BAB 2
LANDASAN TEORI

2.1 Sistem Antrian (*Queueing System*)

Sub-bab ini membahas konsep dasar sistem antrian sebagai landasan teoritis untuk memahami mekanisme pelayanan permintaan pada sistem elevator. Pembahasan diawali dari definisi umum sistem antrian, dilanjutkan dengan disiplin antrian First In First Out (FIFO) sebagai pendekatan paling dasar, konsep waiting time sebagai indikator kualitas layanan, hingga keterbatasan FIFO ketika diterapkan pada sistem elevator yang menjadi motivasi pengembangan algoritma yang lebih adaptif.

2.1.1 Definisi dan Konsep Dasar Sistem Antrian

Sistem antrian merupakan suatu kerangka kerja yang menggambarkan proses kedatangan entitas (*request*) ke dalam sistem, menunggu dalam antrean apabila sumber daya pelayanan sedang digunakan, dan kemudian menerima pelayanan secara berurutan. Secara formal, sistem antrian terdiri atas tiga komponen utama, yaitu: proses kedatangan (*arrival process*), mekanisme pelayanan (*service mechanism*), dan disiplin antrian (*queue discipline*) [12].

Dalam konteks komputasi dan rekayasa perangkat lunak, konsep *request* mengacu pada permintaan yang diajukan oleh pengguna atau entitas tertentu kepada sebuah sistem untuk memperoleh pelayanan. *Service* merujuk pada proses pemenuhan permintaan tersebut oleh *server* atau unit pelayanan. Waktu yang berlalu sejak *request* diterima hingga pelayanan dimulai disebut sebagai *waiting time*, yang merupakan salah satu indikator utama kualitas sebuah sistem antrian [2].

2.1.2 FIFO (*First In First Out*)

Disiplin antrian yang paling umum diterapkan adalah *First In First Out* (FIFO), di mana *request* yang tiba lebih awal akan dilayani lebih dahulu tanpa mempertimbangkan faktor lain seperti lokasi, prioritas, atau urgensi. FIFO menjamin

keadilan relatif di antara semua entitas yang menunggu karena tidak ada diskriminasi berdasarkan karakteristik *request* itu sendiri [12].

Dalam implementasinya pada sistem elevator, FIFO menerjemahkan setiap panggilan lift (*hall call*) ke dalam urutan pelayanan yang ketat berdasarkan waktu kedatangan. Artinya, elevator akan bergerak mengikuti urutan permintaan tanpa mempertimbangkan efisiensi pergerakan kabin, sehingga potensi pemborosan gerakan sangat tinggi pada kondisi beban padat.

2.1.3 *Waiting Time* dalam Sistem Antrian

Waiting time dalam sistem antrian didefinisikan sebagai selisih antara waktu kedatangan sebuah *request* ke sistem dan waktu dimulainya proses pelayanan terhadap *request* tersebut. Parameter ini bersifat kritis dalam evaluasi performa sistem karena secara langsung memengaruhi kepuasan pengguna [2].

Pada sistem elevator, *waiting time* merepresentasikan durasi yang dirasakan penghuni dari momen menekan tombol panggil lift (*hall call*) hingga pintu elevator terbuka di lantainya. Semakin tinggi jumlah *request* yang menumpuk tanpa penanganan yang efisien, semakin panjang pula *waiting time* yang dialami pengguna, terutama pada kondisi *rush hour*.

2.1.4 Keterbatasan FIFO pada Sistem Elevator

Meskipun FIFO menjamin keadilan antrian, penerapannya pada sistem elevator memiliki sejumlah keterbatasan mendasar. Pertama, FIFO tidak mempertimbangkan posisi kabin elevator terhadap lokasi *request*, sehingga kabin dapat bergerak ke arah yang berlawanan dengan *request* berikutnya. Kedua, pada kondisi *high traffic* seperti *morning rush hour*, FIFO menghasilkan pola pergerakan elevator yang tidak efisien karena tidak ada mekanisme optimasi berbasis kondisi aktual sistem [3]. Ketiga, FIFO tidak mendukung redistribusi beban antar elevator dalam sistem multi-lift, sehingga potensi *bottleneck* pelayanan lebih besar dibandingkan algoritma yang lebih adaptif.

Keterbatasan-keterbatasan ini menjadi landasan pengembangan algoritma *scheduling* yang lebih canggih, serta motivasi penerapan mekanisme *adaptive dispatching* yang mampu mengalokasikan *request* secara dinamis berdasarkan kondisi

operasional sistem.

2.2 Sistem Elevator dan Multi-Lift

Elevator adalah perangkat transportasi vertikal yang menggerakkan kabin melalui motor listrik, *wire rope*, dan *counterweight*, dikendalikan oleh sistem kontrol elektronik yang menerima *hall call* dari lantai maupun *car call* dari dalam kabin [4]. Pada gedung apartemen, pola penggunaan elevator bersifat residensial dengan lonjakan permintaan pada *morning rush hour* dan malam hari, serta distribusi *request* yang cenderung *one-directional*, yaitu dari lantai hunian ke *lobby* pada pagi hari dan sebaliknya pada malam hari [5]. Karakteristik tersebut menuntut sistem kontrol yang mampu mengantisipasi pola permintaan satu arah dan mendistribusikan beban secara merata antar elevator, sehingga diperlukan mekanisme *scheduling* dan *dispatching* yang lebih efektif.

2.2.1 *Multi-Lift System* dan *Group Elevator Control System* (GECS)

Berbeda dengan sistem elevator tunggal yang rentan mengalami *bottleneck* karena seluruh *request* dilayani oleh satu unit elevator [4], *multi-lift system* menggunakan lebih dari satu elevator yang beroperasi secara simultan dalam satu *shaft group*, sehingga beban pelayanan dapat didistribusikan dan berpotensi mengurangi *waiting time*, bergantung pada mekanisme koordinasinya [3]. Koordinasi ini dikelola oleh *Group Elevator Control System* (GECS), yaitu sistem kendali terpusat yang memantau status seluruh elevator dan mengalokasikan *hall call* sesuai kondisi *traffic* [5]. Tsai et al. [5] mencatat bahwa GECS konvensional umumnya hanya menangani panggilan secara *real-time* tanpa mempertimbangkan statistik historis kebutuhan penumpang, sehingga keterbatasan tersebut menjadi salah satu dasar penerapan mekanisme *adaptive dispatching* pada penelitian ini.

2.2.2 Karakteristik dan Tantangan Operasional Sistem Multi-Lift

Sistem multi-lift memiliki beberapa karakteristik khas yang membedakannya dari sistem tunggal. Pertama, kapasitas pelayanan total lebih besar karena beberapa elevator beroperasi secara paralel. Kedua, fleksibilitas alokasi lebih tinggi karena

satu *request* dapat dilayani oleh lebih dari satu elevator yang tersedia. Ketiga, kompleksitas koordinasi lebih tinggi karena sistem harus memastikan tidak terjadi redundansi pelayanan, yaitu dua elevator yang menuju lantai yang sama untuk *request* yang sama.

Tantangan utama pada sistem multi-lift meliputi:

1. *Load balancing*, yaitu memastikan beban permintaan terdistribusi secara merata antar elevator agar tidak ada satu elevator yang kewalahan sementara elevator lain berada dalam kondisi *idle*.
2. *Conflict avoidance*, yaitu mencegah dua elevator bergerak ke lantai yang sama secara bersamaan untuk melayani *request* yang berbeda.
3. *Dynamic allocation*, yaitu kemampuan sistem untuk mengalihkan *request* dari elevator yang sudah terlalu padat ke elevator lain yang memiliki kapasitas lebih tersedia.
4. *Handling peak traffic*, yaitu kemampuan sistem dalam mengelola lonjakan permintaan pada kondisi *rush hour* tanpa mengalami penurunan performa yang signifikan.

Keempat tantangan tersebut menjadi aspek penting dalam perancangan sistem kendali elevator modern [3, 5].

2.3 *Elevator Scheduling Algorithm*

Sub-bab ini membahas algoritma scheduling yang menjadi fondasi pergerakan elevator dalam melayani request, dengan fokus utama pada algoritma SCAN dan LOOK yang digunakan sebagai algoritma pembanding dalam penelitian ini. Pembahasan mencakup definisi, cara kerja, formulasi matematis, kompleksitas, serta perbandingan karakteristik kedua algoritma.

2.3.1 Definisi dan Fungsi *Elevator Scheduling*

Elevator scheduling adalah proses penentuan urutan pelayanan terhadap sekumpulan *request* yang telah diterima oleh sebuah elevator. *Scheduling* bekerja pada tingkat individual elevator dan mengatur bagaimana elevator tersebut bergerak untuk melayani seluruh *request* dalam antreannya secara efisien [3].

Fungsi utama *scheduling* pada sistem elevator adalah meminimalkan total jarak tempuh kabin, mengurangi *waiting time* pengguna, dan meningkatkan *throughput* pelayanan. *Scheduling* yang baik dapat secara signifikan mengurangi perjalanan kabin yang tidak produktif (*empty travel*) sekaligus memastikan seluruh *request* terlayani dalam waktu yang wajar.

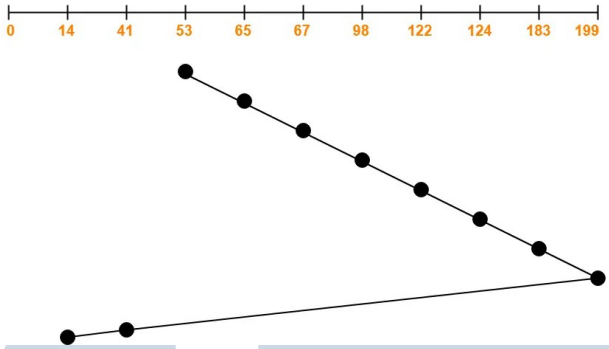
Dalam konteks sistem multi-lift, *scheduling* berperan sebagai lapisan operasional yang menentukan bagaimana masing-masing elevator melayani *request* yang telah dialokasikan kepadanya. Sementara itu, *dispatching* berperan sebagai lapisan strategis yang menentukan elevator mana yang menerima *request* baru. Kedua mekanisme ini bersifat komplementer dan saling mendukung dalam menciptakan sistem multi-lift yang efisien. Pada penelitian ini, algoritma *scheduling* digunakan sebagai fondasi operasional simulasi sekaligus sebagai algoritma pembanding, bukan sebagai fokus utama penelitian.

2.3.2 Algoritma SCAN

Sub-bab ini membahas algoritma SCAN secara menyeluruh, meliputi definisi dan cara kerja, kelebihan dan kekurangan, relevansinya terhadap penelitian, serta formulasi matematis dan analisis kompleksitasnya.

A Definisi dan Cara Kerja

SCAN *algorithm* (disebut juga *elevator algorithm*) merupakan algoritma di mana elevator bergerak ke satu arah tertentu (naik atau turun) sambil melayani seluruh *request* yang dilewatinya, hingga mencapai lantai terjauh atau ujung rentang operasi, kemudian berbalik arah dan melayani *request* di arah sebaliknya [3]. Pola gerak ini menyerupai gerakan kepala baca-tulis pada *hard disk*, sehingga analogi ini lazim digunakan dalam literatur ilmu komputer.



Gambar 2.1. Ilustrasi Pergerakan Algoritma SCAN dalam Melayani Request

Gambar 2.1 mengilustrasikan mekanisme kerja algoritma SCAN pada sistem elevator [6, 13]. Elevator bergerak secara kontinu ke satu arah sambil melayani seluruh *request* yang dilewatinya hingga mencapai lantai paling atas atau paling bawah sebagai batas operasi. Setelah mencapai ujung tersebut, elevator membalik arah dan melayani *request* pada arah sebaliknya. Pola pergerakan ini menjamin seluruh lantai akan dilewati secara sistematis, namun dapat menghasilkan perjalanan kosong ketika tidak terdapat *request* pada lantai-lantai terakhir.

B Kelebihan dan Kekurangan

SCAN mengeliminasi masalah *starvation* karena elevator secara sistematis melewati semua lantai dalam satu siklus. Total pergerakan lebih terprediksi dan lebih efisien dibandingkan FCFS (*First Come First Serve*) maupun SSTF (*Shortest Seek Time First*) pada beban sedang hingga tinggi. Namun, kelemahan utama SCAN adalah kemungkinan elevator bergerak hingga ujung lantai meskipun tidak ada *request* di lantai-lantai terakhir tersebut. Hal ini menghasilkan perjalanan kosong yang tidak perlu, khususnya ketika distribusi *request* tidak merata [6].

C Relevansi terhadap Penelitian

SCAN menjadi referensi penting karena merupakan basis dari LOOK *algorithm* yang digunakan dalam penelitian ini. Memahami kelemahan SCAN secara langsung memotivasi penggunaan LOOK sebagai algoritma *scheduling* yang lebih efisien.

D Formulasi Matematis dan Kompleksitas Algoritma

Secara formal, pergerakan algoritma SCAN dapat dinotasikan sebagai berikut. Misalkan F menyatakan himpunan seluruh lantai gedung dengan $F_{\min}$ dan $F_{\max}$ berturut-turut sebagai lantai terendah dan tertinggi, dan $d \in \{\text{naik}, \text{turun}\}$ menyatakan arah pergerakan aktif elevator. Elevator bergerak secara kontinu pada arah d dan melayani setiap *request* yang berada pada arah tersebut, hingga posisi kabin mencapai batas fisik gedung ($F_{\min}$ atau $F_{\max}$), yang kemudian memicu pembalikan arah terlepas dari ada atau tidaknya *request* tersisa di lantai-lantai menjelang batas tersebut.

Karena setiap siklus penuh SCAN selalu menjangkau seluruh rentang lantai [6], total jarak tempuh satu siklus dapat dirumuskan sebagai:

$$T_{\text{SCAN}} = 2 \times (F_{\max} - F_{\min}) \tag{2.1}$$

yang bersifat konstan dan tidak bergantung pada jumlah maupun distribusi *request* R yang sedang dilayani.

Pseudocode 2.1 menggambarkan logika keputusan algoritma SCAN pada level konsep [6].

```
INPUT  : lantai_sekarang, arah, daftar_request
OUTPUT : log pelayanan

WHILE daftar_request != EMPTY OR
          lantai_sekarang belum mencapai Fmin/Fmax THEN
    IF arah = NAIK THEN
        lantai_sekarang <- lantai_sekarang + 1
        IF ada request pada lantai_sekarang THEN
            layani dan hapus dari daftar_request
        END IF
        IF lantai_sekarang = Fmax THEN
            arah <- TURUN
        END IF
    ELSE
        lantai_sekarang <- lantai_sekarang - 1
```

```
        IF ada request pada lantai_sekarang THEN
            layani dan hapus dari daftar_request
        END IF
        IF lantai_sekarang = Fmin THEN
            arah <- NAIK
        END IF
    END IF
END WHILE
RETURN log pelayanan
```

Kode 2.1: Pseudocode Algoritma SCAN

Kompleksitas algoritma SCAN dapat dianalisis berdasarkan jumlah lantai F dan jumlah *request* R. Pada kasus terbaik (*best-case*), yaitu ketika seluruh *request* berada tepat pada jalur yang akan dilewati kabin pada arah aktifnya, waktu pelayanan per *request* mendekati $O(1)$ karena tidak diperlukan pergerakan tambahan di luar jalur yang sudah dilalui. Namun, pada kasus terburuk (*worst-case*), kabin tetap wajib menempuh seluruh rentang lantai $F_{\max} - F_{\min}$ meskipun hanya tersisa satu *request* di dekat salah satu ujung gedung, sehingga kompleksitas satu siklus penuh adalah $O(F)$, tidak bergantung pada R. Sifat inilah yang menjadi kelemahan struktural SCAN sebagaimana telah disinggung pada bagian B, dan menjadi motivasi pengembangan algoritma LOOK pada subbab 2.3.3.

2.3.3 Algoritma LOOK

Sub-bab ini membahas algoritma LOOK secara menyeluruh sebagai algoritma scheduling utama yang digunakan dalam penelitian ini, meliputi definisi dan cara kerja, kelebihan dan kekurangan, relevansinya terhadap penelitian, serta formulasi matematis dan analisis kompleksitasnya.

A Definisi dan Cara Kerja

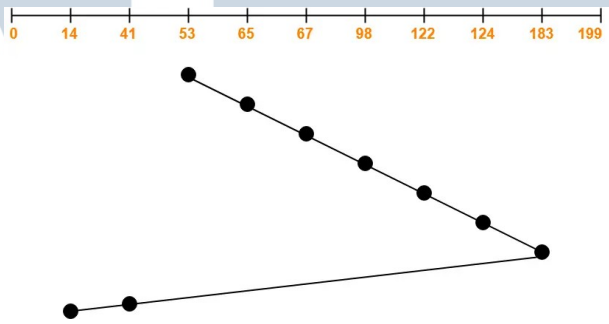
LOOK *algorithm* merupakan varian yang disempurnakan dari SCAN. Perbedaan mendasarnya terletak pada perilaku di ujung perjalanan: elevator tidak bergerak hingga

lantai paling atas atau paling bawah secara absolut, melainkan hanya sejauh *request* terakhir yang ada pada arah aktif saat itu [3].

Secara operasional, algoritma LOOK bekerja melalui tahapan sebagai berikut.

1. Elevator bergerak ke satu arah dan melayani seluruh *request* yang dilewatinya.
2. Ketika tidak ada lagi *request* pada arah aktif, elevator segera berbalik arah tanpa harus mencapai lantai paling ujung.
3. Setelah berbalik arah, elevator melayani seluruh *request* pada arah baru hingga *request* terakhir, kemudian kembali berbalik apabila masih terdapat *request* pada arah sebaliknya.

Prinsip "*look-ahead*" mengacu pada kemampuan sistem untuk memeriksa apakah masih terdapat *request* di depan sebelum memutuskan untuk berbalik arah atau tetap melanjutkan pergerakan [3].



Gambar 2.2. Ilustrasi Pergerakan Algoritma LOOK dalam Melayani Request

Gambar 2.2 menunjukkan mekanisme kerja algoritma LOOK [6, 13]. Berbeda dengan SCAN, elevator hanya bergerak hingga *request* terakhir pada arah aktif, kemudian langsung membalik arah tanpa harus mencapai lantai paling atas atau paling bawah. Dengan menghilangkan perjalanan menuju ujung gedung yang tidak memiliki *request*, algoritma LOOK mampu mengurangi total jarak tempuh elevator sehingga menghasilkan pergerakan yang lebih efisien.

B Kelebihan dan Kekurangan

LOOK mengeliminasi perjalanan kosong yang tidak diperlukan dengan menghindari pergerakan ke lantai ujung tanpa *request*. Hasilnya adalah total pergerakan

elevator yang lebih efisien dan waktu respons yang lebih rendah dibandingkan SCAN, khususnya pada distribusi *request* yang tidak merata [6]. Seperti SCAN, LOOK masih berpotensi menghasilkan ketidakmerataan *waiting time* antara *request* yang berada di ujung arah aktif dengan *request* yang baru saja dilewati oleh elevator. Selain itu, LOOK dalam bentuk murni tidak memiliki mekanisme adaptif untuk merespons kondisi beban elevator atau status elevator lain dalam sistem multi-lift.

C Relevansi terhadap Penelitian

Dalam penelitian ini, SCAN dan LOOK digunakan sebagai algoritma scheduling yang mendasari sistem simulasi karena keduanya merepresentasikan mekanisme pergerakan elevator pada bangunan bertingkat. SCAN dan LOOK digunakan sebagai algoritma pembanding untuk mengevaluasi performa sistem multi-lift dalam melayani request penumpang. *Adaptive dispatching* kemudian diterapkan pada kedua algoritma tersebut untuk mengevaluasi pengaruhnya terhadap efisiensi sistem, khususnya dalam mengurangi waiting time dan meningkatkan distribusi beban antar elevator.

D Formulasi Matematis dan Kompleksitas Algoritma

Berbeda dengan SCAN, algoritma LOOK membatasi pergerakan kabin hanya sejauh *request* terjauh yang tersisa pada arah aktif. Misalkan R_{up} dan R_{down} berturut-turut menyatakan himpunan lantai yang memiliki *request* pada arah naik dan arah turun, dengan $\max(\cdot)$ dan $\min(\cdot)$ menyatakan lantai tertinggi/terendah dalam himpunan tersebut. Elevator membalik arah segera setelah tidak ada lagi elemen pada himpunan *request* arah aktifnya, tanpa harus mencapai $F_{\min}$ atau $F_{\max}$.

Total jarak tempuh satu siklus LOOK dapat dirumuskan sebagai [6]:

$$T_{\text{LOOK}} = 2 \times \big(\max(R_{\text{up}} \cup R_{\text{down}}) - \min(R_{\text{up}} \cup R_{\text{down}})\big) \tag{2.2}$$

yang menunjukkan bahwa $T_{\text{LOOK}} \leq T_{\text{SCAN}}$ (lihat Persamaan 2.1) untuk distribusi *request* apa pun, dengan kesetaraan hanya terjadi ketika *request* tersebar hingga menyentuh kedua ujung gedung ($F_{\min}$ dan $F_{\max}$).

Pseudocode 2.2 menggambarkan logika keputusan algoritma LOOK [6].

```
INPUT  : lantai_sekarang, arah, daftar_request
OUTPUT : log pelayanan

WHILE daftar_request != EMPTY THEN
    IF arah = NAIK THEN
        lantai_sekarang <- lantai_sekarang + 1
        IF ada request pada lantai_sekarang THEN
            layani dan hapus dari daftar_request
        END IF
        IF tidak ada request pada arah NAIK di atas
              lantai_sekarang THEN        // beda kunci vs
    SCAN
            arah <- TURUN
        END IF
    ELSE
        lantai_sekarang <- lantai_sekarang - 1
        IF ada request pada lantai_sekarang THEN
            layani dan hapus dari daftar_request
        END IF
        IF tidak ada request pada arah TURUN di bawah
              lantai_sekarang THEN
            arah <- NAIK
        END IF
    END IF
END WHILE
RETURN log pelayanan
```

Kode 2.2: Pseudocode Algoritma LOOK

Dari sisi kompleksitas, *best-case* LOOK identik dengan SCAN, yaitu $O(1)$ per *request* ketika *request* berada tepat di jalur yang sedang dilalui. Perbedaan struktural muncul pada *worst-case*: karena batas pergerakan LOOK ditentukan oleh sebaran *request* aktual dan bukan oleh batas fisik gedung, *worst-case* LOOK tetap $O(F)$ hanya pada kondisi ekstrem ketika *request* tersebar hingga ke kedua ujung gedung; pada kondisi umum ketika *request* tidak merata di seluruh lantai sebagaimana karakteristik

down-peak traffic pada *morning rush hour* (lihat subbab 2.6.2) jarak tempuh LOOK secara praktis dibatasi oleh sebaran *request* itu sendiri sehingga mendekati $O(R)$ dengan $R \leq F$. Sifat teoritis inilah yang menjelaskan mengapa LOOK secara konsisten menghasilkan *Total Elevator Movement* yang lebih rendah dibandingkan SCAN pada hasil simulasi (dibahas lebih lanjut pada Bab IV), dan sejalan dengan perbandingan kualitatif yang telah dirangkum pada Tabel 2.1 dan Tabel 2.2.

2.3.4 Perbandingan Algoritma *Scheduling*

Tabel 2.1. Perbandingan Karakteristik Algoritma *Elevator Scheduling*

Algoritma	Prinsip Kerja	Kelebihan	Kekurangan	Skenario Terbaik	Starvation
SCAN	Satu arah hingga ujung, lalu berbalik	Tidak ada *starvation*, sistematis	Perjalanan ke ujung meskipun tanpa *request*	Beban tinggi, distribusi merata	Tidak
LOOK	Satu arah hingga *request* terakhir, lalu berbalik	Tidak ada *starvation*, efisien, tanpa perjalanan kosong	Tidak ada mekanisme adaptif bawaan	Beban tinggi, distribusi tidak merata	Tidak

Tabel 2.2. Perbandingan SCAN dan LOOK *Algorithm*

Aspek	**SCAN**	**LOOK**
Batas Pergerakan	Hingga lantai paling atas/bawah secara absolut	Hingga *request* terakhir pada arah aktif
Perjalanan Kosong	Mungkin terjadi jika tidak ada *request* di ujung	Dieliminasi, elevator berbalik di *request* terakhir
Pembalikan Arah	Dilakukan di ujung rentang operasi	Dilakukan setelah semua *request* arah aktif terlayani
Efisiensi Total Pergerakan	Lebih rendah karena perjalanan ekstra ke ujung	Lebih tinggi karena tidak ada perjalanan ekstra
Starvation	Tidak ada	Tidak ada
Kesesuaian dengan Elevator Modern	Baik	Lebih baik, sesuai pola operasi elevator nyata

2.4 *Elevator Dispatching* dan *Adaptive Dispatching*

Sub-bab ini membahas konsep dispatching sebagai mekanisme koordinasi antar elevator dalam sistem multi-lift, serta adaptive dispatching sebagai kontribusi utama yang diusulkan dalam penelitian ini. Pembahasan diawali dari perbedaan mendasar antara scheduling dan dispatching, dilanjutkan dengan faktor-faktor yang memengaruhi keputusan dispatching, hingga mekanisme adaptive dispatching secara rinci.

2.4.1 Definisi dan Perbedaan *Scheduling* dengan *Dispatching*

Dispatching pada sistem elevator adalah proses penentuan elevator mana yang akan dikirim untuk melayani sebuah *hall call* baru. Berbeda dengan *scheduling* yang mengatur urutan pelayanan *request* dalam antrean satu elevator, *dispatching* beroperasi pada level koordinasi antar elevator dalam sistem multi-lift [5].

Perbedaan mendasar antara *scheduling* dan *dispatching* dapat dirumuskan sebagai berikut: *scheduling* menentukan bagaimana dan dalam urutan apa satu elevator melayani *request-request* yang telah dialokasikan kepadanya, sedangkan *dispatching* menentukan elevator mana yang bertanggung jawab melayani sebuah *request* baru.

Dengan demikian, *dispatching* berperan sebelum *scheduling*: keputusan *dispatching* menentukan *request* mana yang masuk ke antrean masing-masing elevator, kemudian algoritma *scheduling* mengatur urutan pelayanan dalam antrean tersebut.

2.4.2 Konsep dan Mekanisme *Dispatching*

Dispatching adalah mekanisme penentuan elevator yang ditugaskan melayani sebuah *hall call*, sebelum urutan pelayanannya diatur oleh *Scheduling Algorithm*. Pada sistem *multi-lift*, keputusan ini krusial agar beban kerja tidak menumpuk pada satu unit elevator saja. Sub-bab ini membahas tiga konsep terkait: *elevator assignment* (pemetaan *request* ke elevator), *dynamic dispatching* (perubahan keputusan alokasi berdasarkan kondisi terkini), dan *real-time dispatching* (pengambilan keputusan secara kontinu dan responsif).

A *Elevator Assignment*

Elevator assignment adalah inti dari proses *dispatching*, yaitu proses pemetaan setiap *hall call* ke salah satu elevator yang tersedia dalam sistem. Kualitas *assignment* menentukan seberapa efisien sistem multi-lift dapat melayani seluruh pengguna. *Assignment* yang buruk dapat menyebabkan satu elevator kewalahan sementara elevator lain *idle*, yang berujung pada peningkatan *waiting time* secara keseluruhan [5].

B *Dynamic Dispatching*

Dynamic dispatching merujuk pada kemampuan sistem untuk mengubah keputusan alokasi secara *real-time* berdasarkan kondisi terkini dari seluruh elevator. Berbeda dengan *dispatching* statis yang mengikuti aturan tetap (misalnya selalu menugaskan elevator terdekat), *dynamic dispatching* mempertimbangkan berbagai variabel kondisional seperti panjang antrean, estimasi waktu kedatangan, dan tingkat kepadatan setiap elevator [6].

C *Real-Time Dispatching*

Real-time dispatching mengacu pada proses evaluasi dan pengambilan keputusan *dispatching* yang dilakukan secara kontinu setiap kali terdapat *request* baru. Sistem harus mampu mengolah informasi status seluruh elevator dan menghasilkan keputusan *assignment* dalam waktu yang sangat singkat agar tidak menambah latensi pelayanan [5].

2.4.3 Faktor yang Memengaruhi Keputusan *Dispatching*

Keputusan *dispatching* pada sistem multi-lift dipengaruhi oleh berbagai faktor yang bersifat dinamis dan saling berinteraksi. Faktor-faktor utama yang perlu dipertimbangkan antara lain:

1. Jarak elevator dari lantai *request*: elevator yang posisinya lebih dekat secara fisik memiliki potensi waktu respons yang lebih singkat.
2. Panjang antrean aktif: elevator dengan jumlah *request* dalam antrean yang lebih banyak akan memiliki waktu pelayanan yang lebih panjang sebelum dapat merespons *request* baru.
3. Arah pergerakan elevator: elevator yang bergerak menuju lantai *request* lebih efisien dibandingkan elevator yang sedang bergerak menjauhi lantai *request*.
4. Status elevator (*active*/*idle*): elevator yang sedang *idle* dapat merespons *request* baru dengan lebih cepat dibandingkan elevator yang sedang dalam perjalanan.
5. Estimasi *waiting time*: perkiraan waktu yang dibutuhkan sebelum elevator dapat tiba di lantai pengguna.

2.4.4 *Adaptive Dispatching*

Sub-bab ini membahas adaptive dispatching secara rinci sebagai kontribusi utama penelitian ini, meliputi definisi, konsep adaptive control, mekanisme pengambilan keputusan dinamis, serta berbagai basis pertimbangan yang digunakan dalam proses dispatching.

A Definisi

Adaptive dispatching adalah mekanisme *dispatching* yang secara dinamis menyesuaikan keputusan alokasi elevator berdasarkan kondisi operasional sistem yang terus berubah. Berbeda dengan *dispatching* konvensional yang umumnya hanya mempertimbangkan satu faktor dominan seperti jarak terdekat, *adaptive dispatching* melakukan evaluasi multifaktor secara *real-time* untuk menentukan elevator yang paling optimal dalam melayani setiap *request* baru [6].

B *Adaptive Control* pada Elevator

Konsep *adaptive control* pada sistem elevator mengacu pada kemampuan sistem untuk memodifikasi perilaku operasionalnya sebagai respons terhadap perubahan kondisi beban, pola *traffic*, dan status elevator. Menurut Wu dan Yang [6], modifikasi dinamis terhadap mekanisme pengambilan keputusan merupakan arah pengembangan yang menjanjikan untuk menangani pola lalu lintas vertikal yang kompleks dan berfluktuasi.

C *Dynamic Decision Making*

Dalam *adaptive dispatching*, setiap keputusan *dispatching* dibuat berdasarkan *snapshot* kondisi sistem pada saat *request* diterima. Sistem mengevaluasi seluruh elevator secara simultan, membandingkan estimasi performa masing-masing elevator jika *request* ditugaskan kepadanya, kemudian memilih elevator dengan estimasi terbaik. Proses ini memastikan bahwa keputusan *dispatching* selalu relevan dengan kondisi aktual sistem [5].

D *Dispatching* Berbasis *Waiting Time*

Pendekatan *dispatching* berbasis *waiting time* berfokus pada minimisasi estimasi *waiting time* yang akan dialami pengguna jika sebuah *request* dialokasikan ke elevator tertentu. Sistem menghitung estimasi waktu kedatangan elevator ke lantai *request* berdasarkan posisi saat ini, arah pergerakan, dan jumlah *stop* yang masih harus dilayani.

Elevator dengan estimasi *waiting time* terendahlah yang dipilih untuk melayani *request* baru tersebut.

E *Dispatching* Berbasis Beban Elevator

Sistem *adaptive dispatching* juga mempertimbangkan tingkat kepadatan atau beban masing-masing elevator. Elevator yang memiliki terlalu banyak *request* dalam antrean akan cenderung menghasilkan *waiting time* yang tinggi meskipun secara posisi berada lebih dekat dengan lantai *request*. Oleh karena itu, beban elevator menjadi faktor penting dalam keputusan *dispatching*.

F *Idle Elevator Utilization*

Salah satu keunggulan *adaptive dispatching* adalah kemampuannya memanfaatkan elevator yang sedang *idle* secara optimal. Elevator *idle* dapat diposisikan secara strategis atau langsung dialokasikan untuk melayani *request* baru guna mengurangi beban elevator yang aktif dan mempercepat waktu respons sistem secara keseluruhan [5].

G *Request Redistribution* dan *Load Balancing*

Adaptive dispatching dapat melakukan redistribusi *request*, yaitu mengalihkan permintaan yang semula akan ditugaskan ke elevator tertentu kepada elevator lain yang memiliki kapasitas pelayanan lebih baik. Mekanisme ini secara efektif mewujudkan *load balancing*, yaitu distribusi beban pelayanan yang merata di antara seluruh elevator dalam sistem multi-lift sehingga tidak ada satu elevator yang menjadi *bottleneck* [3].

2.4.5 Ilustrasi *Adaptive Dispatching* pada Sistem Tiga Elevator

Untuk mengilustrasikan keunggulan *adaptive dispatching*, diasumsikan terdapat tiga elevator (A, B, dan C) serta sebuah *request* baru dari lantai 8 menuju lantai 1 pada pukul 07.15 saat *morning rush hour*. Elevator A berada di lantai 9 dan bergerak turun dengan 6 *request* aktif, Elevator B berada di lantai 3 dan bergerak naik dengan 2 *request* aktif, sedangkan Elevator C berada di lantai 1 dalam kondisi *idle*. *Dispatching*

konvensional berbasis jarak akan memilih Elevator A karena paling dekat dengan lantai asal, meskipun antreannya menyebabkan estimasi *waiting time* menjadi tinggi. Sebaliknya, *adaptive dispatching* mengevaluasi seluruh kondisi elevator dan memilih Elevator C karena mampu memberikan estimasi *waiting time* yang lebih rendah. Ilustrasi ini menunjukkan bahwa elevator terdekat belum tentu memberikan *waiting time* terbaik sehingga keputusan berbasis multifaktor lebih efektif dalam meminimalkan *waiting time*.

2.4.6 Kontribusi *Adaptive Dispatching* dalam Penelitian

Dalam penelitian ini, *adaptive dispatching* merupakan kontribusi utama yang diusulkan. Sistem secara *real-time* mengevaluasi kondisi seluruh elevator berdasarkan posisi, arah, panjang antrean, dan estimasi *waiting time* untuk menentukan elevator yang paling sesuai melayani setiap *request*. Apabila estimasi *waiting time* atau jumlah *request* aktif melebihi *threshold* yang ditetapkan, *request* dialihkan ke elevator lain dengan estimasi pelayanan yang lebih baik. Mekanisme ini diterapkan pada algoritma SCAN dan LOOK untuk mengevaluasi pengaruh *adaptive dispatching* terhadap efisiensi alokasi *request*, *waiting time*, dan distribusi beban antar elevator.

2.5 *Waiting Time* dan Efisiensi Sistem Elevator

Sub-bab ini membahas metrik-metrik yang digunakan untuk mengukur kualitas layanan dan efisiensi operasional sistem elevator, meliputi waiting time, response time, travel time, total elevator movement, dan throughput. Metrik-metrik ini menjadi dasar bagi parameter evaluasi yang digunakan secara formal pada subbab 2.8.

2.5.1 *Waiting Time*

Waiting time didefinisikan sebagai durasi waktu yang berlalu sejak seorang pengguna menekan tombol *hall call* hingga pintu elevator terbuka di lantainya. *Waiting time* merupakan parameter kualitas layanan yang paling langsung dirasakan oleh pengguna dan menjadi indikator utama kepuasan dalam sistem transportasi vertikal [2]. Secara matematis, *waiting time* untuk pengguna ke-*i* dapat diekspresikan sebagai:

$$WT_i = t_{\text{serve},i} - t_{\text{arrive},i} \tag{2.3}$$

di mana $t_{\text{serve},i}$ adalah waktu saat elevator tiba dan membuka pintu di lantai pengguna, dan $t_{\text{arrive},i}$ adalah waktu saat pengguna menekan tombol *hall call*. *Average Waiting Time* (AWT) dihitung sebagai:

$$\text{AWT} = \frac{1}{N}\sum_{i=1}^{N} WT_i \tag{2.4}$$

di mana N adalah total jumlah *request* dalam periode evaluasi. AWT digunakan sebagai metrik utama dalam penelitian ini untuk membandingkan performa *adaptive dispatching* dengan algoritma pembanding.

2.5.2 *Response Time* dan *Travel Time*

Response time adalah total waktu yang dibutuhkan sejak pengguna menekan tombol *hall call* hingga pengguna tiba di lantai tujuan. *Response time* merupakan gabungan dari *waiting time* dan *travel time* [2]. *Travel time* adalah durasi perjalanan elevator dari lantai asal pengguna hingga lantai tujuan, dan secara langsung dipengaruhi oleh jumlah *stop* yang dilakukan elevator selama perjalanan tersebut.

$$RT_i = WT_i + TT_i \tag{2.5}$$

di mana RT_i adalah *response time*, WT_i adalah *waiting time*, dan TT_i adalah *travel time* untuk pengguna ke-i. Minimisasi *response time* memerlukan optimasi pada kedua komponen, yaitu pengurangan *waiting time* melalui *dispatching* yang lebih baik dan pengurangan *travel time* melalui *scheduling* yang lebih efisien.

2.5.3 Total *Movement* Elevator dan *Throughput*

Total *elevator movement* merupakan jumlah kumulatif perpindahan lantai seluruh elevator selama periode evaluasi, yang mencerminkan efisiensi operasional

sistem; semakin kecil total *movement* untuk jumlah *request* yang sama, semakin efisien sistem beroperasi. Sementara itu, *throughput* elevator adalah jumlah *request* yang berhasil dilayani per satuan waktu dan menjadi indikator kemampuan sistem dalam menangani beban permintaan, terutama pada kondisi *morning rush hour*.

2.5.4 Hubungan *Waiting Time* dan Kenyamanan Pengguna

Kenyamanan pengguna sangat dipengaruhi oleh *waiting time*: ketidaknyamanan mulai dirasakan ketika *waiting time* melebihi 30–60 detik dan meningkat signifikan seiring bertambahnya durasi tunggu [14]. Urgensi ini semakin tinggi pada kondisi *morning rush hour* di apartemen karena berkaitan langsung dengan jadwal keberangkatan penghuni, sehingga optimasi *waiting time* pada kondisi tersebut memberikan dampak praktis yang signifikan terhadap kualitas hidup penghuni.

2.6 *Morning Rush Hour Traffic*

Morning rush hour merujuk pada periode lonjakan signifikan permintaan transportasi vertikal secara bersamaan, yang mencerminkan ritme aktivitas penghuni berdasarkan jadwal kerja dan aktivitas sosial [5]. Pada gedung apartemen, kondisi ini umumnya terjadi pada pukul 06.00–12.00 dengan puncak permintaan pada jam-jam awal, serta didominasi pergerakan satu arah dari lantai hunian menuju *lobby*. Karakteristik tersebut menciptakan beban *unidirectional* yang masif dan menjadi tantangan utama bagi sistem *multi-lift*.

2.6.1 Pola Pergerakan dan Distribusi *Request*

Pada *morning rush hour* di apartemen, distribusi *request* elevator memiliki pola yang khas. Permintaan berasal dari berbagai lantai hunian secara hampir bersamaan, dengan tujuan yang terkonsentrasi pada satu atau dua lantai tertentu (biasanya lantai 1/*lobby* dan lantai parkir). Kondisi ini menciptakan konvergensi *request* yang masif menuju lantai bawah, sementara permintaan naik (dari *lobby* ke lantai hunian) sangat minimal [8].

Karakteristik distribusi *request* pada kondisi *morning rush hour* meliputi:

1. *Arrival rate* yang tinggi dalam satuan waktu yang singkat.
2. Dominasi *request* dengan arah turun (*down-peak traffic*).
3. Distribusi asal *request* yang tersebar di berbagai lantai hunian.
4. Tujuan *request* yang sangat terkonsentrasi pada lantai *lobby*.

Kondisi tersebut berbeda secara fundamental dengan pola *traffic* normal (*interfloor traffic*) yang memiliki distribusi asal dan tujuan *request* yang lebih merata.

2.6.2 Pengaruh *Morning Rush Hour* terhadap *Waiting Time* dan Performa Elevator

Lonjakan permintaan pada *morning rush hour* meningkatkan *waiting time* secara eksponensial ketika *arrival rate* melampaui *service rate* elevator [8]. Khonjun et al. [8] membuktikan bahwa kondisi *rush hour* dengan beban puncak dapat menghasilkan *waiting time* yang tidak dapat diterima tanpa mekanisme optimasi yang tepat. Oleh karena itu, skenario *morning rush hour* dijadikan sebagai konteks evaluasi utama penelitian ini, dengan data observasi dikumpulkan pada pukul 06.00–12.00 untuk merepresentasikan kondisi tersebut secara realistis.

2.7 Optimasi Sistem Elevator

Optimasi sistem elevator berfokus pada pengurangan *waiting time* dan peningkatan *throughput* melalui perbaikan mekanisme *dispatching* [6], dengan tujuan meningkatkan pemanfaatan elevator *idle*, memeratakan distribusi beban, dan mengurangi pergerakan tidak produktif. Penelitian ini menerapkan empat strategi *adaptive dispatching* untuk mencapainya: pengurangan *bottleneck*, pemanfaatan elevator *idle*, redistribusi *request* berbasis *threshold*, dan *load balancing* dinamis, dijelaskan pada Subbab 3.7.5–3.7.7.

2.8 Parameter Evaluasi Penelitian

Evaluasi performa sistem *adaptive dispatching* multi-lift dalam penelitian ini menggunakan **sepuluh** parameter utama yang saling melengkapi. Setiap parameter mengukur aspek berbeda dari efisiensi dan kualitas layanan sistem, termasuk tiga

parameter tambahan berupa nilai persentil *waiting time* untuk memahami konsistensi performa di luar nilai rata-rata.

Tabel 2.3. Parameter Evaluasi Performa Sistem Multi-Lift

Parameter	**Definisi**	**Fungsi dalam Penelitian**
***Average Waiting Time* (AWT)**	Rata-rata durasi dari *hall call* hingga elevator tiba di lantai pengguna	Metrik utama keberhasilan *adaptive dispatching* dalam mengurangi waktu tunggu
***Average Travel Time* (ATT)**	Rata-rata durasi perjalanan dari lantai asal ke lantai tujuan pengguna	Mengukur efisiensi pergerakan elevator dalam melayani setiap pengguna
***Response Time* (RT)**	Total waktu dari *hall call* hingga pengguna tiba di lantai tujuan (AWT + ATT)	Mengukur total pengalaman layanan yang dirasakan pengguna
***Total Elevator Movement* (TEM)**	Jumlah kumulatif perpindahan lantai seluruh elevator dalam periode evaluasi	Mengukur efisiensi operasional dan mendeteksi perjalanan elevator yang tidak produktif
***Throughput* Elevator**	Jumlah *request* yang berhasil dilayani per satuan waktu	Mengukur kapasitas dan produktivitas sistem dalam menangani permintaan
Load Distribution	Distribusi jumlah *request* yang dilayani oleh masing-masing elevator	Mengevaluasi efektivitas *load balancing* pada sistem multi-lift
Elevator Utilization	Persentase waktu elevator berada dalam kondisi aktif melayani *request*	Mengidentifikasi elevator yang *underutilized* atau *overloaded*
P50 *Waiting Time*	Persentil ke-50 (median) dari distribusi *waiting time* seluruh pengguna	Menggambarkan waktu tunggu tipikal yang dialami sebagian besar pengguna
P90 *Waiting Time*	Persentil ke-90 dari distribusi *waiting time* seluruh pengguna	Mengukur kualitas layanan pada kelompok pengguna dengan waktu tunggu tinggi
P95 *Waiting Time*	Persentil ke-95 dari distribusi *waiting time* seluruh pengguna	Mengukur performa sistem pada kondisi waktu tunggu yang mendekati ekstrem

P50, P90, dan P95 masing-masing merupakan persentil ke-50, ke-90, dan ke-95 dari distribusi *waiting time* seluruh pengguna selama periode evaluasi. Berbeda dengan AWT yang hanya merepresentasikan rata-rata, ketiga nilai persentil ini digunakan untuk memahami sebaran (distribusi) *waiting time* secara lebih menyeluruh: P50 menggambarkan pengalaman waktu tunggu tipikal yang dirasakan mayoritas pengguna, sementara P90 dan P95 mengungkap seberapa buruk pengalaman kelompok pengguna dengan waktu tunggu terpanjang, yang berpotensi tidak terlihat apabila evaluasi hanya mengandalkan nilai rata-rata.

2.9 Penelitian Terdahulu

Kajian terhadap penelitian terdahulu dilakukan untuk memetakan perkembangan ilmu pengetahuan di bidang *elevator scheduling*, *dispatching*, *multi-lift optimization*, dan *adaptive control*. Penelitian-penelitian berikut dipilih karena relevansinya langsung terhadap topik dan metodologi penelitian ini.

Tabel 2.4. Rangkuman Penelitian Terdahulu

Penulis	**Tahun**	**Metode**	**Hasil**	**Kelebihan**	**Kekurangan**	**Relevansi**
Li & Chu [4]	2025	Intelligent reservation & scheduling berbasis SmartRide	Reduksi *waiting time* 30% dibandingkan FCFS konvensional	Integrasi reservasi penumpang dengan *scheduling* dinamis	Tidak membahas *adaptive dispatching* pada sistem multi-lift	Basis perbandingan reduksi *waiting time* terhadap FCFS
Tsai et al. [5]	2025	Standby scheduling strategy berbasis AI pada smart building EGCS	Reduksi rata-rata *waiting time* >24% pada gedung menengah dan tinggi	Strategi proaktif memposisikan elevator *idle* secara strategis	Bergantung pada *smart building* infrastructure; tidak dibahas pada apartemen konvensional	Inspirasi konsep *idle elevator utilization* dalam *adaptive dispatching* penelitian ini
Wu & Yang [6]	2024	Directional optimization berbasis matriks relasi untuk pola *traffic* kompleks	Lebih efisien menangani *multi-peak traffic* dibandingkan algoritma berbasis pengenalan pola eksplisit	Mampu mengakomodasi variasi pola *traffic* yang kompleks	Kompleksitas komputasi tinggi; tidak fokus pada *waiting time* sebagai metrik utama	Memperkuat pentingnya modifikasi dinamis mekanisme *dispatching* untuk *traffic* kompleks
Wan, Lee & Shin [10]	2024	Deep reinforcement learning untuk traffic pattern-aware elevator dispatching (SMDP)	Kebijakan *dispatching* yang lebih adaptif terhadap berbagai pola *traffic*	Formulasi SMDP yang sistematis dengan *reward function* yang terancang baik	Memerlukan data *training* yang besar; tidak cocok untuk simulasi tanpa infrastruktur ML	Memperkuat pendekatan *adaptive decision making*; tanpa ML menjadi celah yang diisi penelitian ini

(Berlanjut ke halaman berikutnya)

(Lanjutan) Tabel 2.4 – Rangkuman Penelitian Terdahulu

Penulis	**Tahun**	**Metode**	**Hasil**	**Kelebihan**	**Kekurangan**	**Relevansi**
Gharbi [9]	2024	Multi-agent system dengan reinforcement learning untuk elevator group control	Peningkatan efisiensi koordinasi antar elevator dalam grup	Koordinasi multi-agent yang fleksibel dan adaptif	Implementasi kompleks; belum terbukti pada skenario *morning rush hour* apartemen	Referensi konsep koordinasi antar elevator dalam sistem grup
Khonjun et al. [8]	2023	Multi-objective variable neighborhood strategy adaptive search untuk minimasi energi dan makespan	Reduksi *waiting time* 42,44% dan konsumsi energi 29,61% pada *rush hour*	Menangani *multi-objective optimization* secara komprehensif pada kondisi *rush hour*	Fokus pada energi; tidak membahas *adaptive dispatching* berbasis *threshold*	Bukti urgensi optimasi elevator pada kondisi *rush hour*; target penurunan *waiting time*
Maleki, Bhatta & Mashayekhy [11]	2023	*Game-theoretic approach*: pembentukan koalisi *request* antar penumpang	Minimasi gerakan redundan kabin; peningkatan efisiensi energi	Pendekatan teoritis yang solid dengan bukti formal efisiensi	Tidak membahas *waiting time* sebagai metrik utama; tidak fokus pada *rush hour*	Referensi strategi minimasi gerakan redundan dalam sistem multi-lift
Thebuwena, Samarakoon & Ratnayake [2]	2024	Review komprehensif efisiensi energi pada sistem elevator modern	Pemetaan komprehensif faktor efisiensi energi dan operasional elevator	Cakupan luas mencakup berbagai aspek sistem elevator modern	Tidak fokus pada algoritma *dispatching* atau *scheduling* secara spesifik	Dasar pemahaman karakteristik sistem elevator modern dalam penelitian ini

2.9.1 Analisis Kesenjangan Penelitian (*Research Gap*)

Berdasarkan kajian penelitian terdahulu, terdapat tiga kesenjangan yang mendasari penelitian ini: (1) sebagian besar penelitian dengan performa tinggi mengandalkan *machine learning* atau *deep reinforcement learning* [9, 10], sehingga optimasi berbasis *adaptive dispatching* non-*ML* masih terbatas; (2) sebagian besar penelitian berfokus pada gedung perkantoran atau gedung campuran, sedangkan penelitian pada apartemen residensial dengan karakteristik *down-peak traffic* saat *morning rush hour* masih terbatas [5, 8]; dan (3) mekanisme redistribusi *request* berbasis *threshold* yang mempertimbangkan kondisi aktual setiap elevator belum banyak dibahas.

Penelitian ini mengisi ketiga kesenjangan tersebut melalui pengembangan *adaptive dispatching* berbasis algoritmik tanpa *machine learning*, penerapannya pada sistem *multi-lift* apartemen saat *morning rush hour*, serta implementasi mekanisme redistribusi *request* berbasis *threshold*.

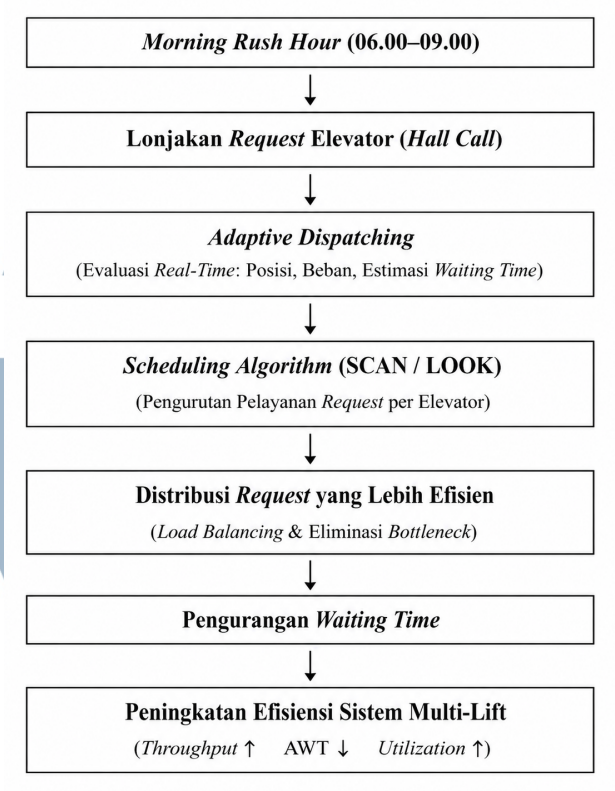
2.10 Kerangka Berpikir Penelitian

Kerangka berpikir penelitian ini berangkat dari permasalahan tingginya *waiting time* pada sistem multi-lift apartemen selama *morning rush hour*. Pada kondisi tersebut, lonjakan *request* yang bersifat *unidirectional* menyebabkan distribusi permintaan menjadi tidak merata, sehingga meningkatkan potensi *bottleneck* pada elevator tertentu dan menurunkan kualitas pelayanan.

Sebagai solusi, penelitian ini menerapkan mekanisme *adaptive dispatching* yang mengevaluasi kondisi seluruh elevator secara *real-time* untuk mengalokasikan setiap *request* kepada elevator yang paling sesuai berdasarkan kondisi sistem. Mekanisme ini diintegrasikan dengan algoritma *scheduling* SCAN dan LOOK sebagai dasar pengelolaan pergerakan masing-masing elevator.

Performa sistem kemudian dievaluasi menggunakan parameter yang telah didefinisikan pada Subbab 2.8, dengan *average waiting time* sebagai metrik utama. Evaluasi dilakukan pada empat skenario, yaitu SCAN, LOOK, SCAN + *Adaptive Dispatching*, dan LOOK + *Adaptive Dispatching*, sehingga kontribusi mekanisme *adaptive dispatching* terhadap peningkatan performa sistem dapat dianalisis secara

objektif.



Gambar 2.3. Kerangka Berpikir

Kerangka berpikir di atas mengilustrasikan bahwa setiap komponen dalam penelitian ini memiliki hubungan kausal yang jelas. *Morning rush hour* sebagai konteks permasalahan, *adaptive dispatching* sebagai solusi utama, algoritma SCAN dan LOOK sebagai fondasi *scheduling* untuk mengatur pergerakan elevator, serta berbagai parameter evaluasi sebagai alat ukur keberhasilan, semuanya terhubung dalam satu rantai logis yang koheren dan dapat diuji melalui simulasi komputasional.