Optimasi Penjadwalan Ujian di Perguruan Tinggi Berbasis Operasi Teori Himpunan

Jovandra Otniel P. S. - 13523141[1,2]
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]*jovandra.siregar@gmail.com*, [2]*13523141@std.stei.itb.ac.id*

***Abstract*—Efficient exam scheduling is a critical aspect of academic management, requiring careful consideration of constraints such as departmental requirements, time slots, locations, and available dates. This work introduces an optimization framework for university exam scheduling, leveraging set theory operations to model and resolve scheduling conflicts systematically. The system incorporates a dynamic data management interface, enabling users to seamlessly add, edit, and delete scheduling parameters, while maintaining an up-to-date database. Conflict detection mechanisms ensure compliance with predefined rules, such as avoiding overlapping schedules for the same or different departments under specific constraints. Additionally, the system supports persistent storage through a JSON-based mechanism, allowing data continuity across sessions. This solution minimizes manual intervention, reduces errors, and enhances resource allocation efficiency. By addressing the complexities of multi-constraint scheduling, the proposed framework offers significant benefits for academic institutions seeking to streamline their scheduling processes and optimize their resource usage.**

***Keywords*—Academic management, schedule conflict detection, exam scheduling, optimization framework, set theory.**

I. PENDAHULUAN

Penjadwalan ujian di perguruan tinggi merupakan salah satu tugas yang sangat penting namun menantang bagi institusi akademik. Proses ini memerlukan penyeimbangan berbagai kendala, seperti memastikan ketersediaan mahasiswa, dosen, dan ruang ujian, sekaligus mematuhi kebijakan institusi serta prinsip keadilan akademik. Kompleksitas ini semakin bertambah di universitas besar, di mana koordinasi antara berbagai departemen, mata kuliah, dan kelompok mahasiswa harus dilakukan dalam kerangka waktu yang terbatas.

Day Classes:

Regular Class Meeting Time	Date of Exam	Time of Exam
08:00 MW, WF, MWF	Monday, 12 May	07:45 – 09:45 AM
08:30 MW, WF, MWF	Monday, 12 May	07:45 – 09:45 AM
11:00 MW, WF, MWF	Monday, 12 May	10:00 – 12 Noon
02:00 MW, WF, MWF	Monday, 12 May	12:30 – 02:30 PM
03:30 MW, WF, MWF	Monday, 12 May	02:45 – 04:45 PM
08:00 TR	Tuesday, 13 May	07:45 – 09:45 AM
11:00 TR	Tuesday, 13 May	10:00 – 12 Noon
02:00 TR	Tuesday, 13 May	12:30 – 02:30 PM
03:30 TR	Tuesday, 13 May	02:45 – 04:45 PM
09:30 MW, WF, MWF	Wednesday, 14 May	07:45 – 09:45 AM
12:30 MW, WF, MWF	Wednesday, 14 May	10:00 – 12 Noon
01:00 MW, WF, MWF	Wednesday, 14 May	10:00 – 12 Noon
09:30 TR	Thursday, 15 May	07:45 – 09:45 AM
12:30 TR	Thursday, 15 May	10:00 – 12 Noon
01:00 TR	Thursday, 15 May	10:00 – 12 Noon
10:00 MW, WF, MWF	Friday, 16 May	12:30 – 02:30 PM
Friday Only Classes	Friday, 16 May	10:15 – 12:15 PM
Make up exam times for those with conflicts	Wednesday, 14 May	03:00 – 05:00 PM
	Thursday, 15 May	02:45 – 04:45 PM

Gambar 1.1 Ilustrasi Jadwal Ujian
(Sumber: umsl.edu)

Pendekatan manual dalam penjadwalan ujian sering kali menimbulkan konflik, ketidakefisienan, dan kesalahan, terutama ketika menghadapi waktu yang tumpang tindih, penggunaan ruang bersama, atau kebutuhan departemen yang beragam. Masalah-masalah ini dapat menyebabkan ketidakpuasan mahasiswa, peningkatan beban kerja administratif, serta pemanfaatan sumber daya yang kurang optimal. Oleh karena itu, diperlukan sistem otomatis yang mampu mengelola kompleksitas tersebut dan menghasilkan jadwal yang teroptimasi.

Dalam penelitian ini, diusulkan sebuah kerangka kerja otomatis untuk penjadwalan ujian berbasis operasi teori himpunan. Teori himpunan menyediakan landasan matematis yang kuat untuk menangani kendala yang saling tumpang tindih dan mendeteksi konflik. Dengan memodelkan parameter-parameter penjadwalan, seperti slot waktu, ruang, dan alokasi mata kuliah sebagai himpunan matematis, sistem ini dapat secara efisien mendeteksi dan menyelesaikan konflik penjadwalan melalui operasi seperti irisan dan gabungan. Pendekatan ini memastikan bahwa tidak ada dua ujian yang berlangsung bersamaan dalam waktu dan tempat yang sama, kecuali jika diizinkan oleh kebijakan institusi.

Sistem yang diusulkan dilengkapi dengan antarmuka yang ramah pengguna untuk mengelola data penjadwalan. Pengguna dapat secara dinamis menambahkan, mengedit, dan menghapus data mata kuliah, slot waktu, ruang, dan tanggal, dengan sistem yang secara otomatis memperbarui basis data untuk mencerminkan perubahan tersebut. Algoritma deteksi konflik diintegrasikan untuk memvalidasi setiap modifikasi, memastikan kepatuhan terhadap aturan penjadwalan. Selain itu, mekanisme penyimpanan berbasis file JSON diterapkan untuk menjaga keberlanjutan data, memungkinkan pengguna untuk menyimpan dan memuat data penjadwalan secara mulus antar sesi.

Salah satu fitur utama sistem ini adalah kemampuannya untuk beradaptasi dengan berbagai kebutuhan institusi. Misalnya, sistem dapat menerapkan aturan khusus untuk ujian dalam departemen yang sama, seperti mencegah jadwal yang bertumpang tindih dalam satu ruang, sekaligus memberikan fleksibilitas untuk ujian antar departemen. Aturan-aturan ini dapat disesuaikan, sehingga sistem ini dapat diterapkan di berbagai lingkungan akademik dengan mudah.

Dengan mengotomatiskan proses penjadwalan, kerangka kerja yang diusulkan ini mengurangi beban administratif dan meminimalkan potensi kesalahan. Sistem ini meningkatkan

efisiensi alokasi sumber daya, memastikan bahwa ruang, slot waktu, dan sumber daya manusia dimanfaatkan secara optimal. Selain itu, sistem ini meningkatkan pengalaman keseluruhan bagi mahasiswa dan dosen dengan menyediakan jadwal yang adil, bebas konflik, dan sesuai dengan tujuan institusi.

Makalah ini disusun sebagai berikut. Bagian II membahas penelitian terkait dalam bidang penjadwalan ujian otomatis dan menyoroti keterbatasan sistem yang ada. Bagian III menyajikan rincian implementasi dan arsitektur sistem, diikuti oleh Bagian IV yang mengevaluasi kinerja sistem menggunakan data penjadwalan dunia nyata. Akhirnya, Bagian V menyimpulkan makalah ini dan menguraikan arah penelitian di masa depan.

II. Landasan Teori

A. Teori dan Operasi Himpunan

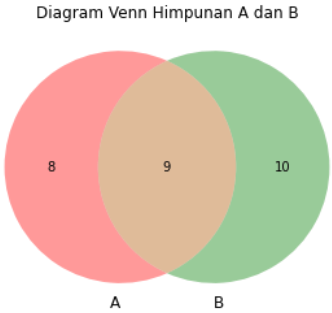
Teori himpunan (set theory) merupakan cabang matematika yang fundamental dalam berbagai disiplin ilmu, termasuk dalam pengembangan algoritma penjadwalan. Dalam sistem penjadwalan ujian ini, teori himpunan digunakan untuk memodelkan berbagai entitas seperti waktu, ruangan, mata kuliah, dan departemen sebagai himpunan yang saling berinteraksi. Operasi-operasi dasar seperti irisan, gabungan, keanggotaan, dan himpunan bagian memainkan peran krusial dalam mendeteksi konflik dan memastikan alokasi sumber daya yang efisien.

Secara spesifik, program ini menggunakan beberapa operasi himpunan berikut:

1. Irisan (Intersection): Untuk mendeteksi tumpang tindih waktu antara dua jadwal ujian.
2. **Gabungan (Union)**: Untuk menggabungkan slot waktu yang tersedia atau mengelola daftar entitas yang terlibat.
3. Keanggotaan (Membership): Untuk memeriksa apakah suatu entitas (seperti ruangan atau tanggal) sudah ada dalam himpunan tertentu.
4. Himpunan Bagian (Subset): Untuk menentukan apakah suatu himpunan merupakan bagian dari himpunan lain, misalnya dalam konteks pemilihan slot waktu.

Penggunaan operasi-operasi ini memungkinkan sistem untuk melakukan analisis yang efisien terhadap data penjadwalan, mengidentifikasi potensi konflik, dan mengoptimalkan alokasi sumber daya secara otomatis.

B. Irisan



Gambar 2.1 Diagram Venn irisan A dan B
(Sumber: Dokumentasi Pribadi)

Irisan atau *intersection* dalam teori himpunan adalah operasi yang menghasilkan himpunan baru yang berisi elemen-elemen yang terdapat di kedua himpunan yang dibandingkan. Secara matematis, irisannya dapat dituliskan sebagai:

$$A \cap B = \{x \mid x \in A \text{ dan } x \in B\}$$

Dalam konteks penjadwalan ujian, irisan digunakan untuk mendeteksi apakah dua jadwal ujian memiliki tumpang tindih waktu. Pada makalah ini, digunakan rentang jam $[a, b)$ pada penjadwalan ujian, di mana a merupakan jam mulai dan b jam selesai. Misalnya, jika jadwal Ujian A memiliki interval waktu dari pukul 08:00 hingga 10:00 dan Ujian B dari pukul 09:00 hingga 11:00, maka:

$$A = \{8,9\}; B = \{9,10\}$$
$$A \cap B = \{9\}$$

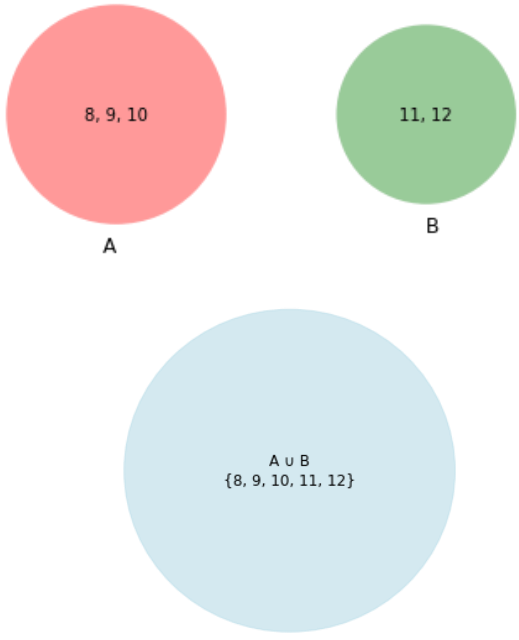
Irisan $\{9\}$ menandakan adanya tumpang-tindih (*overlap*) pada pukul 09:00, yang berarti kedua ujian tidak dapat dijadwalkan pada waktu yang bersamaan jika menggunakan ruangan atau departemen (jurusan) yang sama.

C. Gabungan

Gabungan atau *union* dalam teori himpunan adalah operasi yang menghasilkan himpunan baru yang berisi semua elemen dari kedua himpunan yang dibandingkan, tanpa duplikasi. Secara matematis, gabungannya dapat dituliskan sebagai:

$$A \cup B = \{x \mid x \in A \text{ atau } x \in B\}$$

Dalam sistem penjadwalan ujian, operasi gabungan digunakan untuk mengelola daftar waktu atau ruangan yang tersedia. Misalnya, jika dua mata kuliah memiliki slot waktu yang berbeda, gabungan kedua himpunan waktu tersebut akan mencakup semua slot yang tersedia untuk kedua mata kuliah tersebut.



Gambar 2.2 Diagram Venn Gabungan A dan B
(Sumber: Dokumentasi Pribadi)

Misalkan, himpunan slot waktu yang tersedia untuk Mata Kuliah A adalah $\{8,9,10\}$ dan untuk Mata Kuliah B

adalah {11,12}, Maka gabungan dari kedua himpunan tersebut adalah:

$$A \cup B = \{8,9,10,11,12\}$$

Hal ini memungkinkan program untuk mengetahui semua slot waktu yang tersedia setelah penjadwalan kedua mata kuliah tersebut.

D. Keanggotaan

Keanggotaan atau membership dalam teori himpunan adalah konsep yang menentukan apakah suatu elemen termasuk dalam himpunan tertentu. Secara matematis, hal ini dituliskan sebagai:

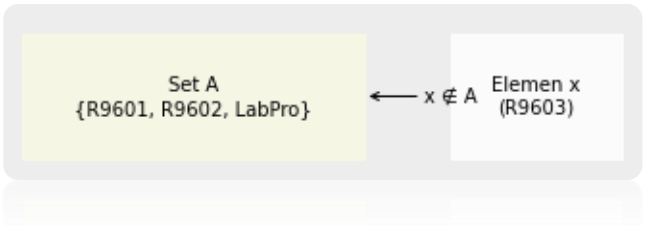
$$x \in A \text{ atau } x \notin A$$

Contohnya, dalam sistem penjadwalan ujian, pemeriksaan keanggotaan digunakan untuk memastikan bahwa entitas seperti ruangan, tanggal, atau slot waktu sudah terdaftar sebelum melakukan operasi penjadwalan. Misalnya, sebelum menambahkan jadwal ujian, sistem akan memeriksa apakah ruangan yang dipilih sudah ada dalam daftar ruangan yang tersedia.

Implementasi keanggotaan dalam sistem penjadwalan ujian dilaksanakan seperti berikut: Jika daftar ruangan yang tersedia adalah $\{R9601, R9602, LabPro\}$ dan pengguna ingin menambahkan ruangan $R9603$, sistem melakukan pengecekan terlebih dahulu menggunakan teori keanggotaan himpunan.

$$R9603 \notin \{R9601, R9602, LabPro\}$$

Karena $R9603$ tidak termasuk dalam himpunan, program menambahkan anggota tersebut ke himpunan. Hasilnya, didapat $\{R9601, R9602, LabPro, R9603\}$. Hal ini bertujuan supaya tidak ada data duplikat dalam himpunan.



Gambar 2.3 Representasi Himpunan A dan Elemen x
(Sumber: Dokumentasi Pribadi)

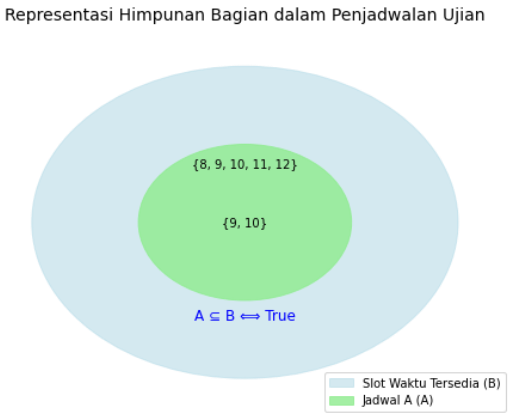
E. Himpunan Bagian

Himpunan bagian atau subset dalam teori himpunan adalah konsep yang menyatakan bahwa semua elemen dalam suatu himpunan A juga terdapat dalam himpunan B. Secara matematis, hal ini dituliskan sebagai:

$$A \subseteq B \iff \forall x(x \in A \Rightarrow x \in B)$$

Dalam konteks penjadwalan ujian, konsep himpunan bagian digunakan untuk memastikan bahwa jadwal yang diusulkan memenuhi semua kriteria yang telah ditetapkan. Misalkan, himpunan slot waktu yang tersedia adalah $\{8,9,10,11,12\}$ dan jadwal mata kuliah A adalah {9,10}, maka:

$$\{9,10\} \subseteq \{8,9,10,11,12\}$$



Gambar 2.4 Diagram Venn Himpunan Bagian A dari B
(Sumber: Dokumentasi Pribadi)

Hal ini menunjukkan bahwa jadwal ujian Mata Kuliah A valid karena seluruh slot waktunya berada dalam himpunan slot waktu yang diizinkan.

F. Beda Setangkup (Symmetric Difference)

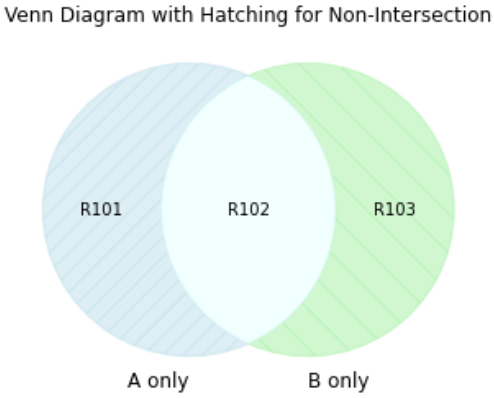
Beda setangkup antara himpunan A dan B adalah himpunan yang berisi anggota (elemen) yang terdapat di A atau B tetapi tidak keduanya. Secara matematis, beda setangkup dapat ditulis sebagai

$$A \Delta B = (A - B) \cup (B - A)$$

Dalam sistem penjadwalan, symmetric difference dapat digunakan untuk mengidentifikasi slot waktu atau ruangan yang unik bagi masing-masing mata kuliah atau departemen, tanpa adanya tumpang tindih. Ini berguna untuk memahami perubahan atau perbedaan antara dua set jadwal atau sumber daya.

Sebagai contoh, himpunan ruangan yang dipakai mata kuliah A adalah $\{R9601, R9602\}$, dan himpunan ruangan untuk mata kuliah B adalah $\{R9602, R9603\}$. Untuk memastikan bahwa tidak ada ruangan yang dipakai secara bersamaan, digunakan operasi beda setangkup:

$$A \Delta B = \{R9601, R9603\}$$



Gambar 2.5 Arsiran Beda Setangkup
(Sumber: Dokumentasi Pribadi)

Artinya ruangan 9601 dan 9603 dapat dipakai oleh masing-masing mata kuliah tanpa tumpang-tindih.

G. Saling Lepas (Disjoint Sets)

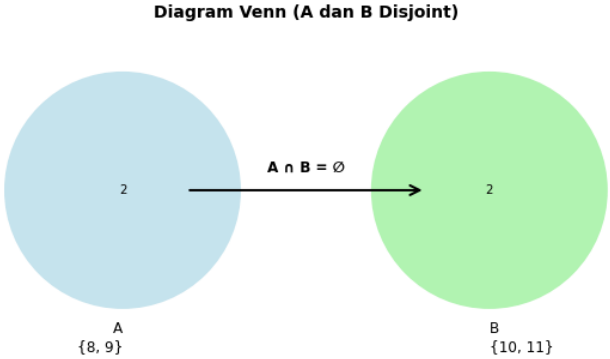
Disjoint Sets atau *himpunan yang saling lepas* adalah dua himpunan yang tidak memiliki elemen yang sama. Secara matematis, dua himpunan A dan B dikatakan disjoint jika:

$$A \cap B = \emptyset$$

Konsep ini merupakan penerapan dari operasi irisan himpunan dengan ketentuan yang lebih spesifik, di mana setiap elemen merupakan anggota dari satu himpunan atau himpunan lain, namun tidak keduanya. Dalam konteks penjadwalan ujian, konsep disjoint sets digunakan untuk memastikan bahwa dua jadwal ujian tidak memiliki tumpang tindih waktu atau tidak menggunakan ruangan yang sama. Ini sangat penting untuk menjaga kelancaran pelaksanaan ujian tanpa konflik jadwal.

Contohnya, jam ujian Mata Kuliah A adalah {8,9} dan jadwal ujian Mata Kuliah B adalah {10,11}. Maka:

$$A \cap B = \{8,9\} \cap \{10,11\} = \emptyset$$



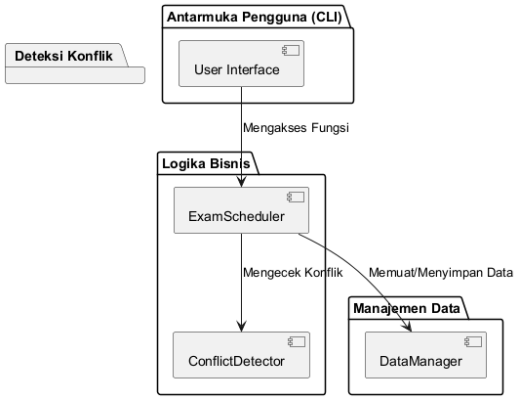
Gambar 2.6 Himpunan Saling Lepas
(Sumber: Dokumentasi Pribadi)

Karena irisan himpunan waktu kosong, kedua jadwal ini adalah disjoint dan tidak mengalami konflik waktu.

III. IMPLEMENTASI

A. Arsitektur Sistem

Arsitektur sistem penjadwalan ujian ini dirancang dengan pendekatan modular, di mana setiap komponen memiliki tanggung jawab tertentu dalam proses penjadwalan. Secara keseluruhan, arsitektur terdiri dari empat lapisan utama: Antarmuka Pengguna (CLI), Logika Bisnis (Business Logic), Manajemen Data, dan Deteksi Konflik. Antarmuka Pengguna menyediakan interaksi berbasis teks bagi pengguna untuk mengelola data penjadwalan, seperti menambah, menghapus, dan melihat jadwal ujian. Logika Bisnis mengelola operasi penjadwalan, termasuk validasi data, deteksi konflik, dan optimasi jadwal menggunakan operasi teori himpunan. Manajemen Data bertanggung jawab untuk menyimpan dan memuat data penjadwalan dari dan ke file JSON, memastikan keberlanjutan data antar sesi penggunaan. Deteksi Konflik memanfaatkan operasi teori himpunan untuk mengidentifikasi tumpang tindih waktu, penggunaan ruangan yang sama, atau konflik departemen.



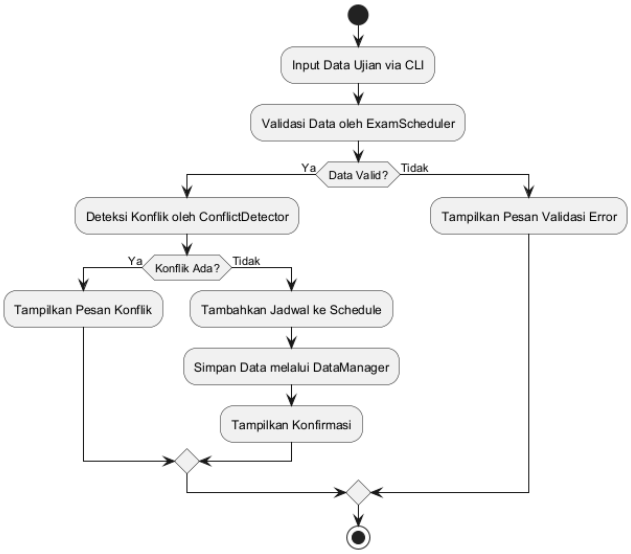
Gambar 3.1 Diagram Kelas Arsitektur Program ExamScheduler
(Sumber: Dokumentasi Pribadi)

Diagram arsitektur sistem penjadwalan ujian diilustrasikan pada Gambar 1. Diagram tersebut menunjukkan interaksi antara komponen utama, di mana Antarmuka Pengguna (CLI) berkomunikasi dengan kelas ExamScheduler. Kelas ExamScheduler selanjutnya berinteraksi dengan DataManager untuk manajemen data dan ConflictDetector untuk deteksi konflik.

B. Struktur dan Alur Data

Struktur data yang dipilih dalam sistem ini dirancang untuk mempermudah manipulasi dan akses data penjadwalan. Sistem menggunakan beberapa struktur data utama, yaitu dictionary dan list, yang masing-masing memiliki peran spesifik. *Dictionary self.schedule* menyimpan jadwal ujian yang diorganisasi berdasarkan kode mata kuliah, di mana setiap mata kuliah berisi daftar jadwal yang terdiri dari departemen, tempat, waktu mulai, waktu selesai, dan tanggal. *Dictionary self.courses* menyimpan asosiasi antara kode mata kuliah dan departemen, sementara list *self.time_slots* menyimpan daftar slot waktu yang tersedia dalam bentuk *dictionary*. Selain itu, list *self.places* dan *self.dates* masing-masing menyimpan daftar tempat atau ruangan ujian yang tersedia dan daftar tanggal yang diizinkan untuk pelaksanaan ujian.

Proses penjadwalan ujian dimulai dari input pengguna melalui antarmuka CLI. Data yang diinputkan kemudian diproses oleh *ExamScheduler*, yang melakukan validasi dan deteksi konflik sebelum akhirnya menyimpan data melalui *DataManager*. Alur data secara umum dimulai dari pengguna yang memasukkan data mata kuliah, departemen, tanggal, tempat, dan slot waktu melalui CLI. Selanjutnya, *ExamScheduler* memvalidasi format tanggal dan rentang waktu, kemudian *ConflictDetector* mengecek apakah jadwal baru menyebabkan konflik dengan jadwal yang sudah ada. Jika tidak ada konflik, data dijadikan bagian dari *self.schedule* dan disimpan melalui *DataManager*. Hasil penjadwalan atau pesan konflik kemudian ditampilkan kepada pengguna.



Gambar 3.2 Flowchart Input Data ExamScheduler
(Sumber: Dokumentasi Pribadi)

C. Implementasi Kelas ExamScheduler

```
import itertools
import json
import os
from datetime import datetime

class ExamScheduler:
    def __init__(self): ···

    def load_data(self, filename="data.json"): ···
    def save_data(self, filename="data.json"): ···

    def validate_date(self, date): ···
    def validate_time_slot(self, start, end): ···

    def add_course(self, course, department): ···

    def set_time_slots(self, slots): ···

    def set_places(self, places): ···
    def set_dates(self, dates): ···
    def add_schedule(self, course, date, place, start, end): ···
    def remove_schedule(self, course, index=None): ···

    def detect_conflicts(self): ···

    def show_schedules(self): ···
    def show_courses(self): ···
    def show_dates(self): ···
    def show_places(self): ···
    def show_time_slots(self): ···
```

Gambar 3.3 Kode Kerangka ExamScheduler
(Sumber: Dokumentasi Pribadi)

Kelas ExamScheduler merupakan inti dari sistem penjadwalan ujian, bertanggung jawab untuk memuat dan menyimpan data, menambah dan menghapus jadwal, serta mendeteksi konflik menggunakan operasi teori himpunan. Kelas ini diinisialisasi dengan struktur data kosong untuk jadwal, mata kuliah, slot waktu, tempat, dan tanggal. Metode *load_data* digunakan untuk memuat data penjadwalan dari file JSON jika tersedia, atau memulai dengan data kosong jika file tidak ditemukan. Metode *save_data* menyimpan data penjadwalan ke file JSON dengan format yang terstruktur, memastikan keberlanjutan data antar sesi penggunaan.

Validasi input dilakukan melalui *metode validate_date* dan *validate_time_slot*, yang memastikan bahwa tanggal yang diinputkan sesuai format YYYY-MM-DD dan slot waktu berada dalam rentang 0-24 jam dengan logis (start - end). Metode *add_course* memungkinkan penambahan mata kuliah beserta asosiasinya ke departemen, sementara metode *set_time_slots*, set_places, dan *set_dates* digunakan untuk mengatur daftar slot waktu, tempat, dan tanggal yang tersedia.

Penambahan jadwal ujian dilakukan melalui metode *add_schedule*, yang melakukan validasi data dan penambahan otomatis entitas yang belum terdaftar. Jika mata kuliah, tanggal, tempat, atau slot waktu belum terdaftar, sistem secara otomatis menambahkannya ke dalam daftar yang relevan. Metode *remove_schedule* memungkinkan penghapusan jadwal ujian berdasarkan indeks yang dipilih oleh pengguna, dengan validasi untuk memastikan indeks yang dipilih valid sebelum melakukan penghapusan. Metode *detect_conflicts* mendeteksi konflik jadwal menggunakan operasi himpunan seperti irisan waktu, kesamaan tempat, dan departemen. Selain itu, metode *show_schedules*, *show_courses*, *show_dates*, *show_places*, dan *show_time_slots* digunakan untuk menampilkan informasi jadwal, mata kuliah, tanggal, tempat, dan slot waktu kepada pengguna.

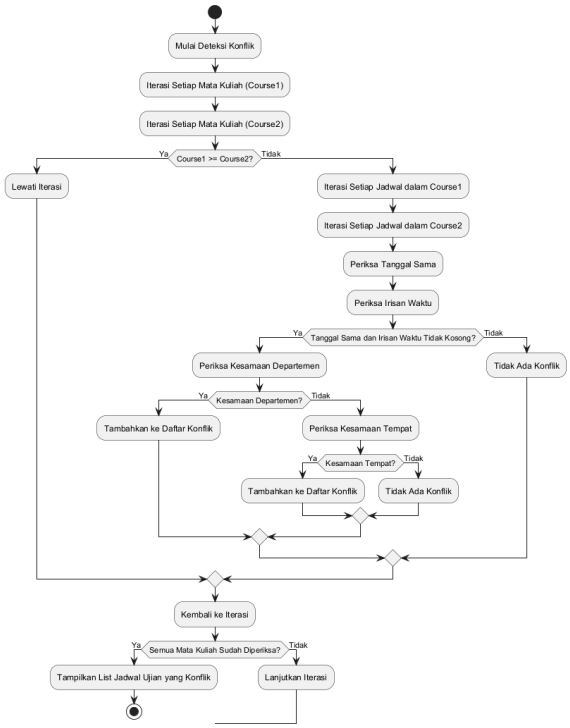
Metode *__init__* menginisialisasi objek ExamScheduler dengan struktur data kosong untuk jadwal, mata kuliah, slot waktu, tempat, dan tanggal. Metode *load_data* memuat data penjadwalan dari file JSON jika tersedia, atau memulai dengan data kosong jika file tidak ditemukan. Metode *save_data* menyimpan data penjadwalan ke file JSON dengan format yang terstruktur, memastikan keberlanjutan data antar sesi penggunaan.

Validasi input dilakukan melalui metode *validate_date* dan *validate_time_slot*, yang memastikan bahwa tanggal yang diinputkan sesuai format YYYY-MM-DD dan slot waktu berada dalam rentang 0-24 jam dengan logis (start - end). Metode *add_course* memungkinkan penambahan mata kuliah beserta asosiasinya ke departemen, sedangkan metode *set_time_slots*, *set_places*, dan *set_dates* digunakan untuk mengatur daftar slot waktu, tempat, dan tanggal yang tersedia.

Penambahan jadwal ujian dilakukan melalui metode *add_schedule*, yang melakukan validasi data dan penambahan otomatis entitas yang belum terdaftar. Jika mata kuliah, tanggal, tempat, atau slot waktu belum terdaftar, sistem secara otomatis menambahkannya ke dalam daftar yang relevan. Metode *remove_schedule* memungkinkan penghapusan jadwal ujian berdasarkan indeks yang dipilih oleh pengguna, dengan validasi untuk memastikan indeks yang dipilih valid sebelum melakukan penghapusan.

Metode *detect_conflicts* mendeteksi konflik jadwal menggunakan operasi himpunan seperti irisan waktu, kesamaan tempat, dan departemen (jurusan). Selain itu, metode *show_schedules*, *show_courses*, *show_dates*, *show_places*, dan *show_time_slots* digunakan untuk menampilkan informasi jadwal, mata kuliah, tanggal, tempat, dan slot waktu kepada pengguna.

D. Implementasi Deteksi Konflik



Gambar 3.4 Flowchart Deteksi Konflik Jadwal
(Sumber: Dokumentasi Pribadi)

Metode *detect_conflicts* melakukan iterasi melalui setiap pasangan mata kuliah dan jadwal ujian untuk mengidentifikasi konflik berdasarkan aturan yang telah ditetapkan. Dengan menggunakan operasi himpunan, sistem memeriksa irisan waktu antara dua jadwal. Jika terdapat irisan waktu dan tanggal yang sama, sistem kemudian memeriksa kesamaan departemen atau tempat. Konflik dianggap ada jika departemen sama tanpa memedulikan tempat, atau jika departemen berbeda namun menggunakan tempat yang sama dan waktu beririsan. Semua konflik yang terdeteksi ditambahkan ke dalam daftar *conflicts*, yang kemudian dikembalikan sebagai hasil metode ini.

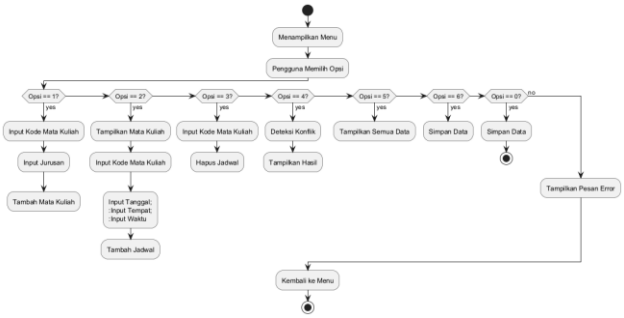
Misalkan terdapat dua jadwal ujian sebagai berikut:

- Jadwal *A*:
    - Departemen: Teknik Informatika
    - Tempat: R9601
    - Waktu: 08:00-10:00
    - Tanggal: 2024-01-10
- Jadwal *B*:
    - Departemen: Teknik Sipil
    - Tempat: R9601
    - Waktu: 09:00-11:00
    - Tanggal: 2024-01-10

Analisis konflik dilakukan dengan memeriksa apakah tanggal sama, apakah terdapat irisan waktu, dan apakah terdapat kesamaan departemen atau tempat. Dalam kasus ini, kedua jadwal berlangsung pada tanggal yang sama, terdapat irisan waktu pada jam 09:00, dan menggunakan tempat yang sama (R101). Meskipun departemen berbeda, konflik tetap terdeteksi karena tempat yang digunakan sama dan waktu beririsan.

E. Antarmuka Pengguna (CLI)

Antarmuka pengguna berbasis teks (*Command Line Interface*) memungkinkan pengguna untuk berinteraksi dengan sistem secara intuitif melalui menu-menu yang telah disediakan. Antarmuka ini menyediakan opsi untuk menambah, menghapus, melihat jadwal, mendeteksi konflik, dan menyimpan data. Struktur menu CLI dirancang untuk memudahkan navigasi pengguna dalam mengelola data penjadwalan.



Gambar 3.5 Flowchart Antarmuka Program
(Sumber: Dokumentasi Pribadi)

Diagram tersebut menggambarkan langkah-langkah yang diambil oleh sistem saat pengguna berinteraksi melalui menu pilihan, memastikan alur kerja yang terstruktur dan responsif terhadap pilihan pengguna.

Program terus menampilkan menu dan memproses pilihan pengguna hingga pengguna memilih untuk keluar. Setiap pilihan menu mengarahkan pengguna ke tindakan yang sesuai, seperti menambah mata kuliah, menambah jadwal, menghapus jadwal, mendeteksi konflik, melihat data, atau menyimpan data. Pilihan untuk keluar dari aplikasi juga memastikan bahwa data disimpan sebelum program diakhiri.

Menu pilihan menyediakan berbagai opsi bagi pengguna untuk mengelola data penjadwalan. Pilihan pertama memungkinkan pengguna untuk menambah mata kuliah baru beserta departemennya. Pilihan kedua memungkinkan pengguna untuk menambah jadwal ujian baru setelah memilih mata kuliah, tanggal, tempat, dan slot waktu. Pilihan ketiga memungkinkan pengguna untuk menghapus jadwal ujian berdasarkan kode mata kuliah. Pilihan keempat mendeteksi dan menampilkan konflik jadwal yang ada. Pilihan kelima menampilkan semua data yang telah diatur, termasuk mata kuliah, jadwal, tanggal, tempat, dan slot waktu. Pilihan keenam menyimpan data ke file JSON, dan pilihan nol memungkinkan pengguna untuk keluar dari aplikasi setelah menyimpan data.

Menu diulang menggunakan *while loop*. Setelah pengguna menyelesaikan tugas di menu tertentu, program akan menampilkan menu kepada pengguna supaya dapat menggunakan fitur yang lain. Loop tersebut berjalan secara terus menerus (*while True*), artinya tidak ada kondisi eksplisit yang menghentikan tampilan menu tersebut. Hal ini bertujuan supaya program terus menyediakan pilihan fitur kepada pengguna.

Satu-satunya bagian yang menghentikan *loop* adalah *break*. Break dijalankan oleh bagian menu *Exit* ketika user ingin keluar dari program. Jika *break* dijalankan, *loop* berhenti dan user keluar dari program tersebut.

```python
__name__ == "__main__":
scheduler = ExamScheduler()
scheduler.load_data()
def menu():
    print("\n=== Exam Scheduler CLI ===")
    print("1. Tambah Mata Kuliah")
    print("2. Tambah Jadwal")
    print("3. Hapus Jadwal")
    print("4. Deteksi Konflik")
    print("5. Lihat Data")
    print("6. Simpan Data")
    print("0. Keluar")
while True:
    menu()
    choice = input("\nPilih menu: ")
    if choice == "1":
        # Tambah mata kuliah
    elif choice == "2":
        # Tambah Jadwal
    elif choice == "3":
        # Hapus Jadwal
    elif choice == "4":
        # Deteksi Konflik
    elif choice == "5":
        # Lihat Data
    elif choice == "6":
        # Simpan Data
    elif choice == "0":
        print("\nKeluar dari aplikasi.")
        scheduler.save_data()
        break
    else:
        print("\nPilihan tidak valid.")
```

Gambar 3.6 Kode Sumber Menu pada Antarmuka
(Sumber: Dokumentasi Pribadi)

IV. EKSPERIMEN

Program ini akan dijalankan untuk mendeteksi konflik jadwal ujian. Beberapa kemungkinan konflik jadwal dimasukkan ke dalam database program. Lalu, fitur deteksi konflik dijalankan untuk mengecek apakah ada konflik di antara jadwal ujian yang diinput user.

Kode	Tanggal	Jam Mulai	Jam Selesai	Tempat	Jurusan
IF1220	10	8	11	9601	IF
IF2123	10	9	11	9601	IF
MA1120	10	10	12	9602	Math
IF2110	10	8	10	9602	IF
IF1221	11	13	15	9602	IF
EL1220	11	13	15	LabEL	Elektro
MS1220	11	14	16	Aula Barat	Mesin
FI1220	12	9	11	9603	Fisika
KI1120	12	9	11	9603	Kimia
KU1001	13	10	12	9605	Umum

Tabel 4.1 Jadwal Ujian
(Sumber: Dokumentasi Pribadi)

```
Konflik jadwal terdeteksi:
Konflik antara:
  Course 1: ('IF1220', {'department': 'IF', 'place': '9601', 'start': 8, 'end': 11, 'date': '2024-01-10'})
  Course 2: ('IF2123', {'department': 'IF', 'place': '9601', 'start': 9, 'end': 11, 'date': '2024-01-10'})
Konflik antara:
  Course 1: ('IF1220', {'department': 'IF', 'place': '9601', 'start': 8, 'end': 11, 'date': '2024-01-10'})
  Course 2: ('IF2110', {'department': 'IF', 'place': '9602', 'start': 8, 'end': 10, 'date': '2024-01-10'})
Konflik antara:
  Course 1: ('IF2110', {'department': 'IF', 'place': '9602', 'start': 8, 'end': 10, 'date': '2024-01-10'})
  Course 2: ('IF2123', {'department': 'IF', 'place': '9601', 'start': 9, 'end': 11, 'date': '2024-01-10'})
Konflik antara:
  Course 1: ('FI1220', {'department': 'Fisika', 'place': '9603', 'start': 9, 'end': 11, 'date': '2024-01-12'})
  Course 2: ('KI1220', {'department': 'Kimia', 'place': '9603', 'start': 9, 'end': 11, 'date': '2024-01-12'})
```

Gambar 4.1 *Screenshot* Hasil Deteksi Konflik
(Sumber: Dokumentasi Pribadi)

Hasil deteksi konflik dari program sistem penjadwalan ujian otomatis diuraikan sebagai berikut:

Konflik antara IF1220 dan IF2123:
Tanggal: 10
Jam: IF1220 (08:00-11:00) dan IF2123 (09:00-11:00)
Tempat: 9601
Jurusan: IF vs IF

Kedua mata kuliah ini dijadwalkan pada tanggal yang sama dengan waktu yang tumpang tindih dan menggunakan tempat yang sama. Selain itu, keduanya berasal dari jurusan yang sama (IF), yang meningkatkan potensi konflik karena keterbatasan sumber daya tempat ujian. Konflik ini sesuai dengan aturan yang telah ditetapkan dalam sistem, dimana tumpang tindih waktu dan penggunaan tempat yang sama dari jurusan yang sama dianggap sebagai konflik.\

Konflik antara IF1220 dan IF2110:
Tanggal: 10
Jam: IF1220 (08:00-11:00) dan IF2110 (08:00-10:00)
Tempat: 9601 vs 9602
Jurusan: IF vs IF

Kedua mata kuliah ini juga dijadwalkan pada tanggal yang sama dengan waktu yang tumpang tindih. Meskipun tempatnya berbeda, kesamaan jurusan (IF) menyebabkan konflik karena kapasitas departemen untuk mengelola ujian secara simultan terbatas. Hal ini menunjukkan bahwa sistem tidak hanya mempertimbangkan tempat dan waktu, tetapi juga relevansi departemen dalam deteksi konflik.

Konflik antara IF2110 dan IF2123:
Tanggal: 10
Jam: IF2110 (08:00-10:00) dan IF2123 (09:00-11:00)
Tempat: 9602 vs 9601
Jurusan: IF vs IF

Meskipun kedua mata kuliah ini menggunakan tempat yang berbeda, tumpang tindih waktu yang signifikan dan kesamaan jurusan (IF) menyebabkan konflik. Sistem berhasil mendeteksi konflik ini karena adanya irisan waktu meskipun tempat berbeda, yang relevan dalam konteks kapasitas dan ketersediaan sumber daya jurusan.

Konflik antara FI1220 dan KI1120:
Tanggal: 12
Jam: FI1220 (09:00-11:00) dan KI1120 (09:00-11:00)
Tempat: 9603
Jurusan: Fisika vs Kimia

Kedua mata kuliah ini dijadwalkan pada tanggal dan waktu yang sama, serta menggunakan tempat yang sama (9603), meskipun berasal dari jurusan yang berbeda. Konflik ini terdeteksi karena penggunaan tempat yang sama dan waktu

yang beririsan, yang merupakan salah satu kriteria konflik meskipun jurusan berbeda.

Selain empat konflik yang terdeteksi, sistem juga berhasil mengidentifikasi jadwal yang tidak menimbulkan konflik, seperti mata kuliah KU1001 pada tanggal 13 dengan waktu 10:00-12:00 di tempat 9605 dari jurusan Umum. Jadwal ini tidak mengalami tumpang tindih waktu atau penggunaan tempat yang sama dengan mata kuliah lain, sehingga tidak menimbulkan konflik sesuai dengan aturan yang ditetapkan.

Hasil deteksi konflik ini menunjukkan bahwa sistem penjadwalan ujian otomatis berbasis teori himpunan mampu mengidentifikasi konflik secara akurat berdasarkan aturan yang telah ditetapkan. Sistem tidak hanya mempertimbangkan tumpang tindih waktu dan penggunaan tempat yang sama, tetapi juga kesamaan jurusan yang dapat mempengaruhi kapasitas pengelolaan ujian. Keempat konflik yang terdeteksi sesuai dengan analisis yang diharapkan berdasarkan kombinasi jadwal yang telah dimasukkan, menunjukkan bahwa sistem berfungsi dengan baik dalam mengelola kompleksitas penjadwalan ujian.

Lebih lanjut, eksperimen ini juga menegaskan bahwa sistem mampu membedakan antara jadwal yang menyebabkan konflik dan yang tidak, memastikan bahwa jadwal yang dihasilkan bebas dari konflik sesuai dengan kebijakan institusi. Sistem juga menunjukkan fleksibilitas dalam menangani berbagai jenis konflik, baik yang disebabkan oleh tumpang tindih waktu, penggunaan tempat yang sama, maupun kesamaan jurusan. Dengan demikian, pendekatan berbasis teori himpunan yang diterapkan dalam sistem ini terbukti efektif dalam mengelola penjadwalan ujian yang kompleks dan memastikan kelancaran proses administrasi akademik.

V. KESIMPULAN

Dalam penelitian ini, telah dikembangkan dan diimplementasikan sistem penjadwalan ujian otomatis berbasis teori himpunan yang efektif dalam mendeteksi konflik jadwal. Melalui eksperimen yang dilakukan, sistem berhasil mengidentifikasi berbagai jenis konflik, baik yang disebabkan oleh tumpang tindih waktu, penggunaan tempat yang sama, maupun kesamaan jurusan. Hasil eksperimen menunjukkan bahwa pendekatan berbasis teori himpunan mampu mengelola kompleksitas penjadwalan ujian dengan akurat, memastikan bahwa jadwal yang dihasilkan bebas dari konflik sesuai dengan ketentuan yang ditetapkan. Hal ini membuktikan bahwa sistem ini dapat diandalkan untuk digunakan dalam lingkungan akademik yang memerlukan penjadwalan ujian yang efisien dan terstruktur.

Salah satu peningkatan yang dapat dilakukan pada sistem ini adalah penambahan komponen menit pada jam mulai dan jam selesai. Dengan memasukkan komponen menit, penjadwalan dapat menjadi lebih presisi, mengurangi kemungkinan terjadinya tumpang tindih waktu yang lebih kecil namun tetap signifikan. Hal ini akan meningkatkan akurasi deteksi konflik dan memberikan fleksibilitas lebih dalam pengaturan jadwal ujian, terutama pada institusi dengan jumlah ujian yang padat dan waktu yang ketat.

Selain itu, beberapa saran untuk pengembangan sistem di masa mendatang meliputi integrasi dengan sistem informasi akademik yang lebih luas untuk memudahkan pengelolaan data secara *real-time* dan otomatis. Penambahan fitur notifikasi otomatis kepada pengguna ketika terjadi konflik juga dapat meningkatkan responsivitas dan efisiensi dalam penanganan jadwal. Penggunaan algoritma optimasi yang lebih canggih dapat diterapkan untuk meningkatkan performa sistem, terutama saat menangani dataset yang lebih besar dan kompleks. Terakhir, pengembangan antarmuka pengguna yang lebih interaktif dan ramah pengguna, seperti berbasis web atau aplikasi *mobile*, dapat memperluas aksesibilitas dan memudahkan pengguna dalam mengelola jadwal ujian.

Dengan demikian, sistem penjadwalan ujian otomatis berbasis teori himpunan ini tidak hanya berhasil memenuhi tujuan awalnya dalam mendeteksi konflik jadwal, tetapi juga menunjukkan potensi untuk pengembangan lebih lanjut guna meningkatkan presisi dan fungsionalitasnya. Implementasi dan pengembangan lebih lanjut dari sistem ini diharapkan dapat memberikan kontribusi signifikan dalam pengelolaan administrasi akademik yang lebih efisien dan efektif.

VI. PENUTUP

Puji Syukur kepada Tuhan Yang Maha Esa atas rahmat-Nya dalam pembuatan makalah IF1220 Matematika Diskrit dengan topik “Optimasi Penjadwalan Ujian di Perguruan Tinggi Berbasis Operasi Teori Himpunan”. Saya mengucapkan terima kasih kepada dosen IF1220 beserta para penulis yang tulisannya menjadi referensi dalam makalah ini.

REFERENSI

[1] R. Munir, “Informatika ITB,” 2016. [Online]. Available: https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2016-2017/Himpunan-%282016%29.pdf. [Diakses 6 Januari 2025].

[2] P. Prof. S.M. Nababan dan M. Drs. Warsito, “Pustaka UT,” 2016. [Online]. Available: https://pustaka.ut.ac.id/lib/wp-content/uploads/pdfmk/MATA4101-M1.pdf. [Diakses 6 Januari 2025].

[3] C. Jongsma, Basic Set Theory and Combinatorics, Cham: Springer, 2019.

[4] BYJU'S, "Set Theory," BYJU'S, 13 Februari 2021. [Online]. Available: https://byjus.com/maths/basics-set-theory/. [Accessed 6 Januari 2025].

[5] Z. K. O. Mujgan Sagir, “Exam scheduling: Mathematical modeling and parameter estimation,” *Mathematical and Computer Modelling,* vol. 52, no. 5-6, pp. 930-941, 2010.

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 6 Januari 2025

Jovandra Otniel P. S. (13523141)