Syntax:
Journal of Software Engineering, Computer Science and Information Technology
p-ISSN: 2776-7027, e-ISSN: 2723-0538
Volume: 7, Nomor: 1, Juni 2026

PENERAPAN DYNAMIC PROGRAMMING PADA PENJADWALAN PENERBANGAN UNTUK MINIMASI DELAY

Ariel Sudarsono[1], Raphael Lee[2], Yohannes[3]

[1,2,3] *Universitas Multi Data Palembang*
Jl. Rajawali No.14, 9 Ilir, Kec. Ilir Tim. II, Kota Palembang, Sumatera Selatan 30113
[1]arielsudarsono@gmail.com, [2]gregoriusraphaellee@gmail.com, [3]yohannesmasterous@mdp.ac.id

ABSTRAK

Keterlambatan penerbangan komersial menjadi tantangan besar dalam manajemen operasional bandara yang berdampak pada efisiensi sistem. Penelitian ini bertujuan mengoptimalkan penjadwalan penerbangan untuk meminimalkan total delay menggunakan algoritma Dynamic Programming. Eksperimen dilakukan menggunakan dataset maskapai penerbangan dari Kaggle dengan membatasi ruang lingkup pada 15.000 baris data pertama dan kapasitas waktu operasional harian sebesar 1.440 menit. Parameter weight ditentukan berdasarkan durasi terbang, sedangkan value dibentuk menggunakan fungsi penalti keterlambatan. Tahap post-processing diterapkan dengan aturan celah waktu minimum 20 menit untuk mengeliminasi konflik rute. Hasil penelitian menunjukkan algoritma berhasil menyusun kombinasi jadwal final berisi 16 penerbangan optimal. Penerapan metode ini terbukti efektif menekan total keterlambatan menjadi 107,0 menit dengan rata-rata delay sebesar 6,68 menit per penerbangan.

***Kata Kunci*—Dynamic Programming, Keterlambatan Penerbangan, Optimasi Jadwal, Post-processing.**

ABSTRACT

Commercial flight delays present a major challenge in airport operational management, impacting system efficiency. This study aims to optimize flight scheduling to minimize total delays using Dynamic Programming. Experiments were conducted using an airline dataset from Kaggle, limiting the scope to the first 15,000 data rows with a daily operational time capacity of 1,440 minutes. The weight parameter was determined based on flight duration, while the value was formulated using a delay penalty function. A post-processing stage applied a minimum 20-minute time gap rule to eliminate route conflicts. The results show that the algorithm successfully generated a final schedule combination of 16 optimal flights. This approach effectively reduced the total delay to 107,0 minutes, achieving an average delay of only 6,68 minutes per flight.

Keywords—Dynamic Programming, Flight Delays, Post-processing, Schedule Optimization.

Syntax:
Journal of Software Engineering, Computer Science and Information Technology
p-ISSN: 2776-7027, e-ISSN: 2723-0538
Volume: 7, Nomor: 1, Juni 2026

I. PENDAHULUAN

Transportasi udara merupakan salah satu moda transportasi yang memiliki peranan penting dalam mendukung mobilitas masyarakat serta distribusi barang secara cepat dan efisien [1]. Seiring meningkatnya aktivitas penerbangan, pengelolaan jadwal penerbangan dan alokasi slot bandara menjadi tantangan yang semakin kompleks bagi maskapai maupun pihak bandara. Kompleksitas tersebut dipengaruhi oleh tingginya kepadatan ruang udara serta keterbatasan kapasitas bandara dan ruang udara yang berpotensi menimbulkan berbagai ketidakefisienan operasional dalam sistem lalu lintas penerbangan [2]. Kondisi tersebut dapat menyebabkan ketidaktepatan dalam penyusunan jadwal penerbangan sehingga meningkatkan risiko terjadinya keterlambatan penerbangan (*delay*). Keterlambatan ini berdampak pada kerugian operasional, penurunan kepuasan penumpang, serta terganggunya jadwal penerbangan lainnya [3].

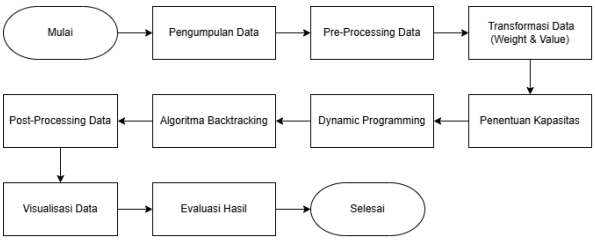
Delay penerbangan dapat dipengaruhi oleh berbagai faktor, seperti kepadatan bandara, konflik jadwal penerbangan, kondisi operasional maskapai, serta propagasi keterlambatan antar penerbangan yang saling terhubung [4], [5]. Dalam sistem transportasi udara yang saling terintegrasi, perubahan kecil pada satu jadwal penerbangan dapat memicu keterlambatan pada penerbangan lain sehingga menyebabkan propagasi delay ke berbagai bandara dan meningkatkan kompleksitas pengelolaan lalu lintas udara [6]. Oleh karena itu, diperlukan pendekatan komputasional yang mampu menentukan solusi penjadwalan secara optimal berdasarkan berbagai kemungkinan kombinasi jadwal penerbangan.

Salah satu metode yang dapat digunakan untuk menyelesaikan permasalahan optimasi jadwal penerbangan adalah algoritma *Dynamic Programming*. *Dynamic Programming* merupakan metode optimasi yang bekerja dengan memecah permasalahan kompleks menjadi submasalah yang lebih kecil, kemudian menyimpan hasil penyelesaian submasalah tersebut agar dapat digunakan kembali tanpa melakukan perhitungan berulang [7]. Metode ini dinilai efektif untuk menyelesaikan permasalahan penjadwalan karena mampu mencari solusi optimal dengan mempertimbangkan efisiensi waktu dan minimasi *delay* pada setiap interval penerbangan [8].

Beberapa penelitian sebelumnya telah menerapkan metode optimasi dalam bidang penjadwalan transportasi dan menunjukkan hasil yang cukup baik dalam meningkatkan efisiensi sistem [9] [10] [11]. Namun, penelitian terkait penerapan *Dynamic Programming* secara khusus pada optimasi jadwal penerbangan untuk minimasi delay masih relatif terbatas. Oleh karena itu, penelitian ini dilakukan untuk menganalisis penerapan algoritma *Dynamic Programming* dalam mengoptimalkan jadwal penerbangan sehingga dapat meminimalkan tingkat keterlambatan penerbangan dan meningkatkan efisiensi operasional transportasi udara.

II. METODOLOGI PENELITIAN

Penelitian ini menggunakan metode kuantitatif dengan pendekatan komputasional untuk mengoptimalkan jadwal penerbangan menggunakan algoritma *Dynamic Programming*. Fokus penelitian adalah meminimalkan delay penerbangan berdasarkan data jadwal dan informasi penerbangan yang diperoleh dari dataset penerbangan pada platform Kaggle [12]. Proses penelitian dapat dilihat pada Gambar 1.



Gbr 1. Alur Penelitian

A. *Pengumpulan Data*

Data dalam penelitian ini bersumber dari dataset penerbangan *"Airline Flight Dataset: Schedule, Performance etc"* yang diperoleh melalui platform Kaggle. Dataset ini mencakup informasi mengenai jadwal serta performa realisasi penerbangan. Dari keseluruhan data yang tersedia, dilakukan tahapan seleksi fitur (*feature selection*) untuk mengambil variabel-variabel yang relevan dengan tujuan penelitian. Variabel yang dipilih meliputi nomor penerbangan *(FlightNum),* waktu keberangkatan (*DepTime*), waktu kedatangan (*ArrTime*), keterlambatan keberangkatan (*DepDelay*), bandara asal (*Origin*), bandara tujuan (*Dest*), dan jarak penerbangan (*Distance*).

B. *Preprocessing Data*

Data yang telah diperoleh akan diproses melalui tahap *preprocessing* data yang berperan penting dalam meningkatkan kualitas, konsistensi, dan keandalan data sehingga dapat menghasilkan performa model yang lebih stabil, akurat, dan efisien dalam proses analisis maupun optimasi [13]. Tahapan ini diawali dengan pembersihan data melalui penghapusan nilai yang hilang *(missing values)* menggunakan metode *dropna().* Selanjutnya, dilakukan transformasi format waktu pada variabel *DepTime* dan *ArrTime* yang semula berbentuk format jam-menit numerik (HH:MM) dikonversi ke

Syntax:
Journal of Software Engineering, Computer Science and Information Technology
p-ISSN: 2776-7027, e-ISSN: 2723-0538
Volume: 7, Nomor: 1, Juni 2026

dalam satuan total menit sejak tengah malam melalui Persamaan 1.

$$Total\ Menit = (Jam \times 60) + Menit \quad (1)$$

Apabila ditemukan kondisi di mana waktu kedatangan lebih kecil dari waktu keberangkatan (*end* < *start*), hal tersebut mengindikasikan penerbangan melewati tengah malam, sehingga waktu kedatangan ditambah dengan 1.440 menit (24 jam) untuk menjaga konsistensi data. Selain itu, nilai negatif pada variabel *DepDelay* disesuaikan menjadi 0 untuk merepresentasikan penerbangan yang berangkat tepat waktu atau lebih awal. Tahap akhir dari preprocessing adalah melakukan pengurutan data secara kronologis berdasarkan waktu kedatangan (*end*) untuk mempersiapkan struktur data yang sesuai dengan kebutuhan algoritma penjadwalan.

C. Transformasi Data (Weight & Value)

Pada tahap transformasi, data penerbangan dikonversi menjadi parameter-parameter spesifik yang sesuai dengan karakteristik pemodelan algoritma *Dynamic Programming*. Proses transformasi data ini didefinisikan ke dalam dua parameter utama, yaitu:

1. *Weight* (bobot), merepresentasikan durasi total penerbangan dalam satuan menit, yang dihitung dari selisih antara waktu kedatangan (*end*) dan waktu keberangkatan (*start*).
2. *Value* (prioritas), merepresentasikan bobot prioritas penerbangan yang berorientasi pada minimasi efek keterlambatan dan efisiensi durasi. Nilai ini dapat dihitung menggunakan Persamaan 2.

$$Value = 1000 - (DepDelay \times 2) - (Weight \times 0.5) \quad (2)$$

Melalui fungsi tujuan ini, penerbangan dengan tingkat keterlambatan (*DepDelay*) yang minimal dan durasi penerbangan (*Weight*) yang lebih singkat akan memperoleh nilai optimasi yang lebih tinggi untuk diprioritaskan.

D. Penentuan Kapasitas

Kapasitas dalam penelitian ini ditentukan berdasarkan batasan total waktu operasional siklus harian penerbangan, yaitu sebesar 1.440 menit (setara dengan 24 jam). Kapasitas ini bertindak sebagai batas akumulasi bobot (*weight*) maksimal dalam ruang keputusan *Dynamic Programming.* Melalui batasan ini, algoritma akan mencari kombinasi jadwal penerbangan yang menghasilkan total *value* terbesar tanpa melampaui batas waktu operasional harian yang telah ditetapkan.

E. Penerapan Algoritma Dynamic Programming

Algoritma *Dynamic Programming* diterapkan untuk membangun matriks keputusan M berukuran $(n+1) \times (W+1)$, di mana n adalah jumlah data penerbangan (15.000 data) dan W adalah kapasitas waktu (1.440 menit). Sesuai dengan prinsip pemecahan sub-masalah secara bertahap, implementasi kode dilakukan melalui pendekatan tabulasi (*tabulation*), di mana hasil kalkulasi setiap sub-masalah jadwal disimpan ke dalam *array* multidimensi untuk menghindari komputasi berulang [14]. Secara rekursif, setiap kandidat penerbangan dianalisis dengan membandingkan opsi untuk mengabaikan penerbangan tersebut atau mengambilnya dengan konsekuensi mengurangi sisa kapasitas waktu yang tersedia, demi mendapatkan akumulasi nilai prioritas (*value*) terbesar. *Pseudocode* algoritma *Dynamic Programming* dapat dilihat pada Gambar 2.

```
M = np.zeros((n + 1, W + 1))
for w in range(W + 1):
    M[0, w] = 0

for i in range(1, n + 1):
    M[i, 0] = 0
for i in range(1, n + 1):
    for w in range(1, W + 1):
        wi = weights[i - 1]
        vi = values[i - 1]

        if wi > w:
            M[i, w] = M[i - 1, w]
        else:
            M[i, w] = max(
                M[i - 1, w],
                M[i - 1, w - wi] + vi
            )
```

Gbr 2. *Dynamic Programming* Optimasi Jadwal Penerbangan

F. Penerapan Algoritma Penelusuran Balik

Algoritma penelusuran balik pada tahap ini berfungsi sebagai mekanisme pencarian jalur solusi optimum (*traceback*) dengan cara mengeksplorasi ruang keputusan yang telah dibentuk oleh matriks pemrograman dinamis secara sistematis [15]. Setelah proses pengisian matriks optimasi *Dynamic Programming* mencapai hasil akhir, algoritma ini diterapkan untuk menelusuri kembali jalur keputusan di dalam matriks M. Proses penelusuran dilakukan secara mundur (*backward search*) mulai dari indeks matriks tertinggi $M[n, W]$. Jika nilai pada baris saat ini lebih besar daripada nilai pada baris sebelumnya ($\mathrm{M[i,k]} > \mathrm{M[i-1,k]}$), maka indeks penerbangan tersebut dicatat sebagai bagian dari solusi optimal. Selanjutnya, sisa kapasitas kolom (k) dikurangi sebesar bobot (*weight*) dari penerbangan yang terpilih. Hasil penelusuran mundur ini kemudian dibalik (*reverse*) untuk membentuk urutan kronologis jadwal penerbangan

Syntax:
Journal of Software Engineering, Computer Science and Information Technology
p-ISSN: 2776-7027, e-ISSN: 2723-0538
Volume: 7, Nomor: 1, Juni 2026

optimal awal. *Pseudocode* algoritma penelusuran balik dapat dilihat pada Gambar 3.

```
selected = []

i = n
k = W

while i > 0 and k > 0:
    if M[i, k] > M[i - 1, k]:
        selected.append(i - 1)
        i = i - 1
        k = k - weights[i]
    else:
        i = i - 1
```

Gbr 3. Penelusuran Balik Pemilihan Solusi Optimal

G. Post-processing Data

Jadwal penerbangan yang telah dihasilkan dari proses penelusuran balik, selanjutnya disaring kembali melalui tahap *post-processing* untuk memastikan aspek keselamatan dan kelayakan operasional. Pada tahap ini, dilakukan proses filtering berbasis aturan (*rule-based filtering*) dengan menerapkan batas celah waktu minimum (*MIN_GAP*) sebesar 20 menit. Proses ini mendeteksi kemungkinan konflik jadwal (tumpang-tindih) pada penerbangan yang memiliki rute yang sama (*Origin* dan *Dest*). Jika ditemukan beberapa penerbangan pada rute yang sama dengan interval waktu yang terlalu berdekatan (kurang dari 20 menit), maka penerbangan yang berpotensi menimbulkan konflik akan dieliminasi melalui proses *filtering*, sehingga hanya penerbangan yang memenuhi kriteria batas jeda aman yang dipertahankan. Proses ini menghasilkan jadwal akhir yang lebih realistis dan lebih stabil untuk diterapkan pada kondisi operasional penerbangan sebenarnya.

H. Visualisasi Data

Hasil optimasi jadwal penerbangan akhir ditransformasikan kembali ke dalam format waktu standar (HH:MM) agar mudah dipahami. Data ini kemudian divisualisasikan menggunakan pustaka *Matplotlib* ke dalam bentuk grafik batang. Visualisasi berfokus pada analisis komparatif untuk menampilkan perbandingan performa rata-rata keterlambatan (*average delay*) antara kondisi sebelum optimasi (*baseline data*) dan kondisi sesudah proses optimasi dilakukan. Tahap ini bertujuan memberikan representasi visual yang jelas mengenai tingkat keberhasilan reduksi *delay*.

I. Evaluasi Data

Evaluasi dilakukan untuk mengukur performa, efektivitas, dan unjuk kerja dari integrasi algoritma *Dynamic Programming* dalam mengoptimalkan jadwal penerbangan. Proses evaluasi dilakukan menggunakan metode analisis komparatif kuantitatif dengan membandingkan parameter performa antara kondisi data aktual (*baseline*) sebelum optimasi dan hasil jadwal rekomendasi setelah proses optimasi dilakukan [16]. Efektivitas algoritma yang diterapkan dievaluasi berdasarkan empat parameter utama pada Tabel 1.

TABEL I
PARAMETER EVALUASI HASIL OPTIMASI JADWAL PENERBANGAN

No.	Parameter Evaluasi	Deskripsi
1.	Total *Delay* Penerbangan	Mengukur total waktu keterlambatan dari seluruh penerbangan yang dipilih oleh algoritma dalam solusi optimal.
2.	Rata-rata *Delay* Penerbangan	Menilai efisiensi penjadwalan secara individual dengan menghitung rata-rata waktu penundaan per penerbangan untuk melihat konsistensi perbaikan jadwal.
3.	Jumlah Penerbangan Terjadwal	Menilai kemampuan algoritma dalam mengakomodasi jumlah penerbangan sebanyak mungkin tanpa melampaui batas kapasitas waktu operasional harian yang telah ditentukan.
4.	Jumlah Konflik Jadwal	Memverifikasi tingkat pengurangan konflik jadwal pada rute yang sama berdasarkan penerapan batas celah waktu minimum (*MIN_GAP*) pada tahap post-processing guna mengurangi potensi tumpang-tindih operasional penerbangan.

III. HASIL DAN PEMBAHASAN

Pada penelitian ini, dari total keseluruhan data, analisis dibatasi pada 15.000 baris data pertama untuk menguji performa algoritma.

A. Hasil Preprocessing dan Transformasi Data

Pada tahap *preprocessing*, pembersihan data menggunakan metode *dropna()* berhasil mengeliminasi baris yang memiliki nilai hilang (*missing values*). Proses konversi waktu dari format HHMM menjadi satuan menit sejak tengah malam menghasilkan variabel baru yaitu *start* (waktu keberangkatan) dan *end* (waktu kedatangan). Untuk data yang mengalami penundaan negatif pada variabel *DepDelay* (berangkat lebih awal), nilainya disesuaikan menjadi 0 menit untuk menjaga validitas perhitungan penalti.

Setelah data dibersihkan, tahap transformasi dilakukan untuk membentuk parameter *Weight* (bobot) dan *Value* (prioritas) dengan luaran lima sampel data pertama yang disajikan pada Tabel 2.

Syntax:
Journal of Software Engineering, Computer Science and Information Technology
p-ISSN: 2776-7027, e-ISSN: 2723-0538
Volume: 7, Nomor: 1, Juni 2026

TABEL II
HASIL TRANSFORMASI DATA

Flight Num	*Dep Delay*	*Start*	*End*	*Weight*	*Value*
2418	0.0	1146	1185	39	980.5
2418	0.0	1145	1184	39	980.5
2956	0.0	1370	1409	39	980.5
2418	0.0	1151	1191	40	980.0
2418	0.0	1148	1188	40	980.0

Pada Tabel 2, variabel *DepDelay, Start, End, dan Weight* dinyatakan dalam satuan menit. Sementara itu, Nilai *Value* dihitung berdasarkan Persamaan 2.

Fungsi tersebut dirancang untuk memberikan prioritas lebih tinggi kepada penerbangan dengan tingkat keterlambatan yang rendah serta durasi penerbangan yang lebih efisien. Semakin kecil nilai *DepDelay* dan *Weight*, maka nilai *Value* yang dihasilkan akan semakin besar. Dengan demikian, proses optimasi menggunakan algoritma *Dynamic Programming* akan cenderung memilih penerbangan yang memiliki risiko keterlambatan lebih rendah dan penggunaan waktu yg lebih optimal.

B. Optimasi Jadwal Menggunakan Dynamic Programming dan Penelusuran balik

Proses optimasi dilakukan dengan menerapkan algoritma *Dynamic Programming* tipe 0/1 *Knapsack* dengan memetakan total kapasitas waktu operasional harian (*W*) sebesar 1440 menit. Algoritma bekerja secara tabulasi dengan membangun matriks keputusan berukuran 15001×1441. Berdasarkan hasil pengisian matriks keputusan, diperoleh Nilai Optimal Maksimum sebesar 34931.0. Nilai ini merepresentasikan akumulasi skor prioritas tertinggi yang bisa dicapai dalam batasan kapasitas waktu harian.

Setelah nilai optimal ditemukan, algoritma penelusuran balik dijalankan secara mundur untuk menelusuri kembali indeks penerbangan yang berkontribusi terhadap nilai optimal tersebut. Proses ini berhasil memilih kombinasi jadwal penerbangan awal yang kemudian diurutkan secara kronologis berdasarkan waktu keberangkatan (*start*).

C. Analisis Post-processing

Meskipun algoritma *Dynamic Programming* telah menghasilkan kombinasi jadwal dengan *value* tertinggi, jadwal tersebut belum sepenuhnya aman dari aspek operasional penerbangan nyata. Oleh karena itu, tahap *post-processing* diterapkan menggunakan aturan penyaringan jarak waktu minimum (*MIN_GAP = 20* menit).

Proses *post-processing* diawali dengan pemindaian terhadap jadwal yang dihasilkan dari tahap penelusuran balik. Jika ditemukan penerbangan dengan rute yang sama (*Origin* dan *Dest*) yang memiliki jeda waktu kurang dari 20 menit terhadap penerbangan sebelumnya, maka penerbangan yang berpotensi menimbulkan konflik akan dieliminasi. Proses filtering ini menghasilkan jadwal yang lebih renggang, realistis, dan sesuai untuk diterapkan pada kondisi operasional bandara yang sebenarnya.

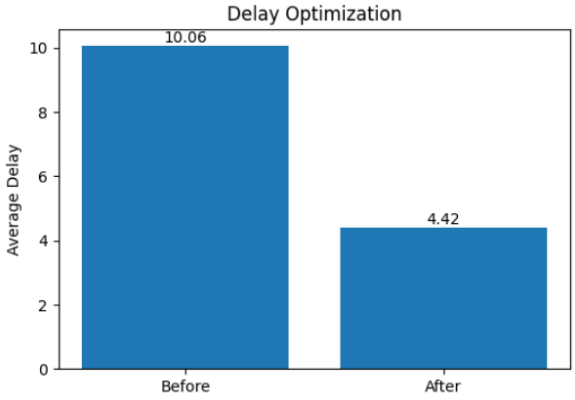
D. Evaluasi Performa Hasil Optimasi

Untuk mengukur efektivitas dan unjuk kerja algoritma yang diusulkan, dilakukan analisis komparatif kuantitatif antara kondisi data sebelum optimasi (*baseline*) dan sesudah optimasi (*post-processed schedule*).

Dari 15.000 data yang dievaluasi, algoritma berhasil menyusun kombinasi jadwal final yang terdiri dari 12 penerbangan optimal. Melalui penerapan proses *filtering* pada tahap *post-processing*, potensi konflik jadwal pada penerbangan dengan rute yang sama berhasil dikurangi melalui penerapan batas jeda aman minimum (*MIN_GAP*) sebesar 20 menit, sehingga jadwal yang dihasilkan menjadi lebih aman dan realistis untuk diterapkan pada kondisi operasional penerbangan.

Keberhasilan utama dari integrasi algoritma ini terlihat pada penurunan drastis tingkat keterlambatan penerbangan. Jadwal final yang direkomendasikan berhasil menekan Total *Delay* menjadi hanya 53,00 menit, dengan pencapaian Rata-rata *Delay* sebesar 4.42 menit per penerbangan.

Perbandingan performa rata-rata keterlambatan sebelum dan sesudah optimasi diilustrasikan pada Gambar 4.



Gbr 4. Perbandingan performa rata-rata keterlambatan

Berdasarkan grafik pada Gambar 4, terlihat bahwa rata-rata keterlambatan pada kondisi sebelum optimasi (*Before*) bernilai jauh lebih tinggi akibat menumpuknya data jadwal penerbangan komersial yang mengalami penundaan. Sebaliknya, setelah algoritma diimplementasikan (*After*), rata-rata keterlambatan menurun secara signifikan ke angka 4,42 menit.

Syntax:
Journal of Software Engineering, Computer Science and Information Technology
p-ISSN: 2776-7027, e-ISSN: 2723-0538
Volume: 7, Nomor: 1, Juni 2026

Hal ini membuktikan bahwa fungsi tujuan matematika yang dirancang pada parameter *Value* berhasil mengarahkan algoritma *Dynamic Programming* untuk memilih penerbangan-penerbangan yang memiliki tingkat efisiensi waktu terbaik.

Rincian kombinasi 12 penerbangan optimal hasil proses *filtering* akhir beserta informasi rute dan waktu operasional penerbangan disajikan secara lengkap pada Tabel 3.

TABEL III
HASIL AKHIR PENJADWALAN PENERBANGAN

Flight Num	*Origin*	*Dest*	*Departure*	*Arrival*	*Dep Delay*
579	HOU	AUS	07:48	08:29	0.00
979	HOU	AUS	08:56	09:34	6.00
3013	HOU	AUS	09:54	10:34	0.00
3848	HOU	AUS	13:04	13:45	14.00
3284	HOU	AUS	14:06	14:47	0.00
1024	HOU	AUS	15:45	16:25	15.00
563	HOU	CRP	17:44	18:25	4.00
791	HOU	AUS	17:49	18:27	14.00
2418	IAH	LCH	19:05	19:44	0.00
3665	HOU	AUS	20:28	21:09	0.00
1647	IAH	AUS	20:59	21:41	0.00
2956	IAH	AUS	22:50	23:29	0.00
				Total *Delay*	53.00
				Rata-Rata *Delay*	4.42
				Jumlah *Flight*	12.00
				Jumlah Konflik	2.00

Berdasarkan data pada Tabel 3, diperoleh beberapa temuan penting terkait karakteristik operasional jadwal penerbangan hasil optimasi sebagai berikut:

1. Penyebaran waktu operasional**:** Algoritma berhasil menyusun jadwal penerbangan yang tersebar secara konsisten sepanjang hari, dimulai dari penerbangan paling awal pada pukul 07:48 (*FlightNum* 579) hingga penerbangan terakhir pada malam hari pukul 22:50 (*FlightNum* 2956). Hal ini menunjukkan bahwa kapasitas waktu operasional harian sebesar 1.440 menit dapat dimanfaatkan secara efektif tanpa menyebabkan penumpukan penerbangan pada rentang waktu tertentu.
2. Analisis jeda aman (*safety gap*): Implementasi aturan *MIN_GAP* = 20 menit pada tahap *post-processing* terbukti mampu mengurangi potensi konflik jadwal pada rute yang sama. Sebagai contoh, pada rute HOU → AUS, selisih antara waktu kedatangan *FlightNum* 579 (08:29) dan waktu keberangkatan *FlightNum* 979 (08:56) adalah 27 menit, sehingga memenuhi batas jeda minimum yang telah ditetapkan. Selain itu, beberapa penerbangan dengan waktu operasional yang berdekatan tetap dapat dipertahankan karena memiliki rute yang berbeda, sehingga tidak menimbulkan konflik operasional secara langsung.
3. Karakteristik efisiensi keterlambatan (*DepDelay*): Dari 12 penerbangan yang terpilih, terdapat 6 penerbangan (50% dari total jadwal) yang memiliki nilai keterlambatan sebesar 0.00 menit, yaitu *FlightNum* 579, 3013, 3284, 2418, 3665, dan 2956. Kondisi ini menunjukkan bahwa algoritma cenderung memprioritaskan penerbangan dengan tingkat keterlambatan rendah. Sementara itu, keterlambatan tertinggi pada jadwal akhir tercatat sebesar 15.00 menit pada *FlightNum* 1024.
4. Analisis total *delay* penerbangan: Berdasarkan hasil optimasi pada Tabel 3, total akumulasi keterlambatan (*Total Delay*) dari seluruh penerbangan yang terpilih adalah sebesar 53.00 menit. Nilai ini menunjukkan bahwa algoritma berhasil menghasilkan kombinasi jadwal dengan tingkat keterlambatan kumulatif yang relatif rendah. Penerapan fungsi *Value* yang mempertimbangkan penalti keterlambatan dan durasi penerbangan terbukti efektif dalam mengarahkan proses optimasi menuju solusi dengan tingkat keterlambatan yang lebih kecil.
5. Analisis rata-rata delay penerbangan: Nilai rata-rata keterlambatan (*Average Delay*) yang diperoleh sebesar 4.42 menit per penerbangan menunjukkan tingkat efisiensi jadwal yang cukup baik. Rendahnya nilai rata-rata ini mengindikasikan bahwa sebagian besar penerbangan yang dipilih memiliki performa waktu yang stabil dan hanya mengalami keterlambatan dalam jumlah kecil. Hasil tersebut menunjukkan kemampuan algoritma dalam mempertahankan kualitas jadwal secara konsisten pada setiap penerbangan yang terpilih.
6. Analisis jumlah penerbangan terjadwal: Algoritma berhasil mengakomodasi sebanyak 12 penerbangan ke dalam jadwal akhir tanpa melampaui batas kapasitas operasional harian sebesar 1.440 menit. Hasil ini menunjukkan bahwa algoritma memiliki kemampuan yang baik dalam memaksimalkan pemanfaatan kapasitas waktu yang tersedia. Selain mempertimbangkan minimasi delay, algoritma juga mampu menjaga keseimbangan antara jumlah penerbangan yang dijadwalkan dan keterbatasan waktu operasional sehingga distribusi penerbangan tetap optimal sepanjang hari.
7. Analisis konflik jadwal: Berdasarkan hasil evaluasi, masih terdapat 2 konflik jadwal potensial yang teridentifikasi pada jadwal hasil optimasi. Konflik tersebut muncul akibat adanya penerbangan dengan rute yang sama dan interval waktu operasional yang relatif berdekatan. Namun demikian, penerapan aturan *MIN_GAP* pada tahap *post-processing* berhasil mengurangi jumlah konflik secara

signifikan dan menghasilkan jadwal yang lebih aman dibandingkan hasil optimasi awal. Dengan demikian, proses *post-processing* berperan penting dalam meningkatkan kelayakan operasional jadwal penerbangan yang dihasilkan oleh algoritma..

KESIMPULAN

Berdasarkan hasil penelitian dan pembahasan yang telah dilakukan, dapat disimpulkan bahwa integrasi algoritma *Dynamic Programming* tipe 0/1 *Knapsack* yang serta tahap *post-processing* berbasis aturan celah waktu minimum (*MIN_GAP*) mampu menghasilkan optimasi jadwal penerbangan yang lebih efisien dan terstruktur. Dari pengujian terhadap 15.000 data penerbangan, algoritma berhasil menyusun kombinasi jadwal akhir yang terdiri dari 16 penerbangan optimal dengan pemanfaatan kapasitas waktu operasional harian sebesar 1.440 menit secara efektif dan merata sepanjang hari. Pendekatan fungsi penalti matematika pada parameter *Value* terbukti mampu memprioritaskan penerbangan dengan tingkat keterlambatan rendah dan durasi operasional yang lebih efisien. Hasil evaluasi menunjukkan bahwa algoritma menghasilkan total *delay* sebesar 107.0 menit dengan rata-rata keterlambatan sebesar 6.69 menit per penerbangan. Selain itu, proses *post-processing* berhasil mengidentifikasi dan menangani 6 konflik jadwal potensial melalui penerapan aturan *MIN_GAP* sebesar 20 menit, sehingga jadwal akhir yang dihasilkan menjadi lebih aman dan realistis untuk diterapkan pada kondisi operasional penerbangan sebenarnya.

REFERENSI

[1] N. P. D. Arwini and I. M. Juniastra, “Peran transportasi dalam dunia industri,” *Jurnal Ilmiah Vastuwidya*, vol. 6, no. 1, pp. 70–77, 2023.
[2] J. Ren, S. Qu, L. Wang, C. Liu, L. Ma, and Z. Sun, "A Flight Slot Optimization Model for Beijing-Tianjin-Hebei Airport Cluster Considering Capacity Fluctuation Factor," *Aerospace*, vol. 12, no. 4, p. 336, 2025.
[3] A. A. D. Windusari, *A Statistical Analysis of Flight Delays and Assessment of the Financial Consequences in Selected European Airlines*, Bandung, Indonesia: Bandung Institute of Technology, 2023.
[4] J. Calzada and X. Fageda, “Airport dominance, route network design and flight delays,” *Transportation Research Part E: Logistics and Transportation Review*, vol. 170, p. 103000, 2023.
[5] C. Mannino, A. Nakkerud, and G. Sartor, “Air traffic flow management with layered workload constraints,” *Computers & Operations Research*, vol. 127, p. 105159, 2021.
[6] Q. Cai, S. Alam, and V. Duong, "A Spatial–Temporal Network Perspective for the Propagation Dynamics of Air Traffic Delays," *Engineering*, vol. 7, no. 7, pp. 965–976, 2021.
[7] Y. Zhang, “A survey of dynamic programming algorithms,” *Applied and Computational Engineering*, 2024.
[8] H. Liu, S. Li, F. Sun, W. Fan, W. Ip, and K. Yung, “Adaptive Dynamic Programming with Reinforcement Learning on Optimization of Flight Departure Scheduling,” *Aerospace*, 2024.
[9] K. Alamatsaz, F. Quesnel, and U. Eicker, "Enhancing Electric Shuttle Bus Efficiency: A Case Study on Timetabling and Scheduling Optimization," *Energies*, vol. 17, no. 13, p. 3149, 2024.
[10] M. Wei, S. Yang, W. Wu, and B. Sun, "A Multi-Objective Fuzzy Optimization Model for Multi-Type Aircraft Flight Scheduling Problem," *Transport*, vol. 39, no. 4, pp. 313–322, 2024.
[11] S. Sunardi, S. Humaidi, M. Situmorang, and M. Sinambela, “Development of a mathematical model for managing schedule delays in air traffic operations,” *Eastern-European Journal of Enterprise Technologies*, 2024.
[12] A. Jangir, "Airline Flight Dataset: Schedule, Performance etc," Kaggle, 2023. [Online]. https://www.kaggle.com/datasets/arunjangir245/airline-flight-dataset-schedule-performance-etc [Accessed: 22-Mei-2026].
[13] D. R. Goda, V. R. Vadiyala, S. R. Yerram, and S. R. Mallipeddi, “Dynamic programming approaches for resource allocation in project scheduling: maximizing efficiency under time and budget constraints,” *ABC Journal of Advanced Research*, vol. 12, no. 1, pp. 1–16, 2023.
[14] J. Xu and S. Wu, “Analysis and Application of Dynamic Programming,” *Journal of Physics: Conference Series*, vol. 1865, 2021.
[15] N. Siregar and H. Jumianto, “Penerapan algoritma backtracking dalam penyelesaian masalah,” *Jusinfo: Jurnal Sains dan Informatika*, vol. 1, no. 1, pp. 1–7, 2025.
[16] A. Aida, D. Hermina, and N. Norlaila, “Jenis data penelitian kuantitatif (korelasional, komparatif, dan eksperimen),” *Al-Manba Jurnal Ilmiah Keislaman dan Kemasyarakatan*, 2025.