Remik: Riset dan E-Jurnal Manajemen Informatika Komputer
Volume 9, Number 3, Agustus 2025
http://doi.org/10.33395/remik.v9i3.14924
e-ISSN : 2541-1330
p-ISSN : 2541-1332

Analisis Perbandingan Algoritma Penjadwalan CPU pada Sistem Operasi Linux

[1]Ahmad Izzuddin Azzam, [2] Herbert Siregar
[1,2]Universitas Pendidikan Indonesia
Bandung, Indonesia

[1]izzuddinazzam@upi.edu, [2]herbert@upi.edu

*Penulis Korespondensi
Diajukan : 20/06/2025
Diterima : 14/07/2025
Dipublikasi : 01/08/2025

ABSTRAK
Penjadwalan *Central Processing Unit* (CPU) merupakan aspek fundamental dalam sistem operasi yang berperan penting dalam efisiensi dan kinerja sistem secara keseluruhan. Seiring dengan perkembangan teknologi perangkat keras dan kebutuhan sistem yang semakin kompleks, berbagai algoritma penjadwalan telah dikembangkan untuk mengoptimalkan penggunaan CPU. Studi ini merupakan tinjauan sistematis literatur *(Systematic Literature Review)* terhadap berbagai pendekatan penjadwalan CPU, mencakup algoritma klasik seperti *First-Come, First-Served* (FCFS), *Shortest Job First* (SJF), *Priority Scheduling*, dan *Round Robin*, serta algoritma modern dan adaptif seperti *Completely Fair Scheduler* (CFS), *Multi-Level Feedback Queue* (MLFQ), dan *Dynamic Round Robin*. Selain itu, studi ini juga menelaah pendekatan berbasis klasterisasi proses dan penjadwalan pada sistem *multicore heterogen*. Hasil dari tinjauan ini menunjukkan bahwa algoritma adaptif dan berbasis konteks sistem memberikan peningkatan signifikan terhadap performa penjadwalan, terutama dalam hal waktu tunggu, *throughput*, dan efisiensi penggunaan prosesor. Temuan ini memberikan kontribusi penting dalam pengembangan algoritma penjadwalan yang lebih optimal dan kontekstual pada sistem komputasi modern.

Kata Kunci: algoritma adaptif modern; dynamic *Round Robin*; efisiensi sistem operasi; penjadwalan cpu linux; sistem multicore heterogen; systematic literature review.

I. PENDAHULUAN
Penjadwalan CPU merupakan komponen penting dalam system operasi untuk mengalokasikan sumber daya CPU secara efisien di antara berbagai proses yang berjalan. Algoritma penjadwalan seperti FCFS, SJF, dan *Round Robin* telah lama digunakan, masing-masing dengan kelebihan dan keterbatasannya dalam hal waktu respons, *throughput*, dan keadilan (Caranto et al., 2020; Harki, Ahmed, & Haji, 2020; Murad, Azmi, Muzahid, & Al-Imran, 2021; Omar, Jihad, & Hussein, 2021). Linux sendiri telah mengalami evolusi signifikan dalam mekanisme penjadwalannya, mulai dari O(1) *Scheduler* yang efisien namun kurang optimal dalam skenario interaktif, hingga *Completely Fair Scheduler* (CFS) yang mengutamakan keadilan jangka panjang (Jose, Sujisha, Gilesh, & Bindima, 2015; Singh & Kumar, 2015). Seiring berkembangnya kebutuhan sistem dan teknologi, pendekatan berbasis logika fuzzy dan pembelajaran mesin mulai digunakan untuk meningkatkan adaptivitas dan efisiensi penjadwalan (Han & Lee, 2020; Jalali Khalil Abadi & Mansouri, 2024; Tehsin et al., 2020). Dalam upaya menilai efektivitas berbagai algoritma ini secara sistematis, artikel ini menggunakan metode PRISMA, panduan yang diakui secara luas untuk tinjauan literatur sistematis (Moher, Altman, & Tetzlaff, 2014; Wang, 2025). Tujuan utama dari studi ini adalah untuk mengidentifikasi dan membandingkan algoritma penjadwalan CPU yang paling efektif dalam konteks sistem operasi Linux.

Remik: Riset dan E-Jurnal Manajemen Informatika Komputer
Volume 9, Number 3, Agustus 2025
http://doi.org/10.33395/remik.v9i3.14924
e-ISSN : 2541-1330
p-ISSN : 2541-1332

II. STRATEGI PENCARIAN LITERATUR

Strategi pencarian literatur dilakukan melalui dua basis data utama, yaitu Scopus dan Google Scholar. Proses pencarian menggunakan kombinasi kata kunci sebagai berikut: ("CPU scheduling" OR "process scheduling" OR "task scheduling" OR "job scheduling") AND ("algorithm" OR "method" OR "technique" OR "approach") AND ("comparison" OR "analysis" OR "evaluation" OR "assessment") AND ("Linux" OR "operating system" OR "OS" OR "Unix").

Pemilihan kata kunci tersebut didasarkan pada istilah-istilah teknis yang umum digunakan dalam studi mengenai algoritma penjadwalan CPU. Tujuan dari strategi ini adalah untuk memperoleh cakupan literatur yang luas, relevan, dan representatif terhadap topik kajian.

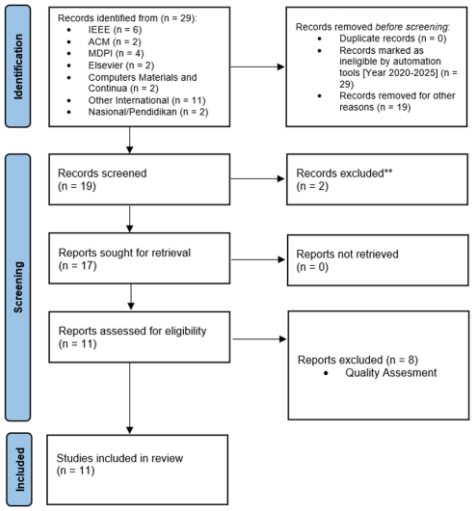
Tabel 1. Table Kriteria

E#	Kriteria	Penjelasan
E1	Topik sesuai dengan fokus SLR	Artikel membahas topik yang relevan dengan penelitian, yaitu algoritma penjadwalan CPU, khususnya pada sistem Linux atau sistem serupa, termasuk varian *Round Robin*, penjadwalan dinamis, multicore, real-time.
E2	Tahun publikasi antara 2020–2025	Artikel harus dipublikasikan dalam rentang tahun 2020 hingga 2025 untuk memastikan bahwa studi yang digunakan bersifat mutakhir dan relevan dengan kondisi terkini teknologi penjadwalan CPU.
E3	Akses terbuka (Open Access)	Artikel tersedia secara gratis dan bebas diakses, sehingga memungkinkan verifikasi dan replikasi hasil oleh peneliti lain, keterbukaan ilmiah.
E4	Subjek termasuk dalam Computer Science	Artikel terbit di jurnal yang termasuk dalam kategori Computer Science, khususnya yang mencakup sub bidang seperti algoritma, sistem operasi.
E5	Jenis dokumen adalah artikel ilmiah	Hanya artikel jurnal ilmiah yang dipertimbangkan, bukan review singkat, editorial, abstrak konferensi, atau laporan teknis, untuk menjamin kualitas dan kedalaman ilmiah.
E6	Status publikasi	Hanya artikel jurnal ilmiah yang dipertimbangkan, bukan review singkat, editorial, abstrak konferensi, atau laporan teknis, untuk menjamin kualitas dan kedalaman ilmiah.
E7	Judul Artikel tidak berbahasa Inggris	Artikel ditulis dalam Bahasa Inggris sebagai bahasa ilmiah internasional, agar dapat dipahami dan direview secara luas oleh komunitas akademik global.

Proses Pemilihan

Tujuan proses seleksi adalah mengumpulkan publikasi relevan tentang penjadwalan CPU, menilai kelayakannya, serta memastikan kualitas dan relevansi isi artikel. Seleksi dilakukan melalui empat tahap sesuai kerangka *PRISMA*: identifikasi, penyaringan, penilaian kelayakan, dan inklusi. Artikel hasil pencarian awal diseleksi berdasarkan judul dan abstrak, lalu dieliminasi jika duplikat atau tidak relevan. Artikel tersisa ditinjau secara menyeluruh berdasarkan kriteria inklusi. Proses ini dirangkum dalam Gambar 1 melalui diagram alur *PRISMA* yang menggambarkan jumlah publikasi yang diidentifikasi, disaring, dan akhirnya dimasukkan dalam tinjauan.

Remik: Riset dan E-Jurnal Manajemen Informatika Komputer
Volume 9, Number 3, Agustus 2025
http://doi.org/10.33395/remik.v9i3.14924
e-ISSN : 2541-1330
p-ISSN : 2541-1332



Gambar 1. Diagram alur *PRISMA* untuk tahapan seleksi studi.

Identifikasi

Tahap identifikasi diawali dengan penelusuran literatur dari berbagai sumber, menghasilkan total 29 artikel. Sumber-sumber tersebut mencakup: IEEE (6 artikel), ACM (2), MDPI (4), Elsevier (2), Computers Materials and Continua (2), jurnal internasional lainnya (11), serta jurnal nasional atau berbasis pendidikan (2). Namun, sebelum dilakukan penyaringan lebih lanjut, sebanyak 10 artikel dikeluarkan dari proses karena tidak memenuhi kriteria awal, di antaranya adalah publikasi otomatis yang tidak sesuai rentang tahun (2020–2025) sebanyak 29 artikel (sudah termasuk yang tumpang tindih), serta 19 artikel dihapus karena alasan lainnya. Setelah proses ini, tersisa 19 artikel untuk tahap penyaringan selanjutnya.

Penyaringan

Pada tahap penyaringan, Dari 29 artikel yang berhasil diidentifikasi, proses penyaringan dilakukan dengan meninjau judul dan abstrak untuk menilai kesesuaian awal terhadap topik penelitian. Pada tahap ini, 10 artikel dieliminasi karena tidak relevan, termasuk artikel yang berasal dari luar disiplin ilmu komputer, bukan artikel ilmiah, atau tidak tersedia dalam akses terbuka. Hasilnya, sebanyak 17 artikel dinyatakan layak untuk ditinjau lebih lanjut. Penyaringan dilakukan berdasarkan sejumlah kriteria eksklusi yang dirancang untuk menjamin kualitas dan relevansi publikasi, antara lain: artikel harus membahas algoritma penjadwalan CPU (E1), terbit dalam rentang tahun 2020–2025 (E2), tersedia secara open access (E3), berasal dari disiplin ilmu komputer (E4), merupakan artikel jurnal ilmiah (E5), telah melalui proses peer-review dan dipublikasikan secara final (E6), serta ditulis dalam Bahasa Inggris (E7). Tujuannya adalah memastikan hanya publikasi yang memenuhi standar akademik tinggi dan relevan secara langsung dengan fokus kajian yang dianalisis lebih lanjut.

Penilaian Kualitas

Setelah tahap penyaringan, dilakukan penilaian kualitas terhadap 17 artikel yang tersisa. Evaluasi ini mencakup aspek kelengkapan metodologi, kejelasan kontribusi terhadap pengembangan algoritma penjadwalan CPU, relevansi hasil penelitian, serta kesesuaiannya dengan bidang ilmu komputer. Penilaian juga mempertimbangkan keterbacaan artikel secara utuh dan kepatuhan terhadap standar akademik.

Dari proses ini, sebanyak 6 artikel dieliminasi karena tidak memenuhi kriteria kualitas yang telah ditetapkan, seperti kontribusi yang tidak jelas atau kurang mendalam. Dengan demikian, hanya 11 artikel yang dinyatakan memenuhi kriteria kelayakan dan dilanjutkan ke tahap analisis deskriptif. Proses ini penting untuk memastikan bahwa hanya literatur dengan integritas dan nilai ilmiah tinggi yang digunakan dalam tinjauan sistematis ini.

e-ISSN : 2541-1330
p-ISSN : 2541-1332

Ekstraksi Data

Proses ekstraksi data dilakukan terhadap 11 artikel yang telah memenuhi seluruh tahapan seleksi dan penilaian kualitas. Informasi yang dikumpulkan mencakup jenis algoritma penjadwalan yang dibahas, seperti *First-Come First-Served* (FCFS), *Shortest Job First* (SJF), *Round Robin* (RR), *Completely Fair Scheduler* (CFS), serta pendekatan hybrid dan berbasis kecerdasan buatan (AI). Selain itu, diekstraksi pula metode penelitian yang digunakan, mencakup simulasi, implementasi sistem nyata, dan evaluasi komparatif, beserta hasil analisis performa yang disajikan dalam masing-masing studi.

Untuk menunjang ketepatan dan keteraturan dalam pengumpulan data, digunakan perangkat lunak manajemen referensi Mendeley sebagai alat bantu. Seluruh data yang diperoleh dianalisis secara deskriptif guna mengidentifikasi pola umum, kecenderungan metodologis, serta kontribusi spesifik dari masing-masing penelitian terhadap pengembangan algoritma penjadwalan CPU, terutama dalam konteks sistem operasi Linux dan sistem serupa

III. METODE

Penelitian ini menggunakan metode Systematic Literature Review (SLR) untuk menganalisis perkembangan algoritma penjadwalan CPU secara sistematis dan objektif. Prosesnya mengikuti kerangka *PRISMA* (*Preferred Reporting Items for Systematic Reviews and Meta-Analyses*), yang meliputi tahapan identifikasi, penyaringan, penilaian kelayakan, dan inklusi (Page et al., 2022). *PRISMA* dirancang untuk membantu peneliti melaporkan secara transparan alasan dilakukannya tinjauan, apa yang dilakukan oleh penulis, dan apa yang ditemukan (Page et al., 2022). *PRISMA* 2020 menggantikan pernyataan tahun 2009 dan mencakup panduan pelaporan baru yang mencerminkan kemajuan dalam metode untuk mengidentifikasi, memilih, menilai, dan mensintesis studi (Page et al., 2022). Struktur dan penyajian item telah dimodifikasi untuk memfasilitasi implementasi (Page et al., 2022). Panduan ini juga mencakup daftar periksa 27 item yang diperluas, daftar periksa abstrak PRISMA 2020, dan diagram alir yang direvisi untuk tinjauan asli dan yang diperbarui (Page et al., 2022). Dengan menggunakan kerangka PRISMA, penelitian ini bertujuan untuk memberikan analisis yang komprehensif dan transparan mengenai algoritma penjadwalan CPU, serta mengidentifikasi kesenjangan penelitian dan memberikan rekomendasi untuk pengembangan di masa depan (Page et al., 2022).

IV. HASIL DAN PEMBAHASAN

Algoritma Penjadwalan CPU Tradisional dan Tujuannya

Algoritma penjadwalan CPU bertujuan memaksimalkan utilisasi CPU dan mengoptimalkan kinerja sistem operasi. Algoritma dasar seperti *First-Come, First Served* (FCFS), *Shortest Job First* (SJF), Priority Scheduling, dan *Round Robin* (RR) sering dijadikan tolok ukur. Misalnya, SJF memprioritaskan proses dengan *burst* terpendek, sehingga rata-rata waktu tunggu berkurang namun algoritma ini memerlukan informasi *burst* waktu yang sulit diperoleh di muka (Dirdal, Vo, Feng, & Davidrajuh, 2024). Sebaliknya, RR memberikan giliran waktu CPU kepada setiap proses secara bergiliran dengan *time quantum* tetap, sehingga menjamin respon awal cepat, tetapi berpotensi menghasilkan banyak *context switch*, *turnaround time* yang panjang, dan *throughput* rendah jika quantum terlalu kecil (Omar et al., 2021). *Completely Fair Scheduler* (CFS), yang digunakan Linux, merupakan algoritma preemptif berprioritas / fair yang memanfaatkan pohon penjadwalan untuk membagi CPU secara proporsional berdasar prioritas proses (Dirdal et al., 2024). Hasil studi simulasi multiprosesor menunjukkan bahwa CFS dan *algoritma Multi-Level Feedback Queue* (MLFQ) unggul dalam performa keseluruhan tanpa memerlukan pengetahuan *burst* masa depan, sehingga mengatasi kelemahan SJF yang memerlukan prediksi durasi proses (Dirdal et al., 2024).

Variasi *Round Robin* Dinamik

e-ISSN : 2541-1330
p-ISSN : 2541-1332

Banyak studi berfokus pada peningkatan RR dengan *time quantum* dinamik sesuai karakteristik proses. ADRR (Amended Dynamic RR) misalnya mengatur quantum secara siklik berdasarkan waktu *burst* tiap prosesi (Shafi et al., 2020). Shafi et al. (2020) mengusulkan ADRR yang "*cyclically adjusted*" quantum berdasarkan *burst* waktu, sehingga secara empiris menghasilkan rata-rata *waiting time* dan *turnaround time* yang lebih baik dibanding RR. konvensional, Improved RR, maupun varian RR multilevel lainnya (Shafi et al., 2020). (Fiad, Maaza, & Bendoukha, 2020) menggunakan model analitik yang mempertimbangkan *burst* tiap proses untuk mengatur quantum secara variabel, sehingga algoritma RR varian mereka meningkatkan rata-rata *waiting time* dan *turnaround time* tanpa memperbesar overhead (Fiad et al., 2020). Studi lain, RR adaptif neuro-fuzzy, juga menunjukkan perbaikan kinerja dengan mengatur quantum berdasar beban proses (lihat referensi dalam (Omar et al., 2021)). Kelebihan umum RR dinamik adalah fleksibilitas menurunkan *context switch* berlebihan dan mencegah proses pendek menunggu terlalu lama; kelemahannya, parameter penyesuaian yang kompleks dapat menambah overhead komputasi penjadwalan sendiri.

Pendekatan Berbasis Klaster (Clustering)

Beberapa penelitian inovatif memanfaatkan teknik klastering untuk mengelompokkan proses dengan karakteristik serupa, kemudian memberikan time slice khusus tiap klaster. Mostafa dan Amano (2020) memperkenalkan algoritma RR berbasis klaster: proses dikelompokkan (dengan K-Means atau model Bayesian) berdasarkan atribut seperti waktu layanan CPU dan *allowed time slice*, lalu setiap proses dalam klaster menerima quantum rata-rata klaster tersebut (Mostafa & Amano, 2020). Misalnya, algoritma DTS (Dynamic Time Slice) memberikan bobot dan quantum berbeda tiap proses, di mana klaster proses mirip dialokasikan quantum sesuai bobot dan *burst* mereka (Mostafa & Amano, 2020). Pendekatan *Adjustable Time Slice* (ATS) juga mengelompokkan proses berdasar fitur (*burst*, prioritas, dsb.) sehingga proses dengan profil serupa mendapat quantum yang sama (Mostafa & Amano, 2020). Hasil eksperimen menunjukkan algoritma klaster dapat menurunkan rata-rata *waiting time*, *turnaround time*, dan jumlah *context switch* dibanding RR standar maupun multilevel RR lainnya (Mostafa & Amano, 2020). Kelebihan pendekatan ini adalah menggabungkan fairness RR dengan efisiensi penjadwalan proses pendek. Kekurangannya, proses berelasi tidak mudah diidentifikasi tanpa metode klaster yang baik, dan pengelompokan berulang dapat menambah kompleksitas.

Algoritma Berlapis dan Prioritas Adaptif

Pendekatan lain adalah struktur antrian ganda atau bertingkat dengan prioritas adaptif. Sebagai contoh, MQAPS (*Multi-Queue Adaptive Priority Scheduling*) memperkenalkan antrian utama (RQ) untuk tugas baru dan antrian sekunder (SQ) untuk tugas yang sudah pernah dieksekusi dan prioritasnya turun (Iqbal et al., 2024). Waktu quantum dihitung dinamis berdasar prioritas; tugas dengan prioritas tinggi diselesaikan lebih cepat tanpa membiarkan tugas rendah starvation seperti pada RR sederhana (Iqbal et al., 2024). Simulasi menunjukkan MQAPS memanfaatkan sumber daya CPU lebih efisien daripada RR konvensional dan penjadwalan multilevel biasa, serta menunjukkan peningkatan dalam keadilan (fairness) dan efisiensi dalam penjadwalan tugas (Iqbal et al., 2024). Prinsip serupa juga dijalankan RRAPS (*Round Robin* Adaptive Priority Scheduling) yang memodifikasi quantum berdasar rata-rata *burst* tiap antrian (Iqbal et al., 2024). Kelebihan utama metode berlapis adalah keadilan lebih baik dan pencegahan starvation pada antrian prioritas rendah. Namun kompleksitas pengelolaan multi-antrian dan penghitungan prioritas adaptif menjadi tantangan implementasi di sistem nyata.

Penjadwalan Dalam Sistem Multicore Heterogen

Pada sistem multicore heterogen, penjadwalan harus mempertimbangkan kecepatan prosesor berbeda dan koherensi memori. Abbasi, Kamal, Gochoo, Jalal, dan Kim. (2021) mengusulkan dua metode: CBS (*Chunk-Based Scheduler*) dan QBICTM (*Quantum-Based Intra Core Task Migration*). Studi ini pertama-tama mengelompokkan tugas yang saling tergantung melalui model Bayesian, sehingga tugas berafinitas ditempatkan pada inti sama demi

e-ISSN : 2541-1330
p-ISSN : 2541-1332

memaksimalkan *cache* hit rate (Dirdal et al., 2024). CBS mengalokasikan tugas ke seluruh inti secara seimbang berdasarkan kecepatan masing-masing inti, sedangkan QBICTM melakukan migrasi tugas secara dinamis ke inti tercepat agar setiap tugas mendapat kesempatan adil eksekusi (Dirdal et al., 2024). Hasil eksperimen menunjukkan penerapan CBS atau QBICTM dapat meningkatkan kecepatan eksekusi rata-rata tugas sebesar 30–55% dibanding penjadwalan tradisional pada OS yang sama (Dirdal et al., 2024). Pendekatan ini efektif menyeimbangkan beban dan meminimalkan *overhead cache miss*, namun memerlukan analisis afinitas dan migrasi berulang yang menambah kompleksitas sistem.

Simulasi Berbasis Petri Net Untuk Penilaian Algoritma

Beberapa studi menggunakan simulasi berbasis Petri Net untuk mengevaluasi algoritma penjadwalan secara *multiprosesor*. Dirdal et al. (2024) mengembangkan platform simulasi Petri Net (GPenSIM) untuk menguji algoritma klasik (FCFS, SJF, RR) hingga algoritma kompleks seperti MLFQ dan Linux CFS (Dirdal et al., 2024). Simulasi menunjukkan bahwa SJF memang menurunkan *response time* ketika informasi *burst* tepat, namun kelemahannya yang memerlukan pengetahuan masa depan membuatnya tidak praktis. Sebaliknya CFS dan MLFQ tampil lebih unggul, karena keduanya tidak bergantung pada pengetahuan *burst* proses dan memberikan kinerja keseluruhan terbaik (Dirdal et al., 2024). Hasil ini merefleksikan prinsip Linux CFS yang berorientasi keadilan dan kinerja stabil di lingkungan multiprosesor. Simulasi Petri Net menekankan bahwa optimasi penjadwalan (mis. meminimalkan *turnaround time* dan *response time*) dapat dicapai dengan desain algoritma yang memadukan *fairness* dan dinamika adaptif berdasarkan beban kerja nyata.

Ringkasan Kekuatan, Kelemahan, Dan Konteks Pengguna

Setiap algoritma memiliki kelebihan dan batasan sesuai konteksnya. RR dan variannya cocok untuk sistem time-sharing karena kesederhanaan dan fairness, tetapi memerlukan penyesuaian quantum untuk menghindari banyak *context switch* (Omar et al., 2021). Modifikasi dinamik (ADRR, ATS, variabel quantum) meningkatkan efisiensi waktu tunggu dan respons dengan memberikan waktu CPU lebih pada proses pendek, namun menambah kompleksitas penghitungannya (Shafi et al., 2020). Pendekatan berbasis klaster memanfaatkan kemiripan proses untuk optimasi bersama, sangat efektif di beban heterogen namun memerlukan data proses yang baik dan algoritma klastering yang valid (Mostafa & Amano, 2020). Metode berlapis seperti MQAPS memberikan framework fleksibel untuk menghindari starvation dan menjaga keadilan, cocok untuk sistem *real-time* dengan prioritas berjenjang (Iqbal et al., 2024). Dalam sistem multicore, algoritma afinitas (CBS/QBICTM) mengurangi *overhead memory* dan memanfaatkan karakteristik *hardware*, penting pada lingkungan heterogen dengan *cache* privat dan shared (Dirdal et al., 2024). Simulasi Petri Net menegaskan bahwa algoritma modern seperti CFS mampu menyeimbangkan efisiensi dan keadilan, sehingga menjadi pilihan default di banyak kernel Linux (Dirdal et al., 2024). Secara keseluruhan, penelitian terbaru menunjukkan bahwa penggabungan teknik adaptif (klaster, prioritas dinamis, simulasi formal) dapat menurunkan waktu tunggu, waktu respons, dan *context switch* sambil memaksimalkan utilisasi CPU di berbagai skenario sistem operasi (Shafi et al., 2020).

V. KESIMPULAN

Penjadwalan CPU tetap menjadi aspek krusial dalam optimalisasi performa sistem operasi, khususnya dalam lingkungan komputasi yang semakin kompleks dan dinamis. Studi ini menunjukkan bahwa algoritma penjadwalan tradisional seperti FCFS, SJF, dan RR, meskipun sederhana dan menjadi dasar historis, memiliki keterbatasan dalam konteks beban kerja modern, terutama terkait kebutuhan informasi *burst* time dan potensi tingginya *context switch*. Berbagai pendekatan baru telah dikembangkan untuk mengatasi kelemahan tersebut. Algoritma *Round Robin* dinamis (seperti ADRR dan ATS) memperbaiki efisiensi waktu tunggu dan respons dengan penyesuaian quantum berbasis karakteristik proses. Teknik berbasis klaster dan prioritas adaptif menawarkan efisiensi yang lebih tinggi dengan mengelompokkan proses dan mengalokasikan

quantum secara lebih cerdas. Pendekatan berlapis seperti MQAPS juga berhasil menjaga keadilan dan menghindari starvation. Pada sistem *multicore heterogen*, penjadwalan berbasis afinitas seperti CBS dan QBICTM terbukti efektif dalam menyeimbangkan beban kerja dan meminimalkan overhead *cache*. Simulasi Petri Net memperkuat validitas performa dari algoritma seperti MLFQ dan Linux CFS dalam berbagai kondisi, memperlihatkan keunggulan dalam fairness dan utilisasi CPU. Secara umum, arah penelitian saat ini mengarah pada integrasi teknik adaptif, analitik, dan simulatif untuk menghasilkan algoritma penjadwalan yang responsif, adil, dan efisien terhadap beragam beban kerja. Temuan ini memberikan dasar penting bagi pengembangan sistem operasi modern yang lebih cerdas dan adaptif.

VI. UCAPAN TERIMA KASIH

Penulis menyampaikan penghargaan dan terima kasih yang sebesar-besarnya kepada semua pihak yang telah memberikan dukungan selama proses penyusunan penelitian ini. Ucapan terima kasih khusus ditujukan kepada Bapak Herbert Siregar, selaku dosen pembimbing, atas arahan, bimbingan, serta motivasi yang sangat berarti dalam menyelesaikan studi ini.

Penulis juga berterima kasih kepada sivitas akademika Program Studi Ilmu Komputer, Universitas Pendidikan Indonesia, atas dukungan lingkungan akademik yang kondusif serta diskusi-diskusi ilmiah yang memperkaya pemahaman penulis. Semoga kontribusi kecil ini dapat memberikan manfaat bagi pengembangan ilmu pengetahuan, khususnya di bidang sistem operasi dan penjadwalan CPU.

VII. REFERENSI

Abbasi, Sohaib Iftikhar, Kamal, Shaharyar, Gochoo, Munkhjargal, Jalal, Ahmad, & Kim, Kibum. (2021). Affinity-based task scheduling on heterogeneous multicore systems using cbs and qbictm. *Applied Sciences (Switzerland)*, *11*(12), 5740. https://doi.org/10.3390/app11125740

Caranto, Hannah S., Olivete, Wyli Charles L., Fernandez, Jeune Vincent D., Cabiara, Cecilia Agatha R., Baquirin, Rey Benjamin M., Bayani, Eugene Frank G., & Fronda, Roma Joy D. (2020). Integrating user-defined priority tasks in a *Shortest Job First Round Robin* (sjfrr) scheduling algorithm. *ACM International Conference Proceeding Series*, 9–13. https://doi.org/10.1145/3379247.3379291

Dirdal, Daniel Osmundsen, Vo, Danny, Feng, Yuming, & Davidrajuh, Reggie. (2024). Developing a platform using petri nets and gpensim for simulation of multiprocessor scheduling algorithms. *Applied Sciences (Switzerland)*, *14*(13), 5690. https://doi.org/10.3390/app14135690

Fiad, Alaa, Maaza, Zoulikha Mekkakia, & Bendoukha, Hayat. (2020). Improved version of *Round Robin* scheduling algorithm based on analytic model. *International Journal of Networked and Distributed Computing*, *8*(4), 195–202. https://doi.org/10.2991/IJNDC.K.200804.001

Han, Junyeong, & Lee, Sungyoung. (2020). Performance improvement of linux cpu scheduler using policy gradient reinforcement learning for android smartphones. *IEEE Access*, *8*, 11031–11045. https://doi.org/10.1109/ACCESS.2020.2965548

Harki, Naji, Ahmed, Abdulraheem, & Haji, Lailan. (2020). Cpu scheduling techniques: A review on novel approaches strategy and performance assessment. *Journal of Applied Science and Technology Trends*, *1*(1), 48–55. https://doi.org/10.38094/jastt1215

Iqbal, Mansoor, Shafiq, Muhammad Umar, Khan, Shouzab, Obaidullah, Alahmari, Saad, & Ullah, Zahid. (2024). Enhancing task execution: A dual-layer approach with multi-queue adaptive priority scheduling. *PeerJ Computer Science*, *10*, e2531. https://doi.org/10.7717/peerj-cs.2531

Jalali Khalil Abadi, Zahra, & Mansouri, Najme. (2024). A comprehensive survey on scheduling algorithms using fuzzy systems in distributed environments. *Artificial Intelligence Review*, *57*(1), 4. https://doi.org/10.1007/s10462-023-10632-y

Jose, Jyothish, Sujisha, Oravanpadath, Gilesh, Malayamparambath, & Bindima, Thayyil. (2015). On the fairness of linux O(1) scheduler. *Proceedings - International Conference on Intelligent Systems, Modelling and Simulation, ISMS*, *2015-Septe*, 668–674. https://doi.org/10.1109/ISMS.2014.120

Moher, David, Altman, Douglas G., & Tetzlaff, Jennifer. (2014). PRISMA (Preferred Reporting Items for Systematic Reviews and Meta-Analyses). *Guidelines for Reporting Health Research: A User's Manual*, *1999*, 250–261.

Mostafa, Samih M., & Amano, Hirofumi. (2020). An adjustable variant of *Round Robin* algorithm based on clustering technique. *Computers, Materials and Continua*, *66*(3), 3253–3270. https://doi.org/10.32604/cmc.2021.014675

Murad, Saydul Akbar, Azmi, Zafril Rizal M., Muzahid, Abu Jafar Md, & Al-Imran, Md. (2021). Comparative study on job scheduling using priority rule and machine learning. *2021 Emerging Technology in Computing, Communication and Electronics, ETCCE 2021*, 1–8. https://doi.org/10.1109/ETCCE54784.2021.9689812

Omar, Hoger K., Jihad, Kamal H., & Hussein, Shalau F. (2021). Comparative analysis of the essential cpu scheduling algorithms. *Bulletin of Electrical Engineering and Informatics*, *10*(5), 2742–2750. https://doi.org/10.11591/eei.v10i5.2812

Page, Matthew J., McKenzie, Joanne E., Bossuyt, Patrick M., Boutron, Isabelle, Hoffmann, Tammy C., Mulrow, Cynthia D., Shamseer, Larissa, Tetzlaff, Jennifer M., Akl, Elie A., Brennan, Sue E., Chou, Roger, Glanville, Julie, Grimshaw, Jeremy M., Hróbjartsson, Asbjørn, Lalu, Manoj M., Li, Tianjing, Loder, Elizabeth W., Mayo-Wilson, Evan, McDonald, Steve, McGuinness, Luke A., Stewart, Lesley A., Thomas, James, Tricco, Andrea C., Welch, Vivian A., Whiting, Penny, & Moher, David. (2022). The PRISMA 2020 statement: an updated guideline for reporting systematic reviews. *Revista Panamericana de Salud Publica/Pan American Journal of Public Health*, *46*. https://doi.org/10.26633/RPSP.2022.112

Shafi, Uferah, Shah, Munam, Wahid, Abdul, Abbasi, Kamran, Javaid, Qaisar, Asghar, Muhammad, & Haider, Muhammad. (2020). A novel amended dynamic *Round Robin* scheduling algorithm for timeshared systems. *International Arab Journal of Information Technology*, *17*(1), 90–98. https://doi.org/10.34028/iajit/17/1/11

Singh, Shirish, & Kumar, Praveen. (2015). CFS performance improvement using binomial heap. *2015 International Conference on Advances in Computing, Communications and Informatics, ICACCI 2015*, 1822–1824. https://doi.org/10.1109/ICACCI.2015.7275881

Tehsin, Sara, Asfia, Yame, Akbar, Naeem, Riaz, Farhan, Rehman, Saad, & Young, Rupert C. D. (2020). Selection of CPU scheduling dynamically through machine learning. *Pattern Recognition and Tracking XXXI*, *11400*, 24. https://doi.org/10.1117/12.2559540

Wang, Clarence Wenfeng. (2025). Writing systematic reviews. In *English for Medical Communication: A Guide to Course Design*. https://doi.org/10.4324/9781003375654-11