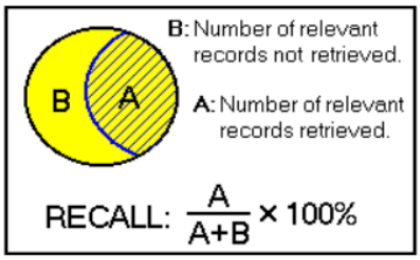
Dari contoh perhitungan pada gambar diatas nilai kemiripan kedua *tree* adalah 0.4. Proses ini akan berulang jika terdapat *subtree* yang lain.

2.4. *Hamming distance*

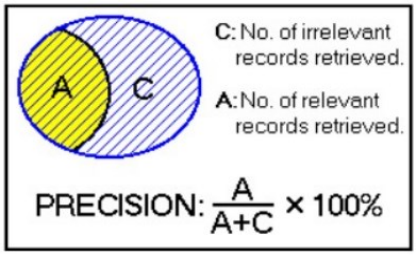
Hamming Distance adalah salah satu algoritma untuk mengukur kedekatan *item*. Jika nilai kedekatannya makin kecil, maka kedua *item* itu semakin dekat dan berlaku sebaliknya. Yang biasanya dibandingkan adalah kata dan bilangan *biner* [5]. Selain digunakan untuk membandingkan kata dan bilangan biner, algoritma *Hamming Distance* juga dapat digunakan pada himpunan kata.

2.5. *Precision* dan *Recall*

Perhitungan *Precision* dan *Recall* merupakan perhitungan yang umum digunakan untuk mengevaluasi hasil pencarian.



Gambar 3. Digram Penjalasan *Recall* [12]



Gambar 4. Digram penjelasan *Precision* [12]

Recall adalah perbandingan antara hasil pencarian yang relevan dengan seluruh data relevan yang ada pada koleksi *database*. Sedangkan *precision* adalah perbandingan antara hasil pencarian yang relevan terhadap semua pencarian yang berhasil di *retrieve*. *Recall* dan *precision* digunakan dalam mengukur tingkat keberhasilan pencarian. Semakin tinggi ukuran *precision* dan *recall*-nya maka semakin bagus strategi pencariannya. Selain itu, suatu sistem dinyatakan efektif apabila hasil penelusuran mampu menunjukkan ketepatan (*precision*) yang tinggi sekalipun perolehan *recall* rendah [12].

3. METODOLOGI PENELITAN

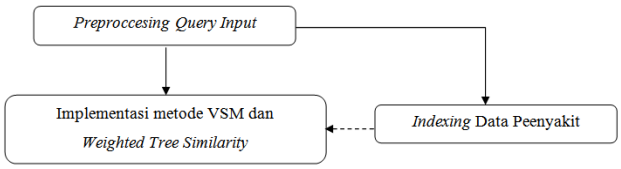
3.1. Studi Literatur

Studi literatur digunakan untuk mendapatkan dasar – dasar referensi yang kuat dalam membantu penyusunan laporan penelitian. Studi literatur dilakukan dengan mengumpulkan literatur yang berkaitan dengan tatalaksana penyakit manusia, metode *Vector Space Model, Weghted Tree Similarity, Stemming* serta *Cosine Similarity*. Sumber – sumber literatur dapat berupa buku, teks, paper, jurnal, karya ilmiah, dan situs – situs yang dapat menunjang penyelesaian penelitian.

3.2. Pengumpulan Data

Tahap pengumpulan data merupakan lanjutan dari studi literatur yang memerlukan data sebagai objek yang akan diolah dan diproses. Data penyakit yang digunakan diambil dari *ebook* Pedoman Pengobatan Dasar di Puskesmas dengan judul "PEDOMAN PENGOBATAN DASAR DI PUSKEMAS" yang diterbitkan oleh Departemen Kesehatan pada tahun 2007.

3.3. Penerapan metode

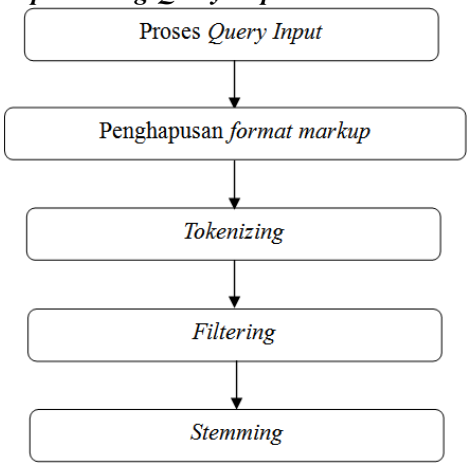


Gambar 5. Proses Penerapan Metode pada Sistem

Pada penelitian ini penerapan metode meliputi Pada penelitian ini penerapan metode meliputi *Preproccesing Query Input* , *Indexing* data penyakit dan Implementasi metode VSM dan *Weighted Tree Similarity*. *Preprocessing query input* merupakan tahapan awal untuk memodelkan *input* sebelum diolah. Pada proses selanjutnya adalah tahapan implementasi metode VSM dan *Weighted Tree Similarity*. Pada tahapan ini akan menghitung proses kemiripan dengan mengambil data penyakit yang sudah diproses dalam *indexing* ke dalam *database*. *Indexing* data

penyakit bertujuan agar pada saat proses penghitungan tidak terlalu lama.

3.3.1. *Preprocessing Query Input*

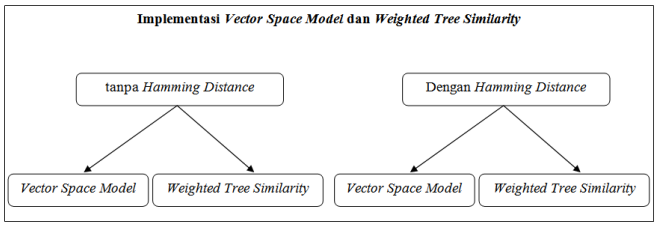


Gambar 6. Proses *Text Preprocessing*

Pada gambar 6 merupakan proses *query input* pada sistem. Proses *query input* dilakukan oleh user, dimana *input* tersebut akan dilakukan tahap *preproccesing*. Pada tahapan ini dilakukan penghapusan *markup* format untuk menghilangkan beberapa format HTML yang tidak digunakan untuk perhitungan. Selanjutnya proses *tokenizing* untuk memotong *input* menjadi *term* kata yang nantinya akan dihitung untuk proses pembobotannya. Kemudian dilakukan proses *Filtering* untuk mengambil beberapa kata penting dan dillanjutkan dengan proses *stopwords removal*. *Stopwords removal* yaitu membuang kata – kata yang dianggap tidak penting dan tidak diperlukan dalam perhitungan. Tahapan terakhir adalah proses *Stemming* untuk memproses kata kembali ke kata dasarnya. Pada penelitian ini digunakan *stemming* Nazief Adriani. Pada tahap *stemming* untuk proses pencarian data, *library* kata dasar yang digunakan adalah *library* dari *term* yang ada pada dokumen. Hal ini dilakukan supaya mempercepat proses *stemming*.

3.3.2. Implementasi metode VSM dan Weighted *Tree Similarity*

Pada proses implementasi metode VSM dan *Weighted Tree Similarity* dapat dilihat pada gambar 7. Proses awal merupakan proses *pre-distance* atau proses penghitungan kedekatan antar dokumen. Langkah ini dibagi menjadi dua jenis, yaitu dengan menggunakan algoritma *Hamming Distance* dan tanpa menggunakan algoritma *Hamming Distance*. Proses ini dilakukan untuk melihat pengaruh penggunaan algoritma *Hamming Distance*. Setelah proses ini selesai dilakukan, kemudian masuk ke dalam tahapan penghitungan kemiripan dengan masing – masing metode.



Gambar 7. *Implementasi* Vector Space Model *dan* Weighted Tree Similarity

3.3.2.1. Implementasi *Hamming Distance*

Pada tahap ini penerapan *Hamming Distance* dilakukan sebelum masuk ke penghitungan pencarian. Implementasi *Hamming Distance* dilakukan dengan melihat hubungan kedekatan antara dokumen dan *query input* dengan melihat *term* apa saja yang terlibat didalamnya. Setelah dilihat kedekatannya, apabila *query* dan dokumen yang akan dibandingkan memiliki persamaan, maka akan dilanjutkan ke dalam proses perhitungan metode selanjutnya. Sedangkan apabila tidak terdapat kedekatan sama sekali antara dokumen dan *query* maka perhitungan akan berhenti dan tidak dihitung ke dalam perhitungan metode selanjutnya.

3.3.2.2. *Vector Space Model*

Pada tahapan ini dijelaskan proses yang terjadi pada metode *Vector Space Model*. Proses tersebut dibagi menjadi tiga, yaitu proses *distance* atau penentuan jarak kedekatan antara dokumen, proses pembobotan dan yang terakhir penghitungan *Similarity*.

Tabel 1. Proses dan Metode pada Vector Space Model

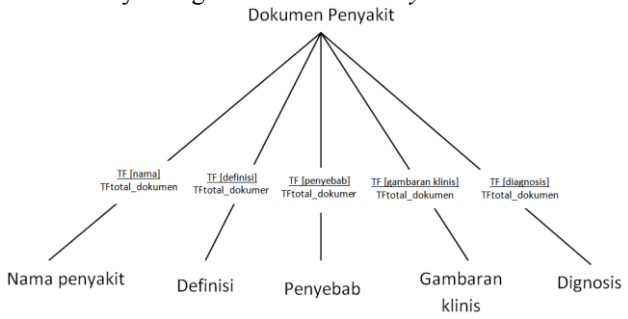
Proses	Vector Space Model
Distance	1
	Hamming distance
Bobot	Pembobotan TF – IDF
Similarity	*Cosine Similarity*

Pada tabel 1, proses VSM dimulai dengan penghitungan jarak (*distance*) untuk mengetahui kedekatan antara *query* dengan dokumen *database*. Pada proses ini dibagi menjadi dua jenis, yaitu menggunakan *Hamming Distance* dan tanpa menggunakan *Hamming Distance* yaitu jarak dianggap 1 atau sama untuk semua dokumen. Langkah selanjutnya adalah proses pembobotan dengan menggunakan TF-IDF. Setelah dilakukan pembobotan, selanjutnya dihitung nilai *similarity* dengan metode *Cosine Similarity*.

3.3.2.3. *Weighted Tree Similarity*

Proses pada WTS (*Weighted Tree Similarity*) langkahnya mirip pada metode VSM, yaitu yang pertama untuk jarak akan dibagi menjadi dua dengan menggunakan *Hamming Distance* dan tanpa menggunakan *Hamming Distance* atau jarak dianggap sama pada *query* dengan semua dokumen. Selanjutnya pada proses pembobotan pada *term* dilakukan dengan pembobotan TF pada tiap parameter.

Pada metode ini, parameter tiap dokumen dibentuk menjadi *tree* data berbobot dan tiap *tree* data dokumen dibuat dengan struktur sama. Nilai bobot tiap parameter diambil dari total *trem frequency* masing – masing parameter dibagi total *term frequency* pada dokumen. Bentuk struktur *tree* pada dokumen penyakit dapat dilihat pada gambar 7. Setelah dihitung bobot pada masing – masing node *tree,* selanjutnya dihitung masing – masing similaritasnya dengan *Cosine Similarity*.



Gambar 8. Susunan *Tree* pada dokumen penyakit

3.3. Pengembangan Aplikasi

Pada penelitian ini dikembangkan sebuah aplikasi pencarian Informasi Pedoman Pengobatan Dasar di Puskesmas dengan menggunakan bahasa pemrograman PHP dan *database* MySql.

3.4. Skenario Pengujian

3.5.1. Pengujian Sistem

Dalam mengevaluasi hasil dari sistem ini dilakukan dengan menghitung *precision* dan *recall* . Penghitungan ini digunakan untuk mengetahui baik atau tidaknya hasil dari pencarian yang digunakan. Pada pemilihan metode terbaik sistem pencarian, dipilih hasil yang memberikan nilai ketepatan (*precision)* lebih tinggi, meskipun nilai *recall* lebih rendah. Selain menghitung *Precision* dan *Recall*, dihitung juga waktu proses yang dilakukan sistem. Perhitungan waktu yang diperlukan digunakan untuk mengetahui efektifas waktu yang digunakan oleh Sistem dari beberapa pemodelan yang digunakan. Prosedur pengujiannya dilakukan dengan mengambil *keyword* dari masing masing *term* di dokumen penyakit yang memiliki frekuensi paling tinggi. Pengujian juga dipisahkan berdasarkan jumlah kata dalam *keyword* yang terbentuk.

3.5.2. Pengujian dengan Pakar

Kemudian yang terakhir dilakukan uji dengan ahli pakar untuk mengetahui apakah sejauh mana relevansi data yang dikembalikan oleh Sistem dengan kata kunci yang dimasukan. Peran pakar disini adalah untuk memilih data mana yang relevan terhadap kata kunci yang dimasukkan. Pakar yang dipilih untuk melakukan pengujian adalah orang yang ahli dibidang penyakit dasar dan kesehatan masyarakat. Kemudian kata kunci yang akan digunakan dalam pengujian didapatkan dari referensi pakar dan penulis. Setelah pakar selesai memilih data mana yang relevan, kemudian akan diproses kembali dalam sistem dengan membandingk VSM dan *Weighted Tree Similarity* untuk mendapatkan nilai *precision* dan *recall.*

4. HASIL DAN PEMBAHASAN

4.1. Deskripsi Data

Data pada penelitian akan menggunakan 109 data penyakit yang ada pada *ebook* Pedoman Pengobatan di Puskesmas yang diterbitkan Kementrian Kesehatan pada tahun 2007Dimana data penyakit tersebut akan disimpan ke dalam *database* yang terdiri dari parameter nama penyakit, definisi, penyebab, gambaran klinis, diagnosis dan penatalaksanaan.

4.2. Penerapan Metode

4.2.1. *Preprocessing Query* Input

Proses *preprocessing query input* bertujuan untuk menyiapkan *query input* yang akan dibandingkan dengan dokumen yang ada di *database*. Pada proses ini dilakukan proses penghapusan format *markup* , *tokeninzing*, *filtering* dan *stemming* . Setelah terbentuk *term* dari *query input* dihitung frekuensi *term* yang akan digunakan sebagai *query* pembanding. Berikut contoh proses *Preproccessing Query Input* dengan *input* user "*Gusi Berdarah dan bengkak"*.

Tabel 2. Penghapusan format *markup* dan *tokenizing*

Query	**Penghapusan Format**	***Tokenizing***
Gusi Berdarah dan bengkak,	gusi berdarah dan bengkak	gusi
		berdarah
		dan
		bengkak,

Proses yang pertama adalah penghapusan format dan *markup* pada *query input* yang dapat dilihat pada tabel 2. Proses ini mengembalikan *query input* dari *user* ke dalam bentuk huruf kecil untuk dilanjutkan ke proses selanjutnya. Selain mengembalikan ke huruf kecil, pada proses ini juga menghapus beberapa format tag yang tidak diperlukan. Selanjutnya, pada proses *tokenizing*, dilakukan pemecahan *query* menjadi beberapa *term* yang dipisahkan berdasarkan spasinya.

Token	***Filtering***	**Hasil *Filtering***
gusi	gusi	gusi
berdarah	berdarah	darah
dan	~~dan~~	bengkak
bengkak,	bengkak~~,~~	

Gambar 9. Proses *Filtering Query Input*

Gambar 9 merupakan proses *filtering.* Pada proses *filtering* bertujuan untuk menghilangkan tanda baca dan symbol yang dianggap tidak penting dan digunakan dalam perhitungan. Pada proses ini juga dilakukan penghapusan *stopwords* yang tidak digunakan dalam perhitungan.

Tabel 3. Hasil *Stemming*

gusi
darah
bengkak

Pada tabel 3 merupakan hasil proses *Stemming* yang bertujuan untuk mengembalikan token yang sudah dipilih kedalam bentuk kata dasarnya. Selanjutnya *term* tersebut yang akan digunakan untuk dihitung frekuensinya.

Tabel 4. Perhitungan Frekuensi *Query Input*

Term	Frekuensi
gusi	1
darah	1
bengkak	1

Langkah yang terakhir dalam proses *preprocessing* adalah menghitung frekuensi dari masing – masing *term query input* . Hasil perhitungan frekuensi dapat dilihat pada tabel 4. Frekuensi kemunculan inilah yang selanjutnya akan digunakan dalam proses pembobotan

4.2.2. Indexing data Penyakit

Tabel 5. Contoh data asli yang belum di*index*

Parameter	Isi dokumen
Nama penyakit	Ginggivitis
Definisi	Ginggivitis adalah Peradangan pada gusi
Penyebab	Karang gusi dan plak
Gambaran Klinis	Gusi bengkak dan mudah berdarah, bau mulut dan nyeri
Diagnosis	Peradangan pada gusi

Sebelum masuk tahap implementasi VSM dan *Weighted Tree Similarity* Dokumen penyakit dokumen penyakit akan dimasukkan dan di-*index* ke dalam *database*. Langkahnya dengan menghitung masing – masing frekuensi *term* dari dokumen kemudian disimpan berdasarkan parameter penyakit. Data pada tabel 5 merupakan contoh data dokumen asli dari penyakit yang di ambil dari buku Pedoman Pengobatan di Puskesmas. Data asli tersebut dimasukkan secara manual oleh admin. Data yang dimasukkan kemudian diolah dan diproses berdasarkan pada proses *text preproccesing* untuk disimpan melalui proses *indexing*.

Contoh Dokumen yang sudah diindex di *Database*

Term	Frekuensi					Total
	NP	D	P	GK	Dg	Frekuensi
ginggivitas	1	1	0	0	0	2
radang	0	1	0	0	1	2
gusi	0	1	1	1	1	4
karang	0	0	1	0	0	1
plak	0	0	1	0	0	1
bengkak	0	0	0	1	0	1
darah	0	0	0	1	0	1
nyeri	0	0	0	1	0	1
bau	0	0	0	1	0	1
mulut	0	0	0	1	0	1

Keterangan :
NP : nama penyakit GK : Gambaran Klinis
D : Definisi Dg : Diagnosis
P : Penyebab

Pada tabel 6merupakan hasil proses *indexing* dokumen penyakit yang ada di *database.* Proses *indexing* menghitung masing – masing frekuensi pada parameter dokumen, karena pada perhitungan *Weighted Tree Similarity* dihitung pada tiap – tiap parameter dan pada VSM perhitungan akan dilakukan dengan menggunakan total frekuensi *term* terhadap dokumen. Selanjutnya data yang sudah di-*index* dihitung ke perhitungan metode selanjutnya.

Pada tahapan *indexing* ditemukan bahwa terdapat *term* yang memiliki banyak frekuensi di semua dokumen. *Term* yang memiliki frekuensi di semua dokumen dapat juga untuk dianggap sebagai *stopword* karena dapat memberikan hasil yang kurang relevan saat perhitungan.

4.2.3. Implementasi Metode VSM dan *Weighted Tree Similarity*

Penerapan implementasi VSM dan *Weighted Tree Similarity* dilakukan untuk mendapatkan metode pencarian yang paling efektif. Dokumen yang akan dibandingkan adalah dokumen penyakit yang sudah diinput dan di*index* ke dalam *database*. Langkah pertama perhitungan dilakukan dengan menghitung masing – masing frekuensi dari dokumen kemudian dilakukan pembobotan terhadap masing – masing data dokumen penyakit. Setelah proses pembobotan selesai kemudian dihitung masing – masing *similarity*nya dengan menggunakan metode *Cosine Similarity*.

Data pada tabel 5 merupakan data dokumen asli dari penyakit yang di ambil dari buku Pedoman Pengobatan di Puskesmas. Data mentah tersebut di*input*kan secara manual oleh admin. Data yang di*input*kan kemudian diolah dan diproses berdasarkan pada proses *text preproccesing* untuk melalui proses *indexing*.

Pada tabel 6 merupakan hasil proses *indexing* dokumen penyakit yang ada di *database.* Proses *indexing* menghitung masing – masing frekuensi pada parameter dokumen, karena pada perhitungan *Weighted Tree Similarity* dihitung pada tiap – tiap parameter dan pada VSM perhitungan akan dilakukan dengan menggunakan total frekuensi *term* terhadap dokumen. Selanjutnya data yang sudah di-*index* dihitung ke perhitungan metode selanjutnya.

4.2.2.1. Penerapan Algoritma *Hamming Distance*

Penggunan *Hamming Distance* dilakukan sebelum masuk pada proses penghitungan metode pencarian. Jadi, proses *Hamming Distance* bertujuan untuk mengetahui bagaimana hubungan kedekatan antara *query input* dengan masing – masing dokumen yang ada pada *database*. Sehingga apabila terdapat dokumen yang sama sekali tidak memiliki kedekatan dengan *query input* tidak akan dilanjutkan dalam perhitungan pencarian. Pada penelitian ini diterapkan penerapan Algoritma *Hamming Distance* pada himpunan kata. Berikut contoh penerapan algoritma *Hamming Distance* pada himpunan kata yang digunakan dalam penelitian ini:

1. Cek panjang himpunan kata. Sebagai contoh berikut :

   Query input = {(gusi),(berdarah)}
   Dokumen *Database* = {(gigi),(berdarah)}

   Dari contoh diatas dilakukan analisa panjang kedua himpunan kata :

   - Pada *Query* input memiliki panjang 2 anggota himpunan, dengan anggota himpunan : gusi dan berdarah.
   - Pada Dokumen *database* memiliki panjang 2 anggota himpunan, dengan anggota himpunan : gigi dan berdarah.

2. Pilih salah satu himpunan sebagai pembanding. Misal kalimat 1 sebagai pembanding.
3. Cek anggota himpunan yang posisinya sama. Jika setiap anggota anggota himpunan pada kedua himpunan memiliki kata yang sama, maka nilainya 0, dan jika berbeda akan diberi nilai 1.

Tabel 13. Contoh penerapan algoritma *Hamming distance*

***Query* Input**	**Dokumen**	**Status**	**Nilai Kedekatan**
gusi		Beda	1
	gigi	Beda	1
berdarah	berdarah	Sama	0
TOTAL			2

4. Tabel 4 merupakan hasil penghitungan, karena kedekatan *Query* dan dokumen adalah 2, maka dapat disimpulkan bahwa *query* dan dokumen cukup mirip.

4.2.2.2.Metode Vector Space Model

Pada implementasi metode VSM akan membandingkan data *query* input yang sudah melalui proses *preproccessing* dengan membandingkan data dokumen penyakit yang ada di *database*.

Tabel 7. Contoh Pembobotan VSM pada *query* dan dokumen penyakit Ginggivitis

Term	**TF**		**IDF**	**Bobo *Term***	
	Q	**P**		**Q**	**P**
ginggivitas	0	2	0,301	0	0,602
radang	0	2	0,301	0	0,602
gusi	1	4	1	1	4
karang	0	1	0,301	0	0,301
plak	0	1	0,301	0	0,301
bengkak	1	1	1	1	1
darah	1	1	1	1	1
nyeri	0	1	0,301	0	0,301
bau	0	1	0,301	0	0,301

Pada Tabel 7 dijelaskan contoh salah satu penghitungan pembobotan TF-IDF dengan *keyword* yang sudah di proses menjadi *query* input. Q merupakan frekuensi dari *query* input sedangkan P merupakan Frekuensi *term* pada dokumen penyakit. Setelah frekuensi Q dan P sudah dihitung, langkah selanjutnya adalah menghitung nilai IDF pada masing – masing dokumen. Dan yang terakhir nilai bobot dihitung dengan mengalikan nila frekuensi masing – masing *term* pada Q dan P dengan nilai IDF.

Setelah proses pembobotan selesai maka didapatkan hasil bobot dari masing – masing *term* terhadap *query* dan dokumen di *database*. Bobot yang sudah didapatkan tersebut digunakan untuk dihitung kemiripannya. Proses penghitungan dilakukan dengan membandingkan kemiripan dari Bobot Q dan Bobot P kedalam bentuk vektor, kemudian dihitung dengan metode *Cosine Similarity*.

Similarity (*Query* , Penyakit)

$$= \frac{\sum(\boldsymbol{Bobot_Q} * \boldsymbol{Bobot_P})}{\sqrt{\sum(\boldsymbol{Bobot_Q}^2)} * \sqrt{\sum(\boldsymbol{Bobot_P}^2)}}$$

$$= \frac{\mathbf{6}}{\mathbf{1{,}73205} * \mathbf{4{,}37927}}$$

$$= \mathbf{0{,}79102}$$

Dari hasil perhitungan *similarity* didapatkan bahwa *similarity* antara *query* yang di*input* kan dengan dokumen penyakit Ginggivitas yang ada di *database* adalah **0,79102**.

4.2.2.3.Metode *Weighted Tree Similarity*

Proses perhitungan pada *Weighted Tree Similarity* dilakukan dengan menghitung masing – masing bobot pada parameter dokumen penyakit. Langkah – langkah perhitungan pada *Weighted Tree Similarity* adalah sebagai berikut:

1. Langkah pertama yaitu membandingkan *term* frekuensi *query* yang sudah di*input* kan, kemudian dibandingkan dengan masing – masing parameter pada dokumen penyakit. Pada Tabel 8,9,10,11 dan 12 dijelaskan pembobotan *term* pada tiap parameter dengan pembobotan TF. Prosesnya hampir sama dengan metode VSM, yaitu Q merupakan frekuensi *term* pada *Query* dan P merupakan frekuensi *term* pada dokumen *database*. Selanjutnya dihitung total frekuensi pada masing – masing Q dan P yang digunakan dalam perhitungan bobot *term* masing – masing *term*.

Tabel 8 Contoh pembobotan TF pada *Query* dan Dokumen Penyakit Ginggivitas parameter nama penyakit

Term	**TF**		**Pembobotan TF**	
	Q	**P**	**Q**	**P**
gusi	1	0	0,33333	0
darah	1	0	0,33333	0
bengkak	1	0	0,33333	0
ginggivitas	0	1	0	1
Total Frekuensi	3	1		

Tabel 9 Contoh pembobotan TF pada *Query* dan Dokumen Penyakit Ginggivitas parameter definisi

Term	**TF**		**Pembobotan TF**	
	Q	**P**	**Q**	**P**
gusi	1	1	0,3333	0,3333
darah	1	0	0,3333	0
bengkak	1	0	0,3333	0
radang	0	1	0	0,3333
ginggivitis	0	1	0	0,3333
Total Frekuensi	3	3		

Tabel 10 Contoh pembobotan TF pada *Query* dan Dokumen Penyakit Ginggivitas parameter penyebab

Term	TF		Pembobotan TF	
	Q	P	Q	P
gusi	1	1	0,3333	0,3333
darah	1	0	0,3333	0
bengkak	1	0	0,3333	0
karang	0	1	0	0,3333
plak	0	1	0	0,3333
Total Frekuensi	3	3		

Tabel 11 Contoh pembobotan TF pada *Query* dan Dokumen Ginggivitas parameter gambaran klinis

Term	TF		Pembobotan TF	
	Q	P	Q	P
gusi	1	1	0,3333	0,1667
darah	1	1	0,3333	0,1667
bengkak	1	1	0,3333	0,1667
nyeri	0	1	0	0,1667
bau	0	1	0	0,1667
mulut	0	1	0	0,1667
Total Frekuensi	3	6		

Tabel 12 Contoh pembobotan TF pada *Query* dan Dokumen Ginggivitas parameter diagnosis

Term	TF		Pembobotan TF	
	Q	P	Q	P
gusi	1	1	0,3333	0,1667
darah	1	1	0,3333	0,1667
bengkak	1	1	0,3333	0,1667
nyeri	0	1	0	0,1667
Total Frekuensi	3	3		

2. Setelah dihitung bobot *term* pada semua parameter, langkah selanjutnya adalah menghitung nilai *similarity* pada tiap parameter terhadap *query* input. Nilai *similarity* dihitung dengan metode *Cosine Similarity* berdasarkan bobot *term* yang sudah dihitung sebelumnya.

Similarity Nama (*Query* , Penyakit)

$$= \frac{\sum(\boldsymbol{Bobot_Q} * \boldsymbol{Bobot_P})}{\sqrt{\sum(\boldsymbol{Bobot_Q}^2)} * \sqrt{\sum(\boldsymbol{Bobot_P}^2)}}$$
$$= \frac{\mathbf{0}}{\mathbf{0,57743 * 1}}$$
$$= \mathbf{0}$$

Similarity Definisi (*Query* , Penyakit)

$$= \frac{\sum(\boldsymbol{Bobot_Q} * \boldsymbol{Bobot_P})}{\sqrt{\sum(\boldsymbol{Bobot_Q}^2)} * \sqrt{\sum(\boldsymbol{Bobot_P}^2)}}$$
$$= \frac{\mathbf{0,01111}}{\mathbf{0,57735 * 0,47141}}$$
$$= \mathbf{0,40825}$$

Similarity Penyebab (*Query* , Penyakit)

$$= \frac{\sum(\boldsymbol{Bobot_Q} * \boldsymbol{Bobot_P})}{\sqrt{\sum(\boldsymbol{Bobot_Q}^2)} * \sqrt{\sum(\boldsymbol{Bobot_P}^2)}}$$
$$= \frac{\mathbf{0,01111}}{\mathbf{0,57735 * 0,47141}}$$
$$= \mathbf{0,40825}$$

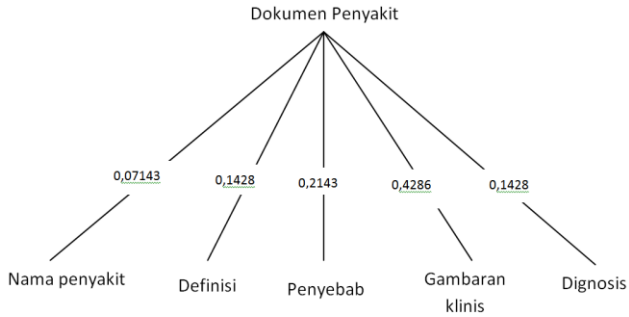
Similarity Gambaran Klinis (*Query* , Penyakit)

$$= \frac{\sum(\boldsymbol{Bobot_Q} * \boldsymbol{Bobot_P})}{\sqrt{\sum(\boldsymbol{Bobot_Q}^2)} * \sqrt{\sum(\boldsymbol{Bobot_P}^2)}}$$
$$= \frac{\mathbf{0,16667}}{\mathbf{0,57735 * 0,40285}}$$
$$= \mathbf{0,70711}$$

Similarity Diagnosis (*Query* , Penyakit)

$$= \frac{\sum(\boldsymbol{Bobot_Q} * \boldsymbol{Bobot_P})}{\sqrt{\sum(\boldsymbol{Bobot_Q}^2)} * \sqrt{\sum(\boldsymbol{Bobot_P}^2)}}$$
$$= \frac{\mathbf{0,01111}}{\mathbf{0,57735 * 0,47141}}$$
$$= \mathbf{0,40825}$$

3. Langkah terakhir pada metode *Weighted Tree Similarity* adalah menghitung nilai *similarity* total dari *query* dan dokumen. Nilai *similarity* dihitung dengan mengalikan nilai *similarity* tiap cabang/ parameter dengan bobot parameter. Bobot parameter didapatkan dari nilai total *term* frekuensi tiap parameter dibagi total frekuensi *term* keseluruhan dokumen. Sehingga didapatkan bobot *tree* parameter seperti pada gambar 10.



Gambar 10. Proses *Filtering Query Input*

4. Setelah didapatkan nilai bobot pada tiap node parameter, selanjutnya dihitung nilai *similarity* total. Langkahnya adalah dengan menghitung nilai total *similarity* pada masing – masing parameter kemudian dikalikan dengan nilai bobot tiap node parameter. Sehingga penghitungan nilai *similarity* total dengan metode *Weighted Tree Similarity* didapatkan sebagai berikut

Similarity Total
= (0*0,7143) + (0,40825*0,1428) + (0,40825*0,2143) + (0,70711*0,4286) + (0,40825*0,1428)
= **0,50715**

Sehingga didapatkan bahwa *similarity* total dengan metode *Weighted Tree Similarity* antara *query* yang di*input* kan dengan dokumen penyakit Ginggivitas yang ada di *database* adalah **0,50715**. Pada hasil penghitungan *similarity* yang diberikan oleh *Weighted Tree Similarity* memberikan nilai *similarity* yang nilainya lebih sedikit dibandingkan VSM. Hal ini kemungkinan dipengaruh oleh nilai bobot paramater *node tree* pada dokumen penyakit yang diambil berdasarkan jumlah *term* yang ada didalamnya. Hal ini bisa dioptimalkan dengan memberikan nilai bobot node yang lebih relevan tiap parameternya, agar nilai *simalirity* lebih tinggi dan hasil lebih relevan.

4.3. Hasil dan Pembahasan

Pada hasil dan evaluasi yang dilakukan, dibagi menjadi 2 kali pengujian, yaitu pengujian dari Sistem dan pengujian dengan pakar.

4.3.1. Pengujian Sistem

4.3.1.1. Penentuan *Keyword* Pengujian

Pada proses pengujian sistem digunakan dengan mengambil *keyword* secara otomatis dari *database*. *Keyword* tersebut diambil dari *term* yang memiliki frekuensi paling tinggi dari masing – masing parameter pada dokumen penyakit. Pengambilan *term* tertinggi diambil kemudian dikombinasikan menjadi sebuah *keyword* yang terdiri dari satu sampai lima kata.

Tujuan dilakukan percobaan ini adalah untuk mendapatkan nilai batasan yang akan digunakan sebagai *threshold* dan juga untuk pengujian sistem. Pada tabel 14 didapatkan hasil dari pembentukan jumlah *keyword* dan banyak kombinasinya. *Keyword* yang terdiri dari 1 kata merupakan *term* dengan frekuensi yang tertinggi dari masing – masing dokumen penyakit. Sedangkan untuk *keyword* yang terdiri dari dua sampai lima kata, merupakan rangkaian dari *term* dengan frekuensi tertinggi dari masing – masing parameter.

Tabel 14. Jumlah Percobaan Sistem

Keyword	**Banyak Kombinasi Percobaan**
1 kata	269
2 kata	16
3 kata	30
4 kata	39
5 kata	23

4.3.1.2. Penentuan Treshold / Batasan *Similarity*

Sebelum masuk ke proses pengujian, terlebih dahulu ditentukan *threshold* yang akan menjadi batasan untuk mengembalikan nilai *similarity*. Nilai *threshold* ditentukan berdasarkan hasil percobaan dari pembentukan *keyword* yang sudah ditentukan. Nilai *threshold* diambil nilai *similarity* terendah dari percobaan yang sudah ditentukan. Nilai *threshold* juga dibagi berdasarkan jumlah kata pada *keyword*. Pada tabel 15 merupakan hasil penentuan nilai *threshold* berdasarkan jumlah kata percobaan untuk metode VSM dan *Weighted Tree Similarity*.

Tabel 15. Hasil Penentuan *Threshold*

Keyword	***Threshold***	
	VSM	***Weighted Tree Similarity***
1 kata	0.0747	0.0065
2 kata	0.1735	0.0562
3 kata	0.2999	0.0959
4 kata	0.7341	0.1906
5 kata	0.6625	0.1271

4.3.1.3. Analisa Hasil Pengujian Sistem

Setelah proses pengujian dari metode yang sudah ditentukan selesai, dilakukan analisis terhadap hasil pengujian dengan menghitung *precision* dan *recall* . Besarnya kecilnya nilai *precision* dan *recall* akan menunjukkan baik tidaknya sebuah metode. Kemudian juga pengujian terhadap waktu eksekusi juga dihitung untuk menghitung cepat tidaknya waktu eksekusi yang digunakan.

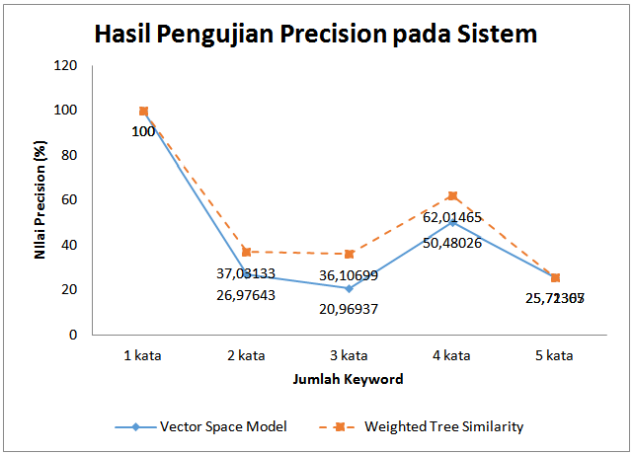
Tabel 16. Hasil Pengujian *Precision* dan *Recall* dengan Metode VSM

Keyword	***Precision***	***Recall***
1 kata	100 %	99,75609 %
2 kata	26,97643 %	100 %
3 kata	20,96937 %	100 %
4 kata	50,48026 %	100 %
5 kata	25,72307 %	95,65217 %
Rata - rata	**44,82983 %**	**99,08165 %**

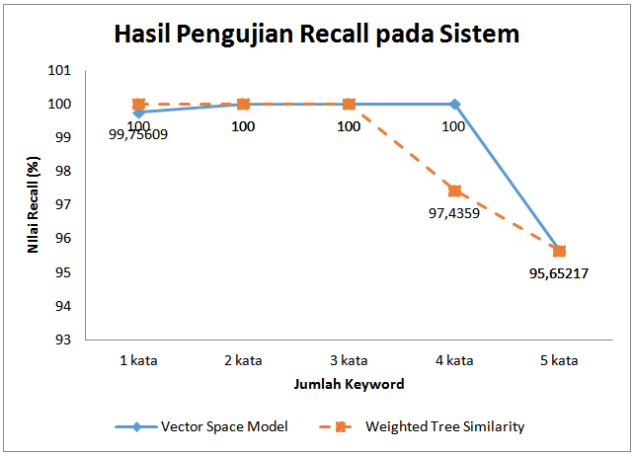
Tabel 17. Hasil Pengujian *Precision* dan *Recall* dengan Metode *Weighted Tree Similarity*

Keyword	***Precision***	***Recall***
1 kata	100 %	100 %
2 kata	37,03133 %	100 %
3 kata	36,10699 %	100 %
4 kata	62,01465 %	97,4359 %
5 kata	25,71365 %	95,65217 %
Rata - rata	**52,17332 %**	**98,61761 %**

Hasil pada percobaan dengan menggunakan metode VSM memberikan nilai rata- rata *precission* 44,82983 % dan *recall* 99,08165% yang bisa dilihat pada tabel 16. Hasil nilai *recall* sudah baik karena hampir mencapai 100%. Sedangkan pada percobaan dengan metode *Weighted Tree Similarity* memberikan hasil yang baik dengan nilai rata – rata *precision* 52,17332 dan *recall* 98,61761 bisa dilihat pada tabel 17. Dari rata – rata nilai *precision* metode ini lebih baik dibandingkan dengan metode VSM. Kemudian dari nilai *recall* sedikit lebih turun, namun selisihnya dengan metode VSM sangat kecil dan tidak terlalu signifikan.



Gambar 10. Grafik Pengujian *Precision* pada Sistem



Gambar 11. Grafik Pengujian *Recall* pada Sistem

Hasil pada percobaan dengan menggunakan metode VSM memberikan nilai rata- rata *precission* 44,82983 % dan *recall* 99,08165% yang bisa dilihat pada tabel 4.15. Hasil nilai *recall* sudah baik karena hampir mencapai 100%. Sedangkan pada percobaan dengan metode *Weighted Tree Similarity* memberikan hasil yang baik dengan nilai rata – rata *precision* 52,17332 dan *recall* 98,61761 bisa dilihat pada tabel 4.16. Dari rata – rata nilai *precision* metode ini lebih baik dibandingkan dengan metode VSM. Kemudian dari nilai *recall* sedikit lebih turun, namun selisihnya dengan metode VSM sangat kecil dan tidak terlalu signifikan.

Tabel 18. Hasil Pengujian Waktu dengan metode VSM

Keyword	**Waktu (detik)**	**Waktu *Hamming distance* (detik)**
1 kata	4,01223	1,31764
2 kata	6,54944	2,57761
3 kata	9,07661	3,82786
4 kata	12,32681	6,82902
5 kata	13,9621	8,00766
Rata - rata	**9,18544**	**4,51196**

Tabel 19. Hasil Pengujian Waktu dengan metode *Weighted Tree Similarity*

Keyword	**Waktu (detik)**	**Waktu *Hamming distance* (detik)**
1 kata	7,92163	1,5081
2 kata	10,88575	3,57722
3 kata	13,96942	5,25262
4 kata	18,8712	9,05895
5 kata	20,45582	10,81487
Rata - rata	**14,42076**	**6,04235**

Selanjutnya pada waktu eksekusi sistem dapat dilihat pada tabel 18 dan 19. Dari hasil tersebut diketahui bahwa pengaruh penggunaan algoritma *Hamming Distance* sangat signifikan. Penggunaan algoritma *Haming Distance* sangat membantu dalam mempercepat proses eksekusi. Karena dengan menggunakan algoritma ini, dokumen yang tidak memiliki kedekatan tidak akan dilanjutkan dalam perhitungan, sehingga waktu eksekusi menjadi lebih cepat.

4.3.2. Pengujian dengan Pakar

Pada pengujian dengan pakar dilakukan dengan menguji beberapa *keyword* yang sudah ditentukan. Selanjutnya *keyword* tersebut akan diuji dengan membandingkan metode VSM dan *Weighted Tree Similarity*. Peran pakar pada pengujian ini akan menentukan hasil pencarian mana saja yang relevan dan tidak relevan. Pengujian ini dilakukan dengan pakar yaitu Bapak dr. Burhanuddin Ichsan, M. Med. Ed, M.Kes. Beliau merupakan Dokter umum yang juga menjabat sebagai Dosen di UMS(Universitas Muhammadiyah Surakarta). Selain seorang dokter, penelitian yang sering dilakukan beliau juga terkait dengan penyakit umum dan kesehatan masyarakat.

Pengujian pakar dilakukan dengan 20 kali percobaan. Untuk hasil pengujian pakar yang asli dan hasil rekapitulasi pengujian pakar dapat dilihat pada lampiran A dan B. Setelah proses pengujian selesai, hasilnya diproses untuk dihitung nilai *precision* dan *recall*. Penghitungan *precision* dan *recall* dilakukan berdasarkan hasil pencarian yang dikembalikan dengan menunjukkan hasil relevan dan tidak relevan yang dipilih oleh pakar. Hasil pengujian yang sudah dilakukan bisa dilihat pada tabel 20 dan 21. Hasil *precision* lebih baik pada metode VSM, sedangkan nilai *recall* lebih tinggi pada metode *Weighted Tree Similarity*.

Tabel 20. Hasil Pengujian Pakar dengan metode VSM

Percobaan	**Precission**	***Recall***
20 kali	34.67365	85.69444

Tabel 21. Hasil Pengujian Pakar dengan metode *Weighted Tree Similarity*

Percobaan	**Precission**	***Recall***
20 kali	41.09162	71.66667

4.3.3. Analisis Metode Terbaik

Dari hasil pengujian pengujian sistem yang dilakukan, disimpulkan pada sub bab 4.4.1. *Weighted Tree Similarity* merupakan daripada metode VSM dengan memberikan nilai pengujian *precision* yang lebih baik. Hal ini juga didukung pada hasil *precision* pada pengujian pakar memberikan hasil lebih baik pada metode *Weighted Tree Similarity* dibandingan VSM, meskipun nilai rata – rata *recall* pada *Weighted Tree Similarity* lebih kecil dibandingkan VSM, selisihnya juga tidak terlalu jauh. Karena pada dasar pengujian *recall* dan *precision* metode pencarian yang lebih efektif adalah yang memberikan nilai ketepatan terbaik yaitu nilai *precision,* meskipun nilai *recall* lebih rendah.

Selanjutnya analisa hasil waktu menunjukkan metode VSM dengan *Hamming Distance* merupakan metode yang paling baik. Namun selisih waktu antara VSM dan *Hamming Distance* dengan *Weighted Tree Similarity* dan *Hamming Distance* tidak terlalu jauh, dan terpaut sekitar satu detik saja. Sehingga dari sini hasil tiga analisa pengujian yang dilakukan, dapat disimpulkan metode *Weighted Tree Similarity* dan *Hamming Distance* merupakan metode terbaik karena memberikan hasil yang lebih efisien dan efektif. Kemudian pada umumnya pencarian yang berhubungan dengan dunia kesehatan lebih mementingkan hasil pencarian yang efisien dan efektif.

5. KESIMPULAN DAN SARAN

Berdasarkan hasil penelitian dan pembahasan yang telah dilakukan, dapat disimpulkan bahwa metode *Weighted Tree Similarity* merupakan metode terbaik. Hal ini sudah dibuktikan pada pengujian sistem dan pakar, metode *Weighted Tree Similarity* memberikan nilai *precision* yang lebih baik dibandingkan dengan VSM, meskipun nilai *recall* lebih kecil dibandingkan pada metode VSM. Sedangkan perbedaan nilai *recall* juga tidak terlalu jauh.

Penggunaan algoritma Hamming Distance sangat berpengaruh dalam membantu mempercepat waktu eksekusi sistem. Hal ini dikarenakan pada algoritma Hamming Distance, apabila terdapat dokumen yang tidak memiliki kedekatan dengan *query* maka tidak akan dilanjutkan ke proses perhitungan selanjutnya.

Adapun beberapa saran yang dipertimbangkan untuk perkembangan penelitian selanjutnya yaitu :

1. Agar proses tidak terlalu lama, pada *term* yang memiliki frekuensi di semua dokumen dianggap sebagai *stopword*.
2. Supaya nilai *similarity* total mejadi lebih tinggi dan hasil yang diberikan lebih relevan, pada metode *Weighted Tree Similarity* dalam menghitung nilai *similarity* total diberikan nilai bobot masing – masing pada parameter yang lebih relevan. Pemilihan nilai bobot dengan melihat parameter mana yang dianggap lebih berpengaruh pada proses perhitungan *similarity*.
3. Untuk pencarian yang lebih relevan dan cepat, dapat mengoptimalkan penggunaan algoritma *Hamming Distance* dengan cara memberi batasan nilai *Hamming*. Batasan tersebut diberikan agar pencarian dengan kata kunci lebih dari satu kata dapat memberikan hasil yang sama dengan makna kata kunci yang dimasukkan tersebut.

6. DAFTAR PUSTAKA

[1] Rila, M., & Setiawan, H. (2001). *Improving Information Retrieval System Performance by Automatic Query Expansion*. *Makalah, Departemen Teknik Informatika, Institut Teknologi Bandung*.

[2] Zafikri A. (2010). Implementasi Metode *Term Frequency Inverse Document Frequency* (TF-IDF) pada Sistem Temu Kembali Informasi.

[3] Amin, F. (2012). Sistem Temu Kembali Informasi dengan Metode *Vector Space Model* . *Jurnal Sistem Informasi Bisnis 02* , 78-83.

[4] Perdana, P. A. (2014). Analisis Kombinasi Algoritma *Weighted Tree Similarity* Dengan Tanimoto Cosine (Tc) Untuk Pencarian Semantik Pada Portal Jurnal. *Prosiding SNST ke-5* , 79-84.

[5] Sarno, R., & Rahutomo, F. (2008). Penerapan Algoritma *Weighted Tree Similarity* Untuk Pencarian Semantik. *JUTI*, 35-42.

[6] Putro, L. S. (2014). Penerapan Kombinasi Algoritma Minhash dan *Binary Hamming Distance* pada *Hybrid* Perekomendasi Lagu.

[7] Sebastiani, F. 2002. *Machine Learning in Automated Text Categorization*

[8] Turney, P. D., & Pantel, P. (2010). From Frequency to Meaning: Vector Space . *Journal of Artificial Intelligence Research*, 141 -188.

[9] Robertson, Stephen 2005, *Understanding Inverse Document Frequency: On Theoretical Arguments for IDF*, *England Journal of Documentation*, Vol. 60, pp. 502 – 520.

[10] Susanto, S., & Sensuse, D. I. (2008, Vol.1 No.2). Pengklasifikasian Artikel Berita Berbahasa Indonesia secara Otomatis Menggunakan Naive Bayes Classifier. *Jurnal Ilmu Komputer dan Informasi*

[11] Sa'adah, Umi ; Sarno, Riyanarto ; Yuhana, Umi L. 2013. *Latent Semantic Analysis and Weighted Tree Similarity For Semantic Search In Digital Library*. The Proceedings of The 7th ICTS, Bali. 159 – 164.

[12] Clarke, S., & Willett, P. 1997. *Estimating the recall performance of search engines*. ASLIB Proceedings, 49 (7), 184-189.

IMPLEMENTASI *ITERATIVE DICHOTOMISER* 3 PADA DATA KELULUSAN MAHASISWA S1 DI UNIVERSITAS SEBELAS MARET

Thanh Thi Bi Dan
Jurusan Informatika
Universitas Sebelas Maret
Jl. Ir. Sutami No. 36 A Surakarta
dantina16@yahoo.com

Sari Widya Sihwi
Jurusan Informatika
Universitas Sebelas Maret
Jl. Ir. Sutami No. 36 A Surakarta
sari.widya.sihwi@gmail.com

Rini Anggrainingsih
Jurusan Informatika
Universitas Sebelas Maret
Jl. Ir. Sutami No. 36 A Surakarta
rinianggra@gmail.com

ABSTRACT

University of Sebelas Maret (UNS) selects new students every year. The selection process usually appears in three ways, known as National Selection for University Entry (SNMPTN), Local Selection for University Entry (SBMPTN), and New Students Admission Selection (SPMB). Every year UNS also passes students through four periods of graduation. From this process, UNS has graduation data that increases every period. These data has not yet been used efectively and left to accumulate unorderly. Necessary technique is needed to process the data that can give standardized and satisfying results, thus by using data mining indeed can solve the problem.

In this research, the algorithm used is Iterative Dichotomiser 3 (ID3). ID3 algorithm is a method of learning that will build a decision tree that becomes model in finding solution of the problem. This algorithm in a greedily procedure searches every possibility of the decision tree. ID3 can be used to process graduation data of UNS bachelor's degree students that can give an significant information categorized in a 12 class structure bases on grade point average and graduation time. This group designed based on attribute gender, faculty, high school origin, enrollment process, parents' salary, and parents' job and this information can be used to determine new students acceptance in University.

From the tree model formed, then it tested by using measuring precision and recall. This testing were done by using data testing. Form the overall data 1066, used 95% for data training. Data testing used 5% from the overall data that is tested ten times randomly. From the testing result, it can be concluded that the implementation of ID3 algorithm on graduation data of UNS bachelor's degree students has average maximum precision 63.96% and average maximum recall 62.47% of data testing in a ten times testing.

Keyword : *Classification, Data mining, Decision tree, Iterative Dichotomiser 3*.

ABSTRAK

Setiap tahun Universitas Sebelas Maret (UNS) melakukan seleksi penenerimaan mahasiswa baru. Proses seleksi tersebut biasanya melalui tiga jalur yaitu Seleksi Nasional Masuk Perguruan Tinggi Negeri (SNMPTN), Seleksi Bersama Masuk Perguruan Tinggi Negeri (SBMPTN), Seleksi Penerimaan Mahasiswa Baru (SPMB). Setiap tahun juga UNS meluluskan mahasiswa melalui empat kali periode wisuda. Dari proses tersebut UNS memiliki data wisuda yang setiap periode pasti bertambah. Data tersebut belum dapat dimanfaatkan secara maksimal, dan data dibiarkan menumpuk begitu saja. Diperlukan teknik untuk mengolah data tersebut agar memberi nilai tambah, yaitu salah satunya dengan memanfaatkan data mining agar mengatasi masalah tersebut.

Pada penelitian ini, algoritma yang digunakan adalah Iterative Dichotomiser 3 (ID3). Algoritma ID3 merupakan suatu metode yang akan membangun sebuah pohon keputusan yang merupakan suatu pemodelan dalam mencari solusi dari persoalan. Algoritma ini melakukan pencarian secara rakus/menyeluruh (greedy) pada semua kemungkinan pohon keputusan. Algoritma ID3 akan digunakan untuk mengolah data kelulusan mahasiswa Strata Satu (S1) UNS sehingga dapat memberikan sebuah informasi penting yang telah ditentukan yaitu berupa 12 kelas berdasarkan IPK dan lama studi. Pengelompokan ini berdasarkan atribut jenis kelamin, fakultas, asal SMA, jalur masuk, gaji orang tua, pekerjaan orang tua dan informasi ini dapat dimanfaatkan UNS untuk pengambilan keputusan penerimaan mahasiswa baru.

Dari model tree yang dibentuk maka dilakukan pengujian data dengan menggunakan pengukuran precision dan recall. Pengujian ini dilakukan menggunakan data testing. Dari data keseluruhan 1066 data, digunakan 95% untuk data training. Data testing yang digunakan 5% dari data keseluruhan yang diuji sebanyak sepuluh kali secara random. Dari hasil pengujian, dapat disimpulkan bahwa implementasi algoritma ID3 pada data kelulusan mahasiswa S1 UNS memiliki rata-rata maksimal precision 63.96% dan rata-rata maksimal recall 62.47% dari sepuluh kali pengujian saat diuji dengan data testing.

Kata kunci : *Data mining, Decision tree, Iterative Dichotomiser 3*, *Klasifikasi.*

1. PENDAHULUAN

Untuk menerima mahasiswa, Universitas Sebelas Maret *(*UNS) melakukan proses penerimaan mahasiswa baru melalui tiga jalur masuk yaitu Seleksi Nasional Masuk Perguruan Tinggi (SNMPTN), Seleksi Bersama Masuk Perguruan Tinggi Negeri (SBMPTN), Seleksi Penerimaan Mahasiswa Baru (SPMB) [1]. Dalam satu tahun, UNS mengadakan empat periode wisuda untuk meluluskan mahasiswa. Setiap periode wisuda, data

kelulusan mahasiswa selalu bertambah, semakin banyak dan menumpuk. Pertumbuhan yang pesat dari akumulasi data kelulusan mahasiswa ternyata menciptakan kondisi yang sering disebut sebagai*"Rich of Data but Poor of Information"* [2] karena data yang terkumpul itu belum digunakan untuk pengambilan keputusan. Kumpulan data itu dibiarkan begitu saja seakan- akan menjadi sebuah "kuburan data *(data tombs)"* [2]. Sebenarnya dari data yang bertumpuk tersebut dapat diperoleh sebuah informasi yang dapat dimanfaatkan oleh UNS untuk mendukung keputusan atau membantu dalam menentukan strategi pendidikan yang lebih baik.

Salah satu teknik yang digunakan untuk mengolah data tersebut agar memberi nilai tambah adalah dengan menggunakan *data mining. Data mining* adalah suatu proses penemuan sebuah informasi baru dalam sekumpulan data yang sangat besar. Penggunaan teknik *data mining* diharapkan dapat memberikan pengetahuan-pengetahuan yang sebelumnya tersembunyi di dalam gudang data sehingga menjadi informasi yang berharga. Berdasarkan fungsinya, *data mining* memiliki beberapa teknik yang dapat digunakan yaitu deskripsi, estimasi, prediksi, klasifikasi, pengklusteran, asosiasi.

Pada kasus data wisuda, dapat digunakan teknik klasifikasi karena teknik ini mampu membagi data yang sangat besar ke dalam kelas-kelas yang berbeda, misalnya klasifikasi kelulusan mahasiswa sehingga memudahkan bagi pihak UNS dalam pengambilan keputusan. Klasifikasi merupakan suatu proses yang mengklasifikasikan data ke dalam kelas-kelas yang berbeda menurut model klasifikasi yang ditetapkan [3]. Salah satu metode dalam klasifikasi *data mining* adalah *decision tree* (pohon keputusan). *Decision tree* merupakan metode klasifikasi yang paling populer digunakan. Selain karena pembangunannya relatif cepat, hasil dari model yang dibangun mudah untuk dipahami [4]. *Decision tree* merupakan salah satu metode klasifikasi yang menggunakan representasi struktur pohon (*tree*) di mana setiap *node* merepresentasikan atribut, cabangnya merepresentasikan nilai dari atribut, dan daun merepresentasikan kelas. *Node* yang paling atas dari *decision tree* disebut sebagai *root*.

Decision tree mempunyai banyak algoritma salah satunya adalah *Iterative Dichotomiser 3* (ID3). Penelitian yang pernah dilakukan menunjukkan bahwa algoritma ID3 merupakan algoritma yang terbaik dan akurat dibandingkan algoritma klasifikasi yang lain [5]. Algoritma ID3 melakukan pencarian secara menyeluruh (*greedy*) pada semua kemungkinan (*decision tree*) karena algoritma ini berusaha membangun *decision tree* secara *top-down* (dari atas ke bawah), dengan mengevaluasi semua atribut yang ada dengan menggunakan suatu ukuran statistik (yang banyak digunakan adalah *information gain*) untuk mengukur efektivitas suatu atribut dalam mengklasifikasikan kumpulan sampel data.

Pada penelitian ini akan dilakukan *data mining* untuk membantu dan memudahkan UNS pada pengambilan keputusan dalam penerimaan mahasiswa baru dengan cara mengklasifikasikan data induk mahasiswa dan data kelulusan mahasiswa, menggunakan metode *decision tree* algoritma ID3. Tujuan yang ingin dicapai dalam penelitian ini adalah untuk mengimplementasi algoritma ID3 pada pengolahan data kelulusan mahasiswa S1 UNS untuk pengambilan keputusan penerimaan mahasiswa baru.

2. DASAR TEORI

2.1 *Data Mining*

2.1.1 Pengertian *Data Mining*

Data mining adalah suatu proses menemukan hubungan yang berarti, pola, dan kecenderungan dengan memeriksa dalam sekumpulan besar data yang tersimpan dalam penyimpanan dengan menggunakan teknik pengenalan pola seperti teknik statistik dan matematika.[6]

2.1.2 Pengelompokan *Data Mining*

Data mining dibagi menjadi beberapa kelompok berdasarkan tugas yang dapat dilakukan, yaitu [6] :

1. Deskripsi: digunakan untuk menggambarkan pola dan kecenderungan yang terdapat dalam data.
2. Estimasi: model menggunakan *record* lengkap yang menyediakan nilai dari variabel *target* sebagai nilai prediksi.
3. Prediksi: prediksi nilai dari hasil akan ada di masa mendatang.
4. Klasifikasi: contoh kasusnya adalah penggolongan pendapatan dapat dipisahkan dalam tiga kategori, yaitu pendapatan tinggi, pendapatan sedang, dan pendapatan rendah.
5. Pengklusteran: merupakan pengelompokan *record*, pengamatan, atau memperhatikan dan membentuk kelas objek-objek yang memiliki kemiripan.
6. Asosiasi: Tugas asosiasi dalam data *mining* adalah menemukan atribut yang muncul dalan satu waktu. Dalam dunia bisnis lebih umum disebut analisis keranjang belanja.

2.1.3 Tahap-tahap *Data Mining*

Berikut adalah tahapan dalam *data mining* [7] :

1. *Data Selection*

   Pemilihan (seleksi) data dari sekumpulan data operasional. Data hasil seleksi akan digunakan untuk proses *data mining,* dan disimpan dalam suatu berkas dan terpisah dari basis data operasional.

2. *Pre-processing/Cleaning*

   Proses *cleaning* dilakukan dengan menghilangkan *noise*, membuang duplikasi data, memeriksa data yang tidak konsisten, dan

memperbaiki kesalahan pada data, seperti kesalahan cetak (tipografi).

3. *Transformation Data*

Tahap ini merupakan proses transformasi pada data yang telah dipilih, sehingga data tersebut sesuai untuk proses *data mining*.

4. *Data mining*

Data mining merupakan proses mencari pola atau informasi yang sangat menarik dalam data terpilih dengan menggunakan teknik atau metode tertentu.

5. *Interpretation/Evalution*

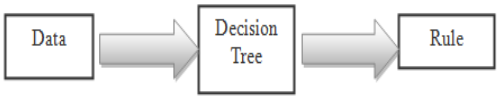
Tahap ini mencakup pemeriksaan apakah pola atau informasi yang ditemukan bertentangan dengan fakta atau hipotesis yang ada sebelumnya.

2.2 Klasifikasi

Klasifikasi merupakan suatu pekerjaan menilai objek data untuk memasukkannya ke dalam kelas tertentu dari sejumlah kelas yang tersedia.

2.3 *Decision Tree* (Pohon Keputusan)

Decision tree merupakan metode yang mengubah data menjadi pohon keputusan *(decision tree)* dan aturan-aturan keputusan *(rule)* yang dijelaskan pada Gambar 2.2 [8] :



Gambar 2.2 Konsep *Decision Tree* [8]

Pada *decision tree* terdapat 3 jenis *node*, yaitu:

a. *Root node*: merupakan *node* paling atas, pada *node* ini tidak ada *input* dan bisa tidak mempunyai *output* atau mempunyai *output* lebih dari satu.
b. *Internal node*: merupakan *node* percabangan, pada *node* ini hanya terdapatsatu *input* dan mempunyai *output* minimal dua.
c. *Leaf node* atau *terminal node*: merupakan *node* akhir, pada *node* ini hanya terdapat satu *input* dan tidak mempunyai *output*.

2.4 Algoritma ID3

Algoritma ID3 merupakan algoritma *decision tree learning* (algoritma pembelajaran pohon keputusan) yang paling dasar. Algoritma ini melakukan pencarian secara rakus/menyeluruh (*greedy*) pada semua kemungkinan pohon keputusan [9].

Untuk membangun *decision tree,* algoritma ini mengevaluasi semua atribut yang ada dengan menggunakan suatu ukuran statistik (yang banyak digunakan adalah *information gain*), untuk mendapatkan nilai *information gain* yang atributnya akan dipilih sebagai akar maka nilai *entropy* dihitung terlebih dahulu. Berikut adalah penjelasan mengenai nilai *Entropy* dan *Information Gain.*

2.4.1 *Entropy*

Entropy adalah ukuran dari teori informasi yang dapat mengetahui karakteristik dari *impurity* dan *homogenity* dari kumpulan data..

$$Entropy(S) = \sum_{i=1}^{n} -p_i log_2 p_i \quad (1)$$

Dimana:
S: himpunan kasus
n: jumlah partisi S
p_i: proporsi dari Si terhadap S

Dari nilai *entropy* tersebut kemudian dihitung nilai *information gain* (IG) masing-masing atribut.

2.4.2 *Information Gain*

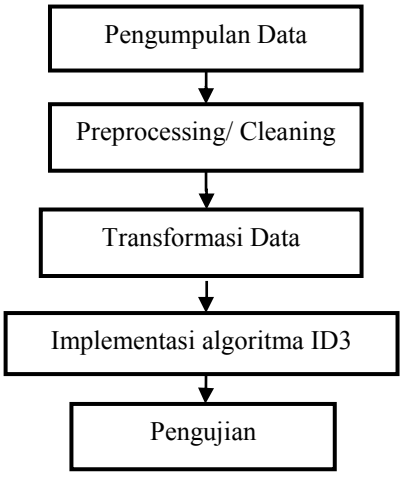
Setelah mendapat nilai *entropy* untuk suatu kumpulan data, maka kita dapat mengukur efektivitas suatu atribut dalam mengklasifikasikan data. Ukuran efektifitas ini disebut *information gain*. Secara matematis, *infomation gain* dari suatu atribut A, dituliskan sebagai berikut:

$$Gain(S, A) = Entropy(S) - \sum_{i=1}^{n} \frac{|S_i|}{|S|} * Entropy(S_i) \quad (2)$$

Dimana:
S : himpunan kasus
A: atribut
i : menyatakan suatu nilai yang mungkin untuk atribut A
|Si|: jumlah kasus pada partisi ke-i
|S|: jumlah kasus dalam S
Entropy (Si): *entropy* untuk sampel-sampel yang memilki nilai i

3. METODOLOGI PENELITIAN

Penelitian akan dilaksanakan berdasarkan tahap penelitian yang ditunjukkan pada Gambar 3.1.



Gambar 3.1 Proses Tahap Penelitian

3.1 Pengumpulan Data

Dalam tahapan ini, terdapat dua teknik yang digunakan:

3.1.1 Studi literatur

Metode ini dilaksanakan dengan melakukan studi kepustakaan untuk mendapatkan data literatur tambahan melalui membaca buku-buku, jurnal, karya ilmiah maupun artikel-artikel yang terkait dengan *data mining,* teknik klasifikasi*, decision tree* dan algoritma ID3.

3.1.2 Pengumpulan Data

Penelitian ini menggunakan data sekunder berupa data kelulusan mahasiswa S1 UNS dari tahun 2005 sampai tahun 2013 yang terdiri dari 19.942 data. Atribut yang digunakan dalam data kelulusan mahasiswa ini berdasarkan kemungkinan bahwa atribut tersebut memiliki pola yang berpengaruh besar terhadap kelulusan mahasiswa, atribut ini meliputi jenis kelamin, fakultas, asal SMA, jalur masuk, gaji orang tua, pekerjaan orang tua, lama studi dan IPK (Indek Prestasi Komulatif).

3.2 *Preprocessing*/ *Cleaning*

Proses menghapus atribut yang tidak terpakai dan isian yang kosong sehingga data yang akan diolah benar-benar relevan dengan yang dibutuhkan.

3.3 Transformasi Data

Proses tranformasi data yang merupakan proses menggabungkan data yang diperoleh dan mengubahnya menjadi satu format yang sama untuk mempermudah proses *mining*.

3.4 Implementasi algoritma ID3

Secara umum langkah algoritmaID3 untuk membangun pohon keputusan adalah sebagai berikut:

1. Pilih atribut sebagai akar
2. Buat cabang untuk tiap nilai
3. Bagi kasus dalam cabang
4. Ulangi proses untuk setiap cabang sampai semua kasus pada cabang memiliki kelas yang sama.

Untuk memilih atribut sebagai akar, didasarkan pada nilai *gain* tertinggi dari masing-masing atribut yang ada. Untuk menghitung *gain* digunkan rumus seperti dalam persamaan (1) . Sementara itu, perhitungan nilai *entropy* dapat dilihat pada persamaan (2).

3.5 Pengujian

Setelah melakukan proses *mining* data maka akan dihasilkan model pohon keputusan. Model yang telah terbentuk itu akan diuji coba dengan data *testing* yang telah ditentukan. Pengujian ini menggunakan data *testing* dengan komposisinya 95% data *training* dan 5% data *testing* lalu diuji sebanyak sepuluh kali dengan data yang diambil secara *random*. Randomisasi dilakukan secara proporsional yaitu jumlah *records* data dari setiap kelas akan diacak 5% data *testing*, yang sisanya dijadikan data *training*. Pengujian ini menggunakan pengukuran *precision* dan *recall.* Penjelasan tentang pengukuran *precision* dan *recall* dapat di lihat pada Tabel 3.1 dan perhitungan di berikut:

Tabel 3.1 Perhitungan *Precision* dan *Recall*

		Predicted Class	
		Yes	No
Actual Class	Yes	a (TP)	b (FN)
	No	c (FP)	d (TN)

Keterangan:
a: TP (true positive)
b: FN (false negative)
c: FP (false positive)
d: TN (true negative)

Precision dihitung dengan rumus:

$$Precision\ (p) = \frac{a}{a+c} \qquad (3)$$

Recall dapat dihitung dengan rumus:

$$Recall(r) = \frac{a}{a+b} \qquad (4)$$

4. PEMBAHASAN

Pada implementasi algoritma ID3 ini melalui beberapa tahapan. Dimulai dari Deskripsi Data, Transformasi Data, Implementasi Algoritma ID3, dan Pengujian.

4.1 Deskripsi Data

Penelitian ini menggunakan data sekunder berupa data kelulusan mahasiswa S1 UNS dari tahun 2005 sampai tahun 2013 dengan jumlah data 19.942 mahasiswa. Setelah melakukan proses pembersihan data, memperoleh sejumlah 1066 data yang terdiri dari berbagai atribut yaitu lama studi, IPK, jenis kelamin, jalur masuk, fakultas, asal SMA, pekerjaan orang tua, gaji orang tua.

4.2 Transformasi Data

Setelah proses *cleaning* data, dilakukan proses transformasi data, sehingga data tersebut sesuai untuk proses *data mining*. Tabel tranformasi data atribut predikator yang disajikan pada Tabel 4.1

No	Atribut Predikator	Kategori	Alias
1.	**Jenis Kelamin**	Laki-laki	1
		Perempuan	2
2.	**Fakultas**	Kedokteran	1
		Mipa	2
		Ekonomi	3
		Hukum	4
		Pertanian	5
		FSSR	6
		FISIP	7
		FKIP	8
		Teknik	9

Tabel 4.1 Transformasi Data Atribut Predikator

Tabel 4.1 Transformasi Data Atribut Predikator (Lanjutan)

No	Atribut Predikator	Kategori	Alias
3.	Asal SMA	SKA (Surakarta)	1
		Solo raya	2
		Jawa tengah	3
		Jawa	4
		Luar jawa	5
		Luar negeri	6
4.	Jalur masuk	SPMB	1
		Swadana	2
		PMDK	3
		Transfer	4
		PBUS	5
		Tugas Belajar	6
5.	Gaji orang tua	kurang dari Rp. 500.000	1
		Rp. 500.000 - Rp. 1.000.000	2
		Rp. 1.000.000 - Rp. 2.500.000	3
		Rp. 2.500.000 - Rp. 5.000.000	4
		Rp. 5.00.000 - Rp. 7.500.000	5
		Rp. 7.500.000 - Rp.10.000.000	6
		Lebih dari Rp. 10.000.000	7
6.	Pekerjaan orang tua	Pensiunan Pegawai Negeri/ABRI	1
		Pensiun Pegawai Swasta	2
		Guru/Dosen Negeri	3
		Guru/Dosen Swasta	4
		Pegawai Swasta bukan Guru/Dosen	5
		Pegawai Negeri bukan Guru/Dosen	6
		Buruh/orang yang bekerja dengan tenaga fisik saja	7
		Ahli profesional yang hanya bekerja secara perorangan	8
		Abri	9
		Pedagang/Wiraswasta	10
		Petani/Nelayan	11
		Lain-lain	12

Atribut Respon	Kategori	Interval Kelas
IPK dan Lama Studi	Lama studi 3,5 – 4 tahun dan IPK 3,51 – 4,00	A1
	Lama studi 3,5 – 4 tahun dan IPK 3,01 – 3,50	A2
	Lama studi 3,5 – 4 tahun dan IPK 2,51 – 3,00	A3
	Lama studi 3,5 – 4 tahun dan IPK 2,00 – 2,50	A4
	Lama studi >4 – 5 tahun dan IPK 3,51 – 4,00	B1

Tabel tranformasi data atribut respon yang disajikan pada Tabel 4.2

Tabel 4.2 Transformasi Data Atribut Respon

Tabel 4.2 Transformasi Data Atribut Respon (Lanjutan)

Atribut Respon	Kategori	*Interval* Kelas
IPK dan Lama Studi	Lama studi >4 – 5 tahun dan IPK 3,01 – 3,50	B2
	Lama studi >4 – 5 tahun dan IPK 2,51 – 3,00	B3
	Lama studi >4 – 5 tahun dan IPK 2,00 – 2,50	B4
	Lama studi > 5 tahun dan IPK 3,51 – 4,00	C1
	Lama studi > 5 tahun dan IPK 3,01 – 3,50	C2
	Lama studi > 5 tahun dan IPK 2,51 – 3,00	C3
	Lama studi > 5 tahun dan IPK 2,00 – 2,50	C4

4.3 Implementasi algoritma ID3

Untuk mengambarkan pembentukan *decision tree* dengan menggunakan algoritma ID3, berikut ini adalah langkah-langkahnya:

- Pilih atribut sebagai akar
- Buat cabang untuk tiap nilai
- Bagi kasus dalam cabang
- Ulangi proses untuk setiap cabang sampai semua kasus pada cabang memiliki kelas yang sama

Berikut ini diberikan contoh sebagai kasus menggunakan lima data kelulusan mahasiswa S1 UNS yang disajikan pada Tabel 4.3

Tabel 4.3 Contoh Data Kelulusan Mahasiswa S1

No	1	2	3	4	5
JK	P	P	L	P	L
Fakultas	Kedokteran	Kedokteran	FISIP	FSSR	Ekonomi
Asal SMA	Ska	Ska	Jawa tengah	Ska	Ska
Jalur masuk	Swadana	spmb	spmb	spmb	spmb
Gaji ortu	1.000.000 - 2.500.000	1.000.000 - 2.500.000	1.000.000 - 2.500.000	1.000.000 - 2.500.000	Kurang dari Rp.500.000
Pekerjaan ortu	Guru/ Dosen Negeri	Pegawai Negeri Bukan Guru/ Dosen	Buruh	Guru/ Dosen Negeri	Lain-lain
IPK	3.51	3.01	3.51	3.62	3.02
Lama studi	3.5	4	4.5	5.5	5.5

Node	Atribut	Nilai atribut	Jml kasus	A1	A2	B1	C1	C2	Entropy	Gain
1	Total		5	1	1	1	1	1	2.32	
	A									0.97
		1	2	0	0	1	0	1	1	
		2	3	1	1	0	1	0	1.584 96	
	B									1.92
		1	2	1	1	0	0	0	1	
		2	1	0	0	0	0	1	0	
		6	1	0	0	0	1	0	0	
		7	1	0	0	1	0	0	0	
	C									0.72
		1	4	1	1	0	1	1	2	
		3	1	0	0	1	0	0	0	
	D									0.72
		1	4	0	1	1	1	1	2	
		2	1	1	0	0	0	0	0	
	E									0.72
		1	1	0	0	0	0	1	0	
		3	4	1	1	1	1	0	2	
	F									1.92
		3	2	1	0	0	1	0	1	
		6	1	0	0	1	0	0	0	
		7	1	0	1	0	0	0	0	
		12	1	0	0	0	0	1	0	

Kemudian data-data pada Tabel 4.3 dilakukan transformasi sesuai dengan Tabel 4.1 dan 4.2, dan data hasil transformasi ditunjukkan pada Tabel 4.4

Tabel 4.4 Hasil Data Transformasi

A	B	C	D	E	F	G
2	1	1	2	3	3	A1
2	1	1	1	3	6	A2
1	7	3	1	3	7	B1
2	6	1	1	3	3	C1
1	2	1	1	1	12	C2

Keterangan :
A: Jenis Kelamin
B: Fakultas
C: Asal SMA
D: Jalur Masuk
E: Gaji orang tua
F: Pekerjaan orang tua
G: Kelas (Target)

Dari data-data pada Tabel 4.4 akan dibangun sebuah *classifier* berdasarkan atribut jenis kelamin, fakultas, asal SMA, jalur masuk, gaji orang tua, dan pekerjaan orang tua. Ada lima kelas yaitu A1, A2, B1, C1, C2.

Berikut ini penjelasan mengenai tiap-tiap langkah dalam pembentukan pohon keputusan dengan menggunakan algoritma ID3 untuk menyeselaikan permasalahan pada Tabel 4.3

a. Menghitung jumlah kasus, jumlah kasus untuk kelas A1, kelas A2, kelas B1, kelas C1, kelas C2, dan *entropy* dari semua kasus dan kasus yang dibagi berdasarkan atribut A, B, C, D, E, F. Setelah itu, lakukan perhitungan *gain* untuk tiap-tiap atribut. Hasil perhitungan ditunjukkan oleh Tabel 4.5

Tabel 4.5 Perhitungan *Node* 1

Baris **TOTAL** kolom *entropy* pada Tabel 4.5 dihitung dengan persamaan (2) sebagai berikut:

$$Entropy(Total) = (-\frac{1}{5}\log_2(\frac{1}{5})) + (-\frac{1}{5}\log_2(\frac{1}{5})) + (-\frac{1}{5}\log_2(\frac{1}{5})) + (-\frac{1}{5}\log_2(\frac{1}{5})) + (-\frac{1}{5}\log_2(\frac{1}{5})$$

$$Entropy(Total) = 2.32$$

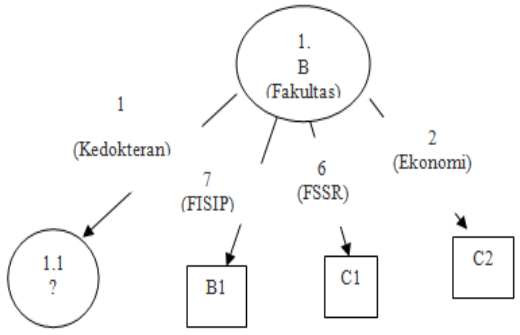
Sementara itu, nilai g*ain* pada baris A dihitung dengan menggunakan persamaan (1) sebagai berikut:

$$Gain(Total, A) = Entropy(Total) \sum_{i=1}^{n} \frac{|A|}{|Total|} * Entropy(A)$$

$$Gain(Total, A) = 2.32 - ((\frac{2}{5} * 1) + (\frac{3}{5} * 1.58496)) = 0.97$$

Untuk nilai *gain* atribut yang lain juga dihitung seperti nilai *gain* atribut A.

Dari hasil pada Tabel 4.5 dapat diketahui bahwa ada dua atribut memiliki *gain* tertinggi yaitu B (Fakultas) dan F (Pekerjaan ortu) yaitu sebesar 1.92. Pada kasus ini, memilih atribut B untuk menjadi *node* akar. Ada empat nilai atribut dari B, yaitu 1, 2, 6, 7. Dari keempat nilai atribut tersebut, nilai atribut 2, 6, 7 sudah mengklasifikasi kasus menjadi kelas C2, C1, dan B1, sehingga tidak perlu dilakukan perhitungan lebih lanjut, tetapi untuk nilai atribut 1 masih perlu dilakukan perhitungan lagi. Dari hasil tersebut digambarkan pohon keputusan pada *node* 1tampak seperti Gambar 4.1



Gambar 4.1 Pohon Keputusan Hasil Perhitungan *Node* 1

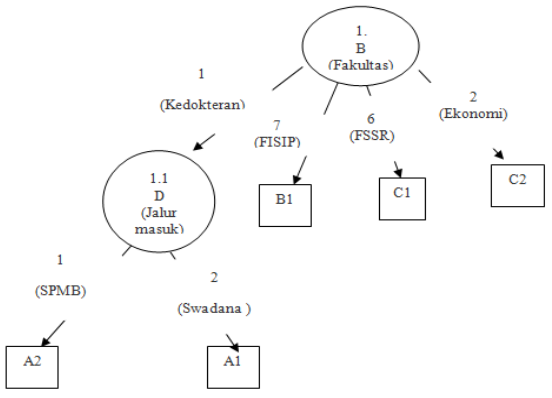
b. Menghitung jumlah kasus, jumlah kasus untuk kelas A1, kelas A2, kelas B1, kelas C1, kelas C2, dan *entropy* dari semua kasus dan kasus yang dibagi berdasarkan atribut A, C, D, E, F yang dapat dijadikan sebagai *node* akar dari nilai atribut Setelah itu, dilakukan perhitungan *gain*

untuk tiap-tiap atribut. Hasil perhitungan ditunjukkan oleh Tabel 4.6

Node	Atribut	Nilai atribut	Jml Kasus	A1	A2	Entropy	Gain
1.1			2	1	1	1.00	0
	A						
		1	0	0	0	0	
		2	2	1	1	1	
	C						0
		1	2	1	1	1	
		3	0	0	0	0	
	D						1
		1	1	0	1	0	
		2	1	1	0	0	
	E						1
		1	0	0	0	0	
		3	2	1	1	1	
	F						1
		3	1	1	0	0	
		7	0	0	0	0	
		6	1	0	1	0	
		12	0	0	0	0	

Tabel 4.6 Perhitungan *Node* 1.1 untuk B-1

Dari hasil pada Tabel 4.6 dapat diketahui bahwa ada tiga atribut dengan *gain* tertinggi adalah D (Jalur masuk), E (Gaji ortu) dan F (Pekerjaan ortu) yaitu sebesar 1.00. Sehingga dipilih salah satu atrubut tersebut untuk menjadi *node* akar dan dikasus ini memilih D menjadi *node* cabang dari nilai atribut 1 (Kedokteran). Ada dua nilai atribut dari D, yaitu 1 dan 2. Dari kedua nilai atribut tersebut, nilai atribut 1 dan 2 sudah mengklasifikasikan kasus menjadi
kelas A2 dan A1, sehingga tidak perlu dilakukan perhitungan lebih lanjut. Pohon keputusan yang terbentuk dari kasus ini disajikan pada Gambar 4.3

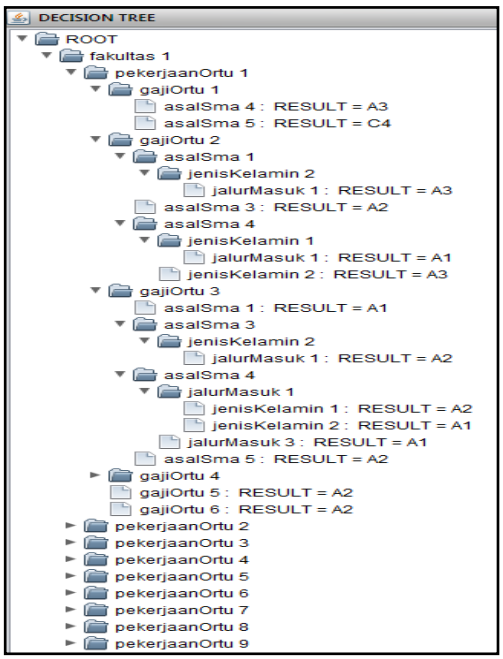


Gambar 4.2 Pohon Keputusan Hasil Perhitungan *Node* 1.1

Gambar 4.2 menunjukkan hasil akhir dari proses pembentukan *tree*. Setelah selesai pembentukan *tree*, selanjutnyaakan dilakukan proses pembuatan aturan berdasarkan *tree* yang terbentuk. Dan aturan dari *tree* di atas adalah sebagai berikut:

- Jika fakultas Kedokteran , lalur masuk SPMB maka lama studi 3,5 – 4 tahun dan IPK 3,01 – 3,50.
- Jika fakultas Kedokteran, jalur masuk Swadana maka lama studi 3,5 – 4 tahun dan IPK 3,51 – 4,00.
- Jika fakultas Ekonomi maka lama studi > 5 tahun dan IPK 3,01 – 3,50.
- Jika fakultas FSSR maka lama studi > 5 tahun dan IPK 3,51 – 4,00.
- Jika fakultas FISIP maka lama studi > 4 – 5 tahun dan IPK 3,51 – 4,00.

Dari langkah-langkah pada tahap ini, akan diimplementasikannya ke dalam sistem sehingga menghasilkan model pohon keputusan. Implementasi ini menggunakan bahasa pemograman Java jdk 6 dengan *editor* Netbean 7.1, *database* MySql. Gambar 4.3 tampilan potongan *tree* yang terbentuk dari sistem setelah menjalankan proses *mining*.



Gambar 4.3 Potongan Pohon Keputusan

4.4 Pengujian Model

Model pohon keputusan yang telah terbentuk diuji dengan menggunakan data *testing*. Pengujian model dilakukan dengan membandingkan hasil model pohon keputusan yang terbentuk dengan data yang diuji. Jika salah satu atau seluruh nilai atribut dan kelas diperoleh sama, maka hasil klasifikasi dinyatakan sesuai. Sebaliknya, jika salah satu atau semua nilai atribut sama tetapi kelas berbeda maka hasil klasifikasi dinyatakan tidak sesuai. Dari seluruh pengujian yang telah dilakukan menunjukkan bahwa hasil yang didapatkan tidak memiliki perbedaan yang signifikan. Berikut ini contoh hasil pengujian dengan menggunakan data *testing* yang diuji sebanyak sepuluh kali yang disajikan pada Tabel 4.7.

Tabel 4.7 Hasil Pengujian Menggunakan Data *Testing* yang Diuji Sebanyak Sepuluh Kali

Percobaan ke-	Jumlah data	Sesuai	Tidak sesuai	Maksimal *Precision*	Maksimal *Recall*
1	54	12	32	50.0	53.9
2	54	16	27	87.5	58.3
3	54	16	26	63.6	58.3
4	54	14	34	50.0	50.0
5	54	19	22	63.6	63.6
6	54	14	27	78.6	68.75
7	54	16	24	56.25	69.2
8	54	17	25	60.0	64.3
9	54	15	31	50.0	58.3
10	54	16	23	80.0	80.0
Hasil rata-rata				**63.96 %**	**62.47%**

Dari Tabel 4.8 diperoleh hasil pengujian menggunakan data *testing* yang telah diuji sebanyak sepuluh kali. Nilai maksimal *precision* dan maksimal *recall* yang diperoleh dari pengujian pertama sampai pengujian kesepuluh terdapat dalam kelas A2 dan B3, karena pada jumlah data *testing* sebanyak 54 data yang berasal dari kelas A2 dan B3 memiliki jumlah *records* yang paling tinggi dibandingkan kelas lain yaitu A2 yang memiliki 17 *records* dan B2 yang memiliki 16 *records*. Sehingga, nilai maksimal *precision* dan maksimal *recall* yang tertinggi ada di kelas A2 dan B2. Hasil rata-rata maksimal *precision* memiliki 63.96% dan rata-rata maksimal *recall* 62.47%.

5. KESIMPULAN

Berdasarkan hasil eksperimen yang telah dilakukan pada penelitian ini, maka penulis menarik kesimpulan, antara lain: [1] Model pohon keputusan dengan algoritma ID3 dapat digunakan untuk memberikan informasi kepada UNS khususnya terkait pengambilan keputusan pada saat penerimaan mahasiswa baru. [2] Penentuan data *training* sangat menentukan tingkat akurasi *tree* yang dibuat. Besar persentase kebenaran *tree* sangat dipengaruhi oleh data *training* yang digunakan untuk membangun model *tree* tersebut. [3] Hasil pengujian data *testing* yang diuji menggunakan algoritma ID3 memiliki nilai rata-rata maksimal *precision* 63.96% dan rata-rata maksimal *recall* 62.47% dari sepuluh kali pengujian dengan data yang diperoleh secara *random*.

Penulis menyarankan sebaiknya penelitian selanjutnya dapat menggunakan algoritma lain atau menggabungkan algoritma ID3 dengan algoritma lain, kemudian hasil pengujian yang diperoleh dibandingkan demi mendapatkan hasil penelitian yang lebih baik.

6. DAFTAR PUSTAKA

[1] Sistem Informasi SPMB. Online Universitas Sebelas Maret. Akses pada tanggal 1 Juni, 2014. http://www.spmb.uns.ac.id

[2] Pramudiono, I. (2003). *Pengantar Data Mining: Menambang Permata Pengetahuan di Gunung Data.* Ilmu Komputer.com.

[3] Hamidah, I. (2012). *Aplikasi Data Mining Untuk Memprediksi Masa Studi Mahasiswa Menggunakan Algoritma C4.5.* Bandung.

[4] Kusrini & Luthfi, E. (2009). *Algoritma Data Mining.* Andi: Yogyakarta.

[5] Wawan & Hendayun, M. (2013). *Kinerja Algoritma Data Mining Decision Tree (ID3, C4.5, C5.0), Naïve Bayes, SVM dan kNN untuk Klasifikasi Email Spam dan Non-Spam.* Bandung.

[6] Larose, D. T. (2005). *Discovering Knowledge in Data: An Introduction to Data mining.* John Willey and Sons, Inc.

[7] Fayyad, U. (1996). *Advances in Knowledge Discovery and Data Mining.* MIT Press.

[8] Basuki, A. & Syarif, I. (2003). *Decision Tree.* Surabaya.

[9] Wahyudin. (2009). Metode Iterative Dichotomizer 3 (ID3) Untuk PenyeleksianPenerimaan Mahasiswa Baru. *Jurnal Pendidikan Teknologi Informasi Dan Komunikasi (Ptik)* , Vol.2, No.2.

Pemodelan Penjadwalan Multilevel Feedback Queue Menggunakan Dynamic Time Quantum Pada Kasus Pemesanan Makanan di Restoran

Tri Wahyu Prasetyo
Jurusan Informatika
Universitas Sebelas Maret
Jl. Ir. Sutami No. 36 A Surakarta
triwahyuprasetyo@gmail.com

Wiharto
Jurusan Informatika
Universitas Sebelas Maret
Jl. Ir. Sutami No. 36 A Surakarta
wi_harto@yahoo.com

Afrizal Doewes
Jurusan Informatika
Universitas Sebelas Maret
Jl. Ir. Sutami No. 36 A Surakarta
afrizal.doewes@gmail.com

ABSTRAK

Dalam hal pelayanan di restoran, lamanya waktu tunggu pesanan disajikan, kesalahan urutan memasak dan tertukarnya pesanan adalah masalah yang membuat pelayanan menjadi tidak maksimal. Untuk menghindari hal tersebut perlu diterapkan sistem penjadwalan memasak yang efisien. Pada penelitian ini metode penjadwalan yang diterapkan adalah Multilevel Feedback Queue menggunakan Dynamic Time Quantum (MLFQ DTQ). Algoritma MLFQ terdiri dari beberapa queue, yang masing-masing memiliki time quantum dan algoritma sendiri (Round Robin atau FCFS). Algoritma ini menjadwalkan pesanan dengan cara mengelompokkan pesanan berdasarkan lama waktu memasaknya ke dalam queue sesuai dengan besar kecilnya time quantum queue. Kemudian dilakukan penjadwalan sesuai algoritma pada masing-masing queue secara berurutan mulai dari queue dengan prioritas tertinggi ke prioritas yang lebih rendah.

Hasil dari penelitian ini adalah, dari 19 percobaan seluruhnya menyatakan bahwa rata-rata response time MLFQ lebih cepat dibanding FCFS, baik pada parameter koki berjumlah 3, 4, maupun 5. Dari 19 percobaan yang sama, pada koki berjumlah 3, 4 dan 5 masing-masing terdapat 15, 14 dan 17 percobaan yang menunjukkan rata-rata response time MLFQ DTQ lebih cepat dibanding MLFQ. Sehingga dapat disimpulkan bahwa algoritma MLFQ DTQ dominan memiliki rata-rata response time yang lebih cepat daripada algoritma MLFQ.

Kata kunci : ***Dynamic Time Quantum, Multilevel Feedback Queue, Penjadwalan, Queue.***

1. PENDAHLUAN

Persaingan dalam bisnis kuliner meningkat seiring dengan bertambahnya restoran-restoran besar di kota-kota di Indonesia. Setiap pengusaha restoran berlomba-lomba dalam mendapatkan costumer sebanyak-banyaknya dengan meningkatkan kualitas produk dan pelayanan. Dalam menghadapi persaingan, setiap restoran harus mempunyai keunggulan dibanding dengan restoran lain seperti dari segi suasana, cita rasa, fasilitas, pelayanan dan kecepatan penanganan masalah yang dihadapi oleh konsumen sehingga konsumen merasa puas, dihargai, menjadi loyal terhadap restoran tersebut [1].

Namun dalam pelaksanaannya, kadang muncul beberapa masalah yang membuat pelayanan menjadi tidak maksimal. Seperti waktu menunggu pesanan yang lama, terjadinya penumpukan pesanan di dapur yang berakibat kesalahan urutan memasak dan tertukarnya pesanan. Untuk menghindari penumpukan pemesanan makanan di dapur/koki, salah satu solusinya adalah dengan menerapkan sistem antrian ataupun penjadwalan untuk semua pesanan yang masuk [2].

Pada umumnya penjadwalan pada restoran dilakukan berdasarkan aturan yang datang pertama dilayani pertama atau *First Come First Serve* (FCFS). Namun algoritma FCFS memiliki kelemahan yaitu jika proses yang sedang dikerjakan memakan waktu yang lama maka akan berakibat meningkatnya *response time* dan *waiting time* proses selanjutnya. Selain FCFS terdapat algoritma penjadwalan lain yaitu *Multilevel Feedback Queue* (MLFQ). Algoritma MLFQ menjadwalkan dengan cara memasukkan proses ke beberapa queue. Masing-masing queue memiliki algoritma sendiri seperti *Round Robin* atau FCFS. Algoritma MLFQ biasa digunakan pada penjadwalan CPU.

Terdapat beberapa penelitian terkait implementasi MLFQ pada penjadwalan CPU. Penelitian [3] yang membahas tentang perbedaan *Multilevel Queue* (MLQ) dan MLFQ pada salah satu kesimpulannya menyebutkan bahwa MLFQ memberikan *performance* yang bagus seperti pada *Shortest Job First* (SJF) untuk proses pendek yang interaktif dan fair bagi proses panjang yang membutuhkan kerja CPU secara intensif. Penelitian [4] yang membahas tentang perbaikan algoritma *Multilevel Feedback Queue* pada kasus penjadwalan CPU dengan menambahkan *sorting* pada *queue Round Robin*, disimpulkan bahwa cara yang diusulkan memberikan hasil yang lebih baik dibandingkan cara MLFQ tradisional.

Penelitian [5] tentang teknik penjadwalan MLFQ menggunakan M/M/c *queue* pada *grid computing*, menyimpulkan cara yang diusulkan bekerja lebih baik pada kebanyakan *scenario* ketika dibandingkan dengan algoritma FCFS dan PBS_PRO. Penelitian [6] tentang optimalisasi MLFQ pada penjadwalan CPU dengan mengimplementasikan *Dynamic Time Quantum* disimpulkan bahwa *starvation* pada proses yang panjang dapat dikurangi serta berkurangnya *turnaround time* dan *waiting time* yang berdampak pada eksekusi yang lebih cepat. Beberapa penelitian tersebut menunjukkkan bahwa penggunaan algoritma MLFQ pada kasus penjadwalan CPU telah terbukti memiliki keunggulan.

Penjadwalan CPU bersifat *preemtive* yaitu proses dapat diinterupsi bila sudah mencapai batas *time quantum*. Sedangkan pada kasus penjadwalan memasak adalah *non preemptive* yaitu proses yang sedang berjalan tidak dapat diinterupsi. Selain itu penjadwalan CPU tidak mengenal *fairness* sedangkan pada penjadwalan memasak, *fairness* menjadi salah satu hal yang diperhatikan. Yang dimaksud *fairness* disini adalah tingkat kecocokan antara urutan mulai

memasak maupun urutan selesai memasak yang dihasilkan algoritma terhadap urutan yang seharusnya(urutan yang dihasilkan FCFS).

Pada penelitian [2] tentang implementasi algoritma *Multilevel Feedback Queue* pada kasus pemesanan makanan di restoran, disimpulkan pada algoritma MLFQ terdapat perbaikan dari segi total waktu jika dibanding dengan algoritma FCFS yang biasa diterapkan di restoran. Namun pembahasan yang dilakukan pada penelitian ini tidak mengulas tentang *response time* dan *fairness*. Sehingga penelitian tersebut belum bisa menjawab bagaimana performa *response time* dan *fairness* algoritma MLFQ pada kasus penjadwalan memasak.

Atas dasar hal tersebut, penulis bermaksud memodelkan penjadwalan *Multilevel Feedback Queue* menggunakan *Dynamic Time Quantum* pada kasus penjadwalan memasak di restoran. Alasan digunakannya DTQ pada MLFQ adalah karena telah dibuktikan pada penelitian sebelumnya bahwa dengan digunakannya DTQ berdampak pada eksekusi yang lebih cepat pada penjadwalan CPU. Selain itu penelitian [7] menyimpulkan bahwa *Round Robin* dengan *dynamic quantum* lebih baik daripada *static quantum*.

2. DASAR TEORI

2.1 Sistem Antrian

Antrian terjadi disebabkan oleh jumlah kebutuhan akan layanan yang melebihi kapasitas fasilitas layanan, sehingga pengguna fasilitas layanan yang tiba tidak bisa segera mendapat layanan dikarenakan kesibukan layanan [2]. Disiplin antrian adalah konsep yang membahas mengenai kebijakan dalam menentukan pelanggan mana yang dipilih dari antrian untuk dilayani, berdasarkan urutan kedatangan pelanggan. Ada empat bentuk disiplin pelayanan yang biasa digunakan dalam praktek yaitu [8]:

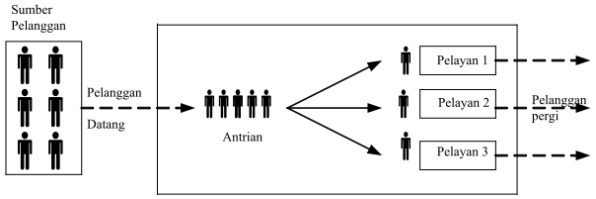
1. FIFO
   First In First Out, siapa yang datang awal dilayani awal, siapa yang datang akhir dilayanai akhir.
2. LIFO
   Last In First Out, siapa yang datang terakhir dilayani lebih dahulu.
3. RS
   Random Service, pelanggan yang dilayani dipilih secara random.
4. *Priority*
   Pelanggan yang dilayani dipilih berdasarkan prioritas.

Dalam sistem antrian, ada dua tipe prioritas, yaitu prioritas *non-preemptive* dan prioritas *preemtive*, berikut adalah penjelasannya [9] :

1. Prioritas *Non-preemptive*
   Proses dengan prioritas rendah yang sedang dilayani tidak dapat diinterupsi oleh proses lain meskipun proses tersebut prioritasnya lebih tinggi. Sehingga setiap proses yang akan dijalankan harus menunggu hingga proses yang sedang dilayani selesai.
2. Prioritas *Preemtive*
   Proses yang sedang dilayani dapat diinterupsi oleh proses yang memiliki prioritas lebih tinggi. Kemudian proses dengan prioritas rendah dapat dilayani kembali setelah proses dengan prioritas tinggi selesai yang dimulai dari titik saat dimana terjadinya interupsi.

2.2 Model Antrian

Berdasarkan sifat pelayanannya, model antrian dapat diklasifikasikan sebagai fasilitas-fasilitas pelayanan dalam susunan saluran dan *phase* yang akan membentuk suatu struktur antrian yang berbeda-beda. Istilah saluran menunjukkan jumlah jalur untuk memasuki sistem pelayanan. Sedangkan istilah *phase* berarti jumlah stasiun-stasiun pelayanan, dimana para langganan harus melaluinya sebelum pelayanan dinyatakan lengkap [8]. Model antrian pada kasus penjadwalan memasak adalah *Multi Chanel Single Phase*. *Multi Chanel Single Phase* terjadi jika dua atau lebih fasilitas pelayanan dialiri oleh suatu antrian tunggal [2]. Contoh lain adalah antrian pada loket di kantor pos dan bank, seperti yang ditunjukkan pada Gambar 1.



Gambar 1. Contoh *Multi chanel single phase* [8].

2.3 Kriteria Penjadwalan

Terdapat beberapa kriteria penjadwalan yang mempengaruhi efisiensi penjadwalan. Kriteria penjadwalan yang menjadi fokus dalam penelitian ini adalah *response time* dan *fairness*. *Response time* adalah jumlah waktu yang dihabiskan mulai dari suatu proses masuk hingga pertama kali proses mendapat respond [3]. *Response time* memberikan pengaruh terhadap lamanya waktu tunggu suatu proses mulai dikerjakan. Jika *response time* dapat diminimalkan maka lamanya waktu tunggu dapat dikurangi sehingga waktu yang diperlukan untuk pengerjaan suatu proses akan menjadi lebih singkat.

Fairness adalah tingkat kesesuaian antara urutan pengerjaan proses yang dihasilkan suatu algoritma dengan urutan pengerjaan proses yang seharusnya (urutan FCFS). Kriteria *fairness* dalam kasus penjadwalan memasak menjadi sangat penting karena berdampak langsung terhadap kepuasan pelanggan. Dalam hal ini akan menjadi tidak adil apabila pelanggan yang datang akhir mendapat layanan lebih awal daripada pelanggan yang telah datang sebelumnya. Dengan memaksimalkan *fairness*, maka kepuasan pelanggan dapat dijaga, sehingga kriteria penjadwalan yang baik dapat tercapai.

2.4 Algoritma Penjadwalan

Algoritma penjadwalan yang baik adalah yang dapat mengoptimalkan performance-nya. Algoritma penjadwalan yang digunakan pada penelitian ini diantaranya adalah :

2.4.1 First Come First Served (FCFS)

FCFS adalah aturan dimana pelaggan yang dilayani terlebih dahulu adalah pelanggan yang datang lebih awal. FCFS tidak cocok bila diterapkan pada kondisi yang real time. Ini dikarenakan suatu proses dapat memonopoli dengan eksekusi yang panjang/lama sehingga menghalangi proses pendek untuk dieksekusi. Selain itu, jika proses yang sedang dikerjakan memakan waktu yang lama maka akan berakibat meningkatnya *response time* dan *waiting time* proses selanjutnya.

2.4.2 Round robin (RR)

Round Robin scheduling adalah versi *preemptive* dari *First Come First Served scheduling*. Pada RR proses dimasukkan pada *first in first out sequence* tetapi setiap proses diijinkan untuk berjalan dalam waktu yang terbatas. Interval waktu ini biasa disebut *time slice* atau *quantum*. Jika suatu proses tidak selesai dieksekusi hingga *time slice* habis, maka proses tersebut *preempted* atau berhenti hingga mendapat giliran dieksekusi kembali. Proses yang *preempted*

ditempatkan pada *ready list* paling belakang dimana proses tersebut harus menunggu proses yang ada pada list tersebut tereksekusi dahulu. Mengalokasikan *time quantum* terlalu kecil dan terlalu besar, dapat mengakibatkan penurunan *performance* yang disebabkan oleh meningkatnya *context switching* yang berlebihan.

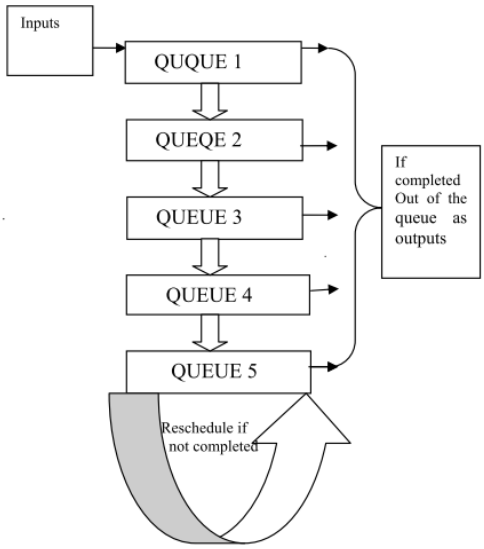
Pada *Round Robin*, jika *time quantum* terlalu besar, mengakibatkan *response time* dari proces terlalu tinggi/lama yang mungkin tidak ditoleransi pada lingkungan yang interaktif. Jika *time quantum* terlalu kecil, maka mengakibatkan *context switch* yang berlebihan sehingga menghasilkan *throughput* yang minimal [10]. Tidak ada cara standar untuk menentukan *quantum*. *Quantum* yang besar akan memaksimalkan *waiting time* dan *quantum* yang kecil dapat meningkatkan *context switching* yang berlebihan antar proses. *Quantum* pada *Round Robin* dapat ditentukan secara statis maupun dinamis. Untuk meningkatkan *performance*, *quantum* dialokasikan secara dinamis pada semua iterasi. *Performance* dari *Round Robin* dengan *Dynamic Quantum* lebih baik dibanding Round Robin dengan *Static Quantum*[7].

2.4.3 Multilevel Feedback Queue (MLFQ)

Multilevel Feedback Queue menjadwalkan proses dengan cara memasukkan proses-proses ke dalam beberapa *queue* berdasarkan besar *time quantum*. Masing-masing *queue* memiliki algoritma sendiri seperti *Round Robin* atau FCFS. Jika masih terdapat proses yang belum selesai dikerjakan maka proses tersebut dapat berpindah dari *queue* satu ke *queue* lain atau berpindah ke *queue* dengan prioritas lebih rendah. Setiap *queue* memiliki *unique time slice* yang berangsur-angsur meningkat dari *queue level* atas ke *queue* yang *levelnya* lebih rendah.

Salah satu penelitian tentang pengembangan algoritma MLFQ yang telah dilakukan adalah penggunaan *Dynamic Time Quantum* untuk meningkatkan *performance* MLFQ. Pada algoritma *Multilevel Feedback Queue* menggunakan *Dynamic Time Quantum, time quantum* dihitung secara dinamis dan berbeda-beda setiap *queue*. Jumlah level dibuat berdasarkan *time quantum* dan *burst time* dari proses [6].

Langkah awal pengerjaan algoritma *Multilevel Feedback Queue* menggunakan *Dynamic Time Quantum* adalah menentukan jumlah *queue*. Pada contoh arsitektur ini dipakai 5 *queues* (Q1 ,Q2 ,Q3 ,Q4 dan Q5) seperti yang ditunjukkan pada Gambar 2. Semua proses yang jadwalkan akan dimasukkan ke Q1. Proses dikerjakan berurutan berdasarkan level prioritas paling tinggi atau Q1. Ketika *time quantum* habis, dan proses belum selesai maka proses tersebut masuk ke level berikutnya. Langkah ini diulangi hingga sampai pada Q5. Ketika proses mencapai *queue* paling rendah, kemudian sisa *burst time* akan dikirim ke *queue* yang sesuai nilainya untuk dijadwalkan ulang sehingga eksekusi proses dapat diselesaikan [6].



Gambar 2. Alur eksekusi pada level *queue* yang berbeda [6].

Untuk mendapatkan nilai *time quantum* yang optimal, langkah awal adalah menghitung *mean* dan *median* dari semua *burst time*. *Mean* dan *median* dihitung menggunakan rumus dibawah ini [6].

$$Mean = \sum_{i=1}^{n} \frac{CPU\ burst\ time\ of\ all\ processes}{no.\ of\ processes}$$

$$Median = \begin{cases} Y_{(n+1)/2}, & \text{if n=odd} \\ 1/2(Y_{n/2} + Y_{(1+n/2)}), & \text{if n=even} \end{cases}$$

Setelah didapat *mean* dan *median*, *time quantum* pertama adalah nilai mutlak dari pengurangan *mean* dan *median,* seperti pada rumus dibawah. Sedangkan nilai *time quantum* selanjutnya adalah dua kali *time quantum* sebelumnya. Penghitungan *time quantum* dilakukan berulang hingga nilai *time quantum* yang dihasilkan adalah lebih besar atau sama dengan *burst time* proses yang terbesar.

$$t_{q1} = |Mean - Median| \qquad t_{q3} = 2 * t_{q2}$$
$$t_{q2} = 2 * t_{q1} \qquad t_{q4} = 2 * t_{q3}$$

3. METODOLOGI

3.1 Studi literatur dan Pemahaman

Studi literatur dan pemahaman ini dilakukan dengan *browsing* internet, pencarian melalui buku, artikel dan jurnal yang relevan dengan objek yang dikaji sehingga diperoleh ketepatan langkah dalam melakukan penelitian. Literasi yang didapat merupakan bahan materi yang berhubungan dengan topik penelitian, diantaranya yaitu mengenai beberapa algoritma penjadwalan seperti *First Come First Served, Round Robin, Multilevel Feedback Queue* dan *Multilevel Feedback Queue dengan Dynamic Time Quantum*.

3.2 Pengumpulan Data

Data yang digunakan dalam penelitian adalah data tentang masakan yang meliputi nama item dan lama waktu pengerjaan. Pengumpulan data dilakukan dengan cara mengamati daftar menu kemudian mencari informasi lama waktu masak. Metode yang digunakan dalam menentukan lama waktu pengerjaan setiap item adalah dengan mengukur waktu secara langsung ketika koki sedang memasak. Waktu pengerjaan diukur dengan satuan detik. Data diambil dari 4 rumah makan yang berbeda. Data waktu yang diperoleh kemudian diambil nilai rata-rata pada setiap item yang sama.

3.3 Penyeleksian Data

Tahap ini dilakukan untuk mendapatkan data makanan dan minuman yang bisa dikerjakan bersamaan apabila terdapat lebih dari satu porsi yang sama dalam sekali pengerjaan. Setiap sekali pengerjaan bersama, jumlah porsi dibatasi maksimal sebanyak 3 porsi. Pembatasan sebanyak 3 porsi dikarenakan berdasarkan keterangan dari koki, jika dalam sekali pengerjaan melebihi 3 porsi akan mengurangi kualitas rasa dari masakan.

3.4 Implementasi

Aplikasi yang dibuat adalah simulasi untuk penjadwalan memasak dengan 3 algoritma pembanding yaitu *First Come First Served, Multilevel Feedback Queue* dan *Multilevel Feedback Queue* menggunakan *Dynamic Time Quantum*.

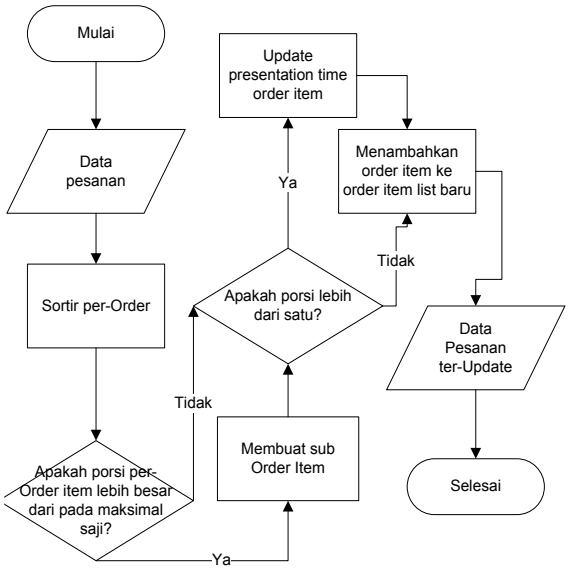
3.4.1 Membuat data pesanan

Pada penelitian ini, data pesanan dibuat secara random. Data pesanan yang dibuat dibagi menjadi dua yaitu makanan dan minuman. Proses pembuatan data pesanan serta penjadwalannya dilakukan secara terpisah antara makanan dan minuman. Parameter dalam membuat data pesanan adalah :

a. Jumlah *order*
   Order adalah satu kelompok pesanan atau kumpulan dari beberapa *order item.*
b. Jumlah *order item* dalam satu *order*
   Order item adalah satu pesanan dengan satu jenis *menu item* yang terdiri dari satu porsi atau lebih. *Menu item* adalah satu jenis makanan atau minuman.
c. Jumlah porsi dalam satu *order item*
d. *Menu id*
   Menu id adalah *id* dari *menu item* yang dipesan.
e. *Time interval*
   Time interval adalah selisih waktu kedatangan antara suatu *order* dengan *order* berikutnya.

3.4.2 Pengerjaan pesanan

Sebelum *order* dijadwalkan, data makanan dan minuman dilakukan proses sortir seperti yang ditunjukkan pada Gambar 3. Proses sortir diterapkan pada setiap *order*. Sortir dilakukan untuk menyatukan *item-item* pesanan yang memiliki kesamaan menu *id* ke dalam satu *order item*. Selanjutnya *order item* diperiksa jumlah porsinya, jika jumlah porsi lebih besar dari pada maksimal saji (tiga porsi) maka *order item* dipecah menjadi beberapa *sub order item* dengan maksimal jumlah porsi adalah tiga porsi.

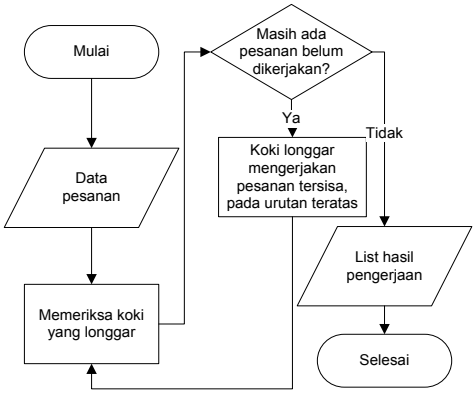


Gambar 3. Proses sortir order

Jika dalam suatu *order item* terdapat lebih dari satu porsi, maka *presentation time*-nya akan dilakukan penambahan sebesar 10% dari *presentation time* untuk setiap pertambahan 1 porsi. Sebagai contoh bila suatu *menu item* dimasak dengan waktu 100 detik dan terdapat 3 porsi pada suatu *order item*, maka *presentation time* yang dihasilkan adalah 100 detik + (20% * 100 detik) yaitu 120 detik. Angka 10% yang digunakan adalah hasil dari rata-rata penambahan total *presentation time* pada jumlah porsi berbeda sesuai data penelitian. Dan berikut adalah alur pengerjaan sesuai algoritma :

3.4.2.1 FCFS

Prinsip First Come First Serve adalah datang awal dikerjakan awal. Pada algoritma ini, koki secara bergantian mengerjakan pesanan sesuai dengan urutan kedatangan. Pesanan yang berada pada urutan berikutnya akan dikerjakan oleh koki yang selesai lebih awal. Alur pengerjaan algoritma First Come First Serve dapat diilustrasikan seperti Gambar 4.



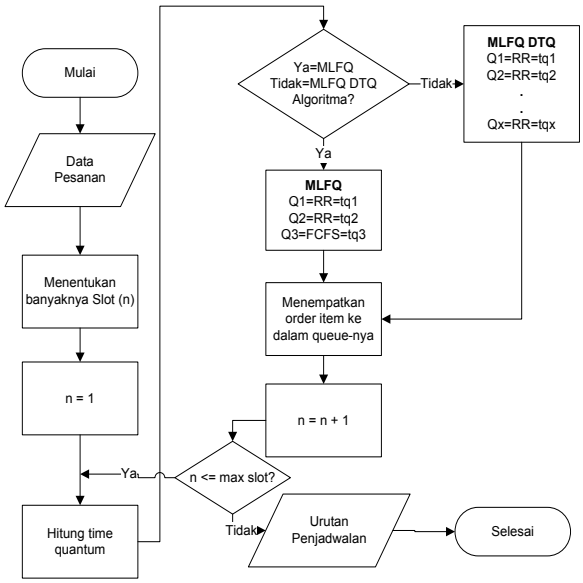
Gambar 4. Alur pengerjaan FCFS

3.4.3.2 MLFQ dan MLFQ DTQ

Pada algoritma ini variabel yang ditentukan diawal adalah jumlah koki dan besar *slot time. Slot time* adalah jarak atau ukuran waktu setiap *slot*. *Slot* digunakan untuk mengelompokkan pesanan berdasarkan waktu kedatangan. Sebagai contoh bila ditentukan *limit time* adalah 60, waktu kedatangan order paling awal 08:29:48, waktu kedatangan order paling akhir 08:34:50, maka pembagian *slot*nya adalah

1. *Slot* 1 → 08:29:48 - 08:30:48
2. *Slot* 2 → 08:30:49 - 08:31:49
3. *Slot* 3 → 08:31:50 - 08:32:50
4. *Slot* 4 → 08:32:51 - 08:33:51
5. *Slot* 5 → 08:33:52 - 08:34:52

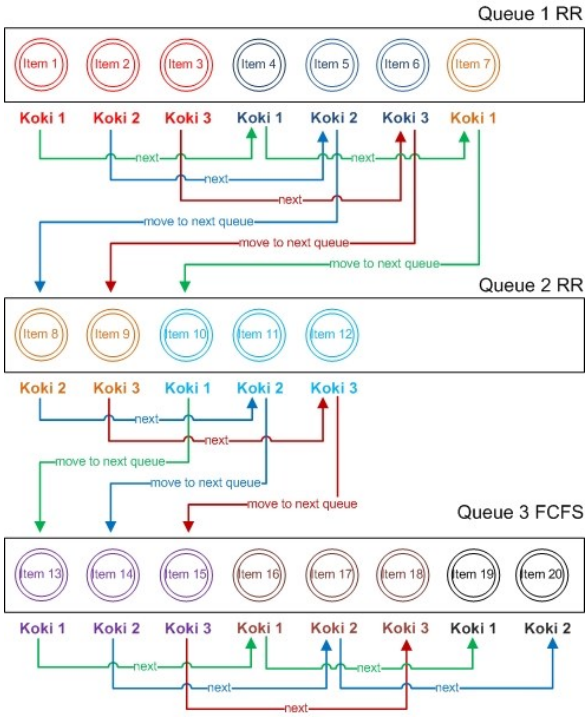
Pengerjaan dimulai dari *slot* yang paling awal. Untuk setiap *slot* langkah awal pengerjaan adalah menghitung *time quantum*(tq). Batas maksimal *time quantum* adalah lebih dari atau sama dengan *presentation time* pesanan terlama dalam *slot* tersebut. Untuk algoritma MLFQ, akan dibuat maksimal 3 *queue*(Q) yaitu Q1 dengan algoritma *Round Robin* dan tq1, Q2 dengan algoritma *Round Robin* tq2 dan Q3 dengan algoritma FCFS. Sedangkan algoritma MLFQ DTQ akan dibuat *queue* sejumlah *time quantum* yang ada. Gambar 5 mengilustrasikan pengerjaan pada MLFQ & MLFQ DTQ.



Gambar 5. Alur pengerjaan MLFQ & MLFQ DTQ

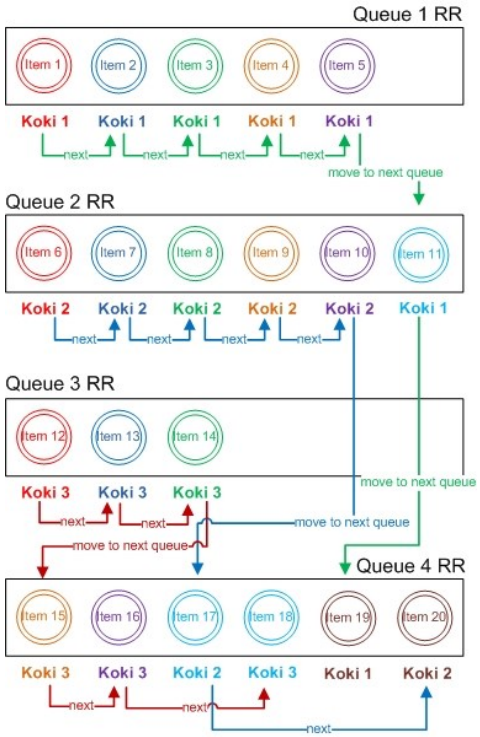
Langkah selanjutnya adalah menempatkan setiap pesanan ke dalam *queue* yang telah ditentukan *time quantum*nya. Setiap pesanan yang akan masuk ke dalam *queue* diurutkan berdasarkan urutan kedatangannya. Setiap pesanan akan masuk dan diperiksa mulai dari *queue* pertama. Apabila *presentation time* pesanan lebih kecil atau sama dengan *time quantum* pada queue pertama maka pesanan ditempatkan pada *queue* pertama. Jika *presentation time* pesanan lebih besar dari pada *time quantum queue* pertama, maka pesanan akan berpindah ke *queue* berikutnya dan diperiksa kembali kesesuaian antara besar *presentation time* pesanan dengan *time quantum queue*. Proses tersebut berulang-ulang hingga semua pesanan menempati *queue*nya.

Pada penelitian ini diusulkan urutan pengerjaan yang berbeda. Cara pengerjaan yang telah digunakan sebelumnya pada MLFQ dapat disebut sebagai urutan secara horizontal. Urutan pengerjaan yang diusulkan pada penelitian ini adalah urutan secara vertikal. Berikut adalah penjelasan cara pengerjaan urutan memasak secara horizontal dan vertikal. Gambar 6 adalah ilustrasi alur pengerjaan secara horizontal. Warna yang sama menandakan pengerjaan *item* dalam waktu hampir bersamaan oleh koki yang beda.



Gambar 6. Contoh pengerjaan MLFQ secara horizontal

Pengerjaan secara horizontal adalah mengerjakan pesanan mulai dari *queue* pertama dan setiap koki mengerjakan sesuai urutan pesanan dalam queue yang telah dijadwalkan. Satu queue dapat dikerjakan oleh lebih dari satu koki. Koki yang selesai paling awal kemudian mengerjakan pesanan yang menempati urutan teratas dalam *queue*. Jika pesanan dalam suatu *queue* telah selesai dikerjakan semua, maka koki bisa pindah ke *queue* selanjutnya yang masih terdapat pesanan yang belum dikerjakan. Urutan yang dikerjakan menurut ilustrasi gambar adalah (K1:I1, K2:I2, K3:I3), (K1:I4, K2:I5, K3:I6), (K1:I7, K2:I8, K3:I9), (K1:I10, K2:I11, K3:I12), (K1:I13, K2:I14, K3:I15), (K1:I16, K2:I17, K3:I18), (K1:I19, K2:I20). *Item* dalam tanda kurung dikerjakan dalam waktu yang bersamaan, K adalah koki dan I adalah *item*.



Gambar 7. Contoh pengerjaan MLFQ secara vertikal

Gambar 7 adalah ilustrasi alur pengerjaan secara vertikal. Pengerjaan secara vertikal adalah pertama-tama setiap koki diinisialisasi mengerjakan pesanan hanya pada suatu *queue*. Jika jumlah *queue* lebih banyak dari pada jumlah koki, maka setiap koki diinisialisasi mengerjakan satu *queue*, kemudian koki yang telah selesai mengerjakan pesanan pada *queue*-nya dapat berpindah ke *queue* baru yang belum mendapat koki. Jika tidak terdapat *queue* sisa, maka koki dapat berpindah ke *queue* berikutnya untuk membantu koki lain menyelesaikan pesanan pada *queue*-nya. Sedangkan jika jumlah *queue* lebih sedikit dari pada jumlah koki, maka setiap *queue* diinisialisasi dikerjakan oleh satu koki, kemudian koki yang belum mendapat jatah *queue* akan ditempatkan pada *queue-queue* tersebut dengan cara menambahkan satu koki pada setiap *queue* dimulai dari *queue* teratas, secara berulang hingga semua koki mendapat jatah *queue*. Urutan yang dikerjakan menurut ilustrasi gambar adalah (K1:I1, K2:I6, K3:I12), (K1:I2, K2:I7, K3:I13), (K1:I3, K2:I8, K3:I14), (K1:I4, K2:I9, K3:I15), (K1:I5, K2:I10, K3:I16), (K1:I11, K2:I17, K3:I18), (K1:I19, K2:I20).

3.5 Analisa Pengujian

Spesifikasi komputer yang digunakan dalam simulasi ini adalah labtop dengan *prosesor core* i5, ram 2 GB, vga Nvidia 512 MB dan hardisk 500 GB. Simulasi dilakukan dengan melakukan penjadwalan terhadap beberapa pesanan menggunakan algoritma yang berbeda yaitu *First Come First Served, Multilevel Feedback Queue* dan *Multilevel Feedback Queue* menggunakan *Dynamic Time Quantum*. Parameter yang digunakan untuk pengujian adalah jumlah koki dan besar *slot time*. Hasil *output* dari penjadwalan akan dianalisa mengenai rata-rata *response time* pengerjaan dan *fairness* (apakah urutan memasak yang dihasilkan sesuai dengan urutan FCFS) dari segi *start time* dan *finish time*.

Algoritma FCFS digunakan sebagai pembanding dalam pengujian *fairness* karena urutan yang dihasilkan baik *start time* dan *finish time* dianggap merepresentasikan *fairness* dalam kasus penjadwalan memasak. Konsep *fairness* yang dimaksud dalam penelitian ini adalah sama dengan konsep algoritma FCFS yaitu yang datang lebih awal akan mendapat prioritas dikerjakan lebih awal dari pada yang datang setelahnya. Selain itu, urutan memasak yang digunakan di restoran pada umumnya adalah menggunakan konsep dari algoritma FCFS.

Response time dihitung dari waktu pesanan mulai dimasak dikurangi waktu kedatangan pesanan. *Response time* dihitung per-*item* yang dikerjakan. Penilaian *response time* pada penelitian ini adalah berdasarkan nilai rata-rata yang dapat dihitung dari total *response time* seluruh item dibagi banyaknya *item*. *Start time* adalah waktu atau kapan suatu pesanan mulai dikerjakan. *Finish time* adalah waktu atau kapan suatu pesanan selesai dikerjakan. Rata-rata *response time* dihitung dengan rumus dibawah ini.

$$\text{Response time} = \text{Start time item} - \text{Arrival time it}$$

$$\text{Rata} - \text{rata response time} = \frac{\text{Total seluruh response time}}{\text{Banyaknya item}}$$

Pengujian *fairness* dilakukan pada algoritma MLFQ dan MLFQ DTQ dengan pembanding adalah FCFS. Pengujian dilakukan dengan parameter "sebelum", "sesudah" dan "*index* sama". Misal terdapat hasil urutan *finish time* pesanan seperti pada Tabel 1.

Tabel 1. Contoh finish time untuk parameter "sebelum"

Algoritma	Urutan Finish Time											
FCFS	2	1	3	4	5	6	7	8	9	10	11	12
MLFQ	2	1	3	4	5	6	8	9	7	10	11	12

Parameter "sebelum" digunakan untuk membandingkan kesesuaian antara MLFQ terhadap FCFS dalam hal suatu order harus selesai dikerjakan sebelum order-order tertentu. Contoh parameter "sebelum" berdasarkan Tabel 1 adalah order 8 pada FCFS selesai dikerjakan sebelum order 9, 10, 11 dan 12 maka order 8 pada MLFQ minimal harus dikerjakan sebelum order 9, 10, 11 dan 12. Karena order 8 pada MLFQ selesai sebelum order 9, 7, 10, 11 dan 12 maka order 8 pada MLFQ sesuai dengan order 8 pada FCFS. Contoh lain, order 7 pada FCFS selesai sebelum order 8, 9, 10, 11 dan 12. Sedangkan order 7 pada MLFQ selesai sebelum order 10, 11 dan 12 maka order 7 pada MLFQ tidak sesuai dengan order 7 pada FCFS.

Tabel 2. Contoh finish time untuk parameter "sesudah"

Algoritma	Urutan Finish Time											
FCFS	2	1	3	4	5	6	7	8	9	10	11	12
MLFQ	2	1	3	4	5	6	8	9	7	10	11	12

Parameter "sesudah" digunakan untuk membandingkan kesesuaian antara MLFQ terhadap FCFS dalam hal suatu order dikerjakan sesudah order-order tertentu. Contoh parameter "sesudah" berdasarkan Tabel 2 adalah order 6 pada FCFS dikerjakan sesudah order 2, 1, 3, 4 dan 5 maka order 6 pada MLFQ minimal harus dikerjakan sesudah order 2, 1, 3, 4 dan 5. Karena order 6 pada MLFQ dikerjakan sesudah order 2, 1, 3, 4 dan 5 maka order 6 pada MLFQ sesuai dengan order 6 pada FCFS. Contoh lain, order 8 pada FCFS dikerjakan sesudah order 2, 1, 3, 4, 5, 6 dan 7. Sedangkan order 8 pada MLFQ dikerjakan sesudah order 2, 1, 3, 4, 5 dan 6 maka order 8 pada MLFQ tidak sesuai dengan order 8 pada FCFS.

Parameter "index sama" digunakan untuk membandingkan kesesuaian posisi atau urutan suatu order pada MLFQ terhadap posisi atau urutan order yang sama pada FCFS. Sebagai contoh, berdasarkan Tabel 3 order 3 pada FCFS berada di urutan ke 3 sedangkan order 3 pada MLFQ berada di urutan ke 3 maka index order 3 pada MLFQ

dan FCFS sesuai. Contoh lain, order 7 pada FCFS berada di urutan ke 7 sedangkan order 7 pada MLFQ berada di urutan ke 9 maka index order 7 pada MLFQ dan FCFS tidak sesuai.

Tabel 3. Contoh urutan finish time parameter "index sama"

Algoritma	Urutan Finish Time											
FCFS	2	1	3	4	5	6	7	8	9	10	11	12
MLFQ	2	1	3	4	5	6	8	9	7	10	11	12

Penilaian *fairness start time* dan *finish time* didapat dari hasil rata-rata skor parameter "sebelum", "sesudah" dan "*index* sama" seperti rumus di bawah ini.

$$\text{Rata} - \text{rata fairness} = \frac{\text{skor "sebelum"} + \text{skor "sesudah"} + \text{skor "index sama"}}{3 * \text{total order}} * 100\%$$

Hasil perhitungan *fairness finish time* dari Tabel 1 adalah parameter "sebelum" terdapat 11 *order* yang sesuai, parameter "sesudah" terdapat 10 *order* yang sesuai dan parameter "*index* sama" terdapat 9 *order* yang sesuai. Sehingga rata-rata *fairness finish time* yang didapat adalah ((11+10+9)/(3*12)) x 100% = 83%.

4. HASIL DAN PEMBAHASAN

Data pesanan dibuat secara random. Tabel 4 berisi *renge* minimal dan maksimal dari parameter yang digunakan dalam simulasi.

Tabel 4. Parameter Create Order

Parameter	Min	Max
Menu id	1	30
Jumlah *order* item dalam 1 *order*	1	5
Jumlah porsi dalam 1 *order item*	1	5
Jumlah *order*	10	20
Time Interval	10 detik	20 detik

Pada penelitian ini sampel data pesanan yang digunakan adalah sebanyak 15 *order parent*. Pada proses penjadwalan, parameter yang digunakan seperti pada Tabel 5.

Tabel 5. Parameter Penjadwalan

Parameter	Value
Total Koki	3, 4, 5
First Slot Time	10 detik
Increment Slot Time	5 detik
Max Slot Time	100 detik

Max Slot Time adalah setengah dari selisih kedatangan orderParent pertama dengan orderParent terakhir dalam satu kasus, nilai tersebut dipilih agar dalam satu kasus minimal terdapat 2 slot penjadwalan.

4.1 Perbandingan Rata-rata Response time

a. Perbandingan percobaan FCFS dan MLFQ

Tabel 4. Perbandingan FCFS dan MLFQ

Koki	FCFS	Equal	MLFQ
3	0	0	19
4	0	0	19
5	0	0	19

b. Perbandingan percobaan MLFQ dan MLFQ DTQ

Tabel 5. Perbandingan MLFQ dan MLFQ DTQ

Koki	MLFQ	Equal	MLFQ DTQ
3	1	3	15
4	2	3	14
5	0	2	17

c. Perbandingan percobaan FCFS dan MLFQ V

Tabel 6. Perbandingan FCFS dan MLFQ V

Koki	FCFS	Equal	MLFQ V
3	0	0	19
4	0	0	19
5	0	0	19

d. Perbandingan percobaan MLFQ V dan MLFQ DTQ V

Tabel 7. Perbandingan MLFQ V dan MLFQ DTQ V

Koki	MLFQ	Equal	MLFQ DTQ
3	7	1	11
4	9	3	7
5	7	0	12

e. Perbandingan percobaan MLFQ dengan MLFQ V

Tabel 8. Perbandingan MLFQ dan MLFQ V

Koki	MLFQ	Equal	MLFQ V
3	19	0	0
4	19	0	0
5	19	0	0

f. Perbandingan percobaan MLFQ DTQ dan MLFQ DTQ V

Tabel 9. Perbandingan MLFQ DTQ dan MLFQ DTQ V

Koki	MLFQ DTQ	Equal	MLFQ DTQ V
3	19	0	0
4	19	0	0
5	19	0	0

Hasil dari 19 percobaan, pada Tabel 4 menunjukkan rata-rata *response time* MLFQ lebih cepat dibanding FCFS, baik pada parameter koki berjumlah 3, 4, maupun 5. Tabel 5 rata-rata *response time* MLFQ DTQ dominan lebih cepat dibandingkan MLFQ, dengan perbandingan 15 percobaan pada koki berjumlah 3 orang, 14 percobaan pada koki berjumlah 4 orang, dan 17 percobaan pada koki berjumlah 5 orang. Tabel 6 rata-rata *response time* MLFQ vertikal lebih cepat dibandingkan dengan FCFS, baik pada parameter koki berjumlah 3, 4, maupun 5.

Hasil perbandingan pada Tabel 7 didapat rata-rata *response time* MLFQ vertikal lebih cepat dibandingkan dengan MLFQ DTQ vertikal, baik pada parameter koki berjumlah 3, 4 maupun 5. Tabel 8 rata-rata *response time* MLFQ lebih cepat dibandingkan dengan MLFQ vertikal, baik pada parameter koki berjumlah 3, 4, maupun 5. Tabel 9 rata-rata *response time* MLFQ DTQ lebih cepat dibandingkan dengan MLFQ DTQ vertikal, baik pada parameter koki berjumlah 3, 4, maupun 5.

4.2 Perbandingan Rata-rata Response Slot 30, 45 dan 60

Tabel 10. Perbandingan rata-rata Response Time per-Slot

Slot Time	Koki	Rata-rata Response				
		FCFS	MLFQ	MLFQ DTQ	MLFQ V	MLFQ DTQ V
30	3	2856.64	2690.06	2645.76	2728.02	2711.74
30	4	2073.85	1960.39	1942.16	1980.20	1991.5
30	5	1609.48	1499.65	1487.92	1540.35	1551.30
45	3	2856.64	2602.28	2596.55	2668.43	2664.45
45	4	2073.84	1904.93	1902.10	1959.70	1949.08
45	5	1609.48	1504.33	1493.94	1520.51	1516.93
60	3	2856.64	2520.59	2520.59	2611.12	2611.12
60	4	2073.85	1856.95	1856.95	1900.80	1990.80
60	5	1609.48	1436.47	1422.23	1483.91	1479.14

Tabel 10 adalah hasil dari perbandingan rata-rata response time algoritma FCFS, MLFQ, MLFQ DTQ, MLFQ V dan MLFQ DTQ V dengan slot time 30, 45 dan 60 detik pada uji coba menggunakan koki berjumlah 3, 4, dan 5. Berdasarkan Tabel 12, jika diamati rata-rata response time algoritma MLFQ DTQ adalah yang paling cepat dibandingkan dengan FCFS, MLFQ, MLFQ vertikal dan MLFQ DTQ vertikal.

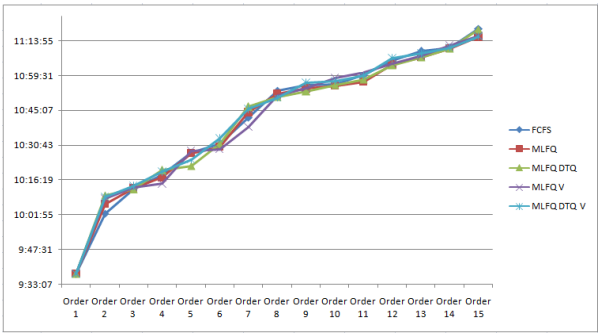
4.3 Perbandingan Start Time dan Finish Time

4.4.1 Start Time

Gambar 8. Grafik *start time* pada *slot time* 30 detik dan 3 koki

Grafik pada Gambar 8 menggambarkan perbandingan antara algoritma FCFS, MLFQ, MLFQ DTQ, MLFQ V dan MLFQ DTQ V dalam hal *start time order* dengan parameter *slot time* 30 detik dan koki 3 orang. Dapat dilihat pada *order* 3, 4, 6, 8, 13 dan 14, FCFS memiliki *start time* yang lebih lama dibandingkan dengan algoritma lain. Pada algoritma MLFQ dan MLFQ DTQ, semua *order* dikerjakan lebih awal jika dibandingkan dengan *order* pada FCFS. Sedangkan pada algoritma MLFQ V dan MLFQ DTQ V, semua *order* kecuali *order* 11 dikerjakan lebih awal jika dibandingkan dengan *order* pada FCFS.

4.4.2 Finish Time



Gambar 9. Grafik *finish time* pada *slot time* 30 detik dan 3 koki

Grafik pada Gambar 8 menggambarkan perbandingan antara algoritma FCFS, MLFQ, MLFQ DTQ, MLFQ V dan MLFQ DTQ V dalam hal *finish time order* dengan parameter *slot time* 30 detik dan koki 3 orang. Berdasarkan grafik dalam Gambar 9, algoritma MLFQ memiliki *finish time* lebih cepat jika dibandingkan algoritma lain pada *order* 6, 10, 11, 14 dan 15. Sedangkan algoritma MLFQ DTQ memiliki *finish time* lebih cepat jika dibandingkan algoritma lain pada *order* 3, 5, 9, 12 dan 13.

4.4 Perbandingan Fairness

Tabel 11. Perbandingan Fairness Start Time

Slot Time	Koki	Fairness Start Time			
		MLFQ	MLFQ DTQ	MLFQ V	MLFQ DTQ V
30	3	64 %	64 %	82 %	82 %
30	4	64 %	64 %	73 %	73 %
30	5	73 %	73 %	82 %	73 %
45	3	71 %	71 %	82 %	73 %
45	4	71 %	73 %	78 %	62 %
45	5	58 %	58 %	73 %	64 %
60	3	64 %	64 %	78 %	78 %
60	4	64 %	64 %	73 %	73 %
60	5	64 %	64 %	69 %	69 %

Tabel 11 adalah hasil dari perbandingan *fairness start time* algoritma MLFQ, MLFQ DTQ, MLFQ V dan MLFQ DTQ V terhadap FCFS dengan *slot time* 30, 45 dan 60 detik pada uji coba menggunakan koki berjumlah 3, 4, dan 5. Tabel 12 adalah hasil dari perbandingan *fairness finish time* algoritma MLFQ, MLFQ DTQ, MLFQ V dan MLFQ DTQ V terhadap FCFS dengan *slot time* 30, 45 dan 60 detik pada uji coba menggunakan koki berjumlah 3, 4, dan 5.

Tabel 12. Perbandingan Fairness Finish Time

Slot Time	Koki	Fairness Finish Time			
		MLFQ	MLFQ DTQ	MLFQ V	MLFQ DTQ V
30	3	91 %	82 %	87 %	78 %
30	4	82 %	73 %	69 %	78 %
30	5	69 %	69 %	73 %	69 %
45	3	69 %	60 %	78 %	78 %
45	4	73 %	69 %	91 %	82 %
45	5	64 %	64 %	78 %	73 %
60	3	51 %	51 %	51 %	51 %
60	4	51 %	51 %	47 %	47 %
60	5	42 %	42 %	47 %	47 %

Pada kasus penjadwalan pemesanan makanan di restoran, faktor *fairness* sangat penting karena berkaitan dengan keadilan pelayanan antara pengunjung. Sehingga fairness pada penelitian ini dianggap lebih penting daripada keunggulan *response time*. Dari data Tabel 11 dapat disimpulkan pengerjaan secara vertikal memiliki *fairness start time* yang lebih baik dari pada pengerjaan secara biasa/horizontal. Sedangkan pada Tabel 14 perbandingan *fairness finish time* pengerjaan vertikal sedikit lebih unggul daripada pengerjaan secara horizontal. Dan pada pengerjaan vertikal terlihat MLFQ Vertikal lebih unggul daripada MLFQ DTQ Vertikal baik dari *fairness start time* maupun *finish time*.

Dari semua perbandingan yang telah dilakukan, algoritma MLFQ Vertikal dianggap paling unggul dari algoritma pembanding lain. Karena pada MLFQ Vertikal memiliki *fairness* paling unggul meskipun *response time*-nya tidak lebih bagus daripada MLFQ DTQ, namun *response time* MLFQ Vertikal masih lebih baik jika dibandingkan FCFS.

5. KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan pada penelitian, hasil dan pembahasan yang telah dilakukan dan dipaparkan di atas, maka dapat ditarik kesimpulan :

- Hasil dari 19 percobaan dapat diperoleh, rata-rata *response time* MLFQ lebih cepat dibanding FCFS, baik pada parameter koki berjumlah 3, 4, maupun 5. Pada percobaan yang sama disimpulkan bahwa rata-rata *response time* MLFQ DTQ dominan lebih cepat dibandingkan MLFQ, dengan perbandingan 15 percobaan pada koki berjumlah 3 orang, 14 percobaan pada koki berjumlah 4 orang, dan 17 percobaan pada koki berjumlah 5 orang.
- Hasil dari 19 percobaan dapat diperoleh, rata-rata *response time* MLFQ vertikal lebih cepat dibandingkan dengan FCFS. Pada percobaan yang sama disimpulkan bahwa rata-rata *response time* MLFQ lebih cepat dibandingkan dengan MLFQ vertikal dan disimpulkan pula rata-rata *response time* MLFQ DTQ lebih cepat dibandingkan dengan MLFQ DTQ vertikal.
- Pengerjaan secara horizontal atau biasa memiliki keunggulan pada segi *response time*, sedangkan pengerjaan secara vertikal memiliki keunggulan pada segi *fairness*.

- MLFQ vertikal dianggap paling bagus dari pada algoritma pembanding yang lain karena paling unggul dari segi *fairness* dan memiliki rata-rata *response time* yang lebih baik jika dibandingkan dengan FCFS.

5.2 Saran

Untuk pengembangan penelitian ini selanjutnya dapat dilakukan pada cara penentuan besar *slot time* yang menghasilkan rata-rata *response time, fairness, start time* dan *finish time* paling optimal.

6. DAFTAR PUSTAKA

[1] B. H. Salim, "Studi Deskriptif Custumer Complaint di Restoran Hotel Meritus Surabaya," *Calyptra : Jurnal Ilmiah Mahasiswa Universitas Surabaya*, vol. 2, no. 2, 2013.

[2] D. A. Ginting, S. S. Aristoteles, and O. D. Endah, "Implementation of Multilevel Feedback Queue Algorithm in Restaurant Order Food Application Development for Android and iOS Platforms," *International Journal of Computer Applications*, vol. 80, no. 13, pp. 24–30, 2013.

[3] K. K. Bharathi, "Studying Main Differences between Multilevel Queue (MLQ) and Multilevel Feedback Queue (MLFQ),"*International Journal of Computer Applications in Engineering Sciences*, vol. 3, no. 1, p. 29, 2013.

[4] R. K. Yadav and A. Upadhyay, "A Fresh Loom for Multilevel Feedback Queue Scheduling Algorithm,"*International Journal of Advances in Engineering Sciences*, vol. 2, no. 3, pp. 21–23, 2012.

[5] D. Chouhan, S. D. Kumar, and J. A. Ajay, "A MLFQ Scheduling Technique Using M/M/c Queues for Grid Computing,"*International Journal of Computer Science Issues*, vol. 10, no. 2, 2013.

[6] H. S. Behera, R. K. Naik, and S. Parida, "Improved Multilevel Feedback Queue Scheduling Using Dynamic Time Quantum and Its Performance Analysis,"*International Journal of Computer Science and Information Technology*, vol. 3, pp. 3801–3807, 2012.

[7] M. R. A. Dhumal, M. T. A. Maktum, and M. L. Ragha, "Dynamic Quantum based Genetic Round Robin Algorithm,"*International Journal of Advanced Research in Computer and Communication Engineering*, vol. 3, no. 03, 2014.

[8] L. F. Napitu, "Analisis Antrian Pada Pt. Bank Rakyat Indonesia Cabang Pematang Siantar Unit Pasar Horas," *Universitas Sumatera Utara*, 2008.

[9] H. Ilyas, *Rekayasa Trafik Telekomunikasi*. UNJANI, 2013.

[10] S. M. Mostafa, S. Z. Rida, and S. H. Hamad, "Finding time quantum of round robin CPU scheduling algorithm in general computing systems using integer programming,"*International Journal of Research and Reviews in Applied Sciences*, vol. 5, no. 1, pp. 64–71, 2010.

Volume 4. Nomor 2. Desember 2015

Pembangunan Sistem Informasi Presensi Perkuliahan Menggunakan Basis Data Terdistribusi Dengan Metode Replikasi-Asynchronous — 48 – 56

Arifin Satria Ajinusa, Antonius Bima Murti W, Abdul Aziz

Mobile Positioning Menggunakan Network Identity Sebagai Sistem Absensi Berbasis Mobile Studi Kasus: Universitas Sebelas Maret — 57 – 62

Endar Suprih Wihidayat, Azis Rahmanto, Afrizal Doewes

Optimasi Produksi Menggunakan Algoritma Fuzzy Linear Programming (Studi Kasus : Produksi Tas UKM Cantik Souvenir) — 63 – 72

Lia Primadani, Sarngadi Palgunadi, Bambang Harjito

Analisis Perbandingan Metode Vector Space Model dan Weighted Tree Similarity dengan Cosine Similarity pada kasus Pencarian Informasi *Pedoman Pengobatan Dasar di Puskesmas* — 73 – 83

Viko Basmalah Wicaksono, Ristu Saptono, Sari Widya Sihwi

Implementasi Iterative Dichotomiser 3 Pada Data Kelulusan Mahasiswa S1 di Universitas Sebelas Maret — 84 – 91

Thanh Thi Bi Dan, Sari Widya Sihwi, Rini Anggrainingsih

Pemodelan Penjadwalan Multilevel Feedback Queue Menggunakan Dynamic Time Quantum Pada Kasus Pemesanan Makanan di Restoran — 92 – 100

Tri Wahyu Prasetyo, Wiharto, Afrizal Doewes

Diterbitkan oleh :
Program Studi Informatika
Fakultas Matematika dan Ilmu Pengetahuan Alam
Universitas Sebelas Maret
Jl. Ir. Sutami 36A Kentingan Surakarta 57126 Indonesia
Telp. 62-271-663451, Fax. 62-271-663451
email : itsmart_uns@yahoo.co.id

ISSN : 2301-7201