JITEL (Jurnal Ilmiah Telekomunikasi, Elektronika, dan Listrik Tenaga) p-ISSN: 2774-7972
Vol. 3, No. 2 IRWNS 2023, pp. 127-134 e-ISSN: 2775-6696
DOI: https://doi.org/10.35313/jitel.v3.i2.2023.127-134

Implementasi algoritma LFUDA pada arsitektur NDN

Taufik Irfan[1*], Slameta[2], Rifa Hanifatunnisa[3]
[1,2,3]Jurusan Teknik Elektro, Politeknik Negeri Bandung
Jl. Gegerkalong Hilir, Ciwaruga, Kec. Parongpong, Kabupaten Bandung Barat, Jawa Barat 40559, Indonesia
[1*]taufik.irfan@polban.ac.id, [2]slameta@polban.ac.id,[3]rifahani@polban.ac.id

ABSTRAK

NDN (*Named Data Network*) merupakan sebuah arsitektur jaringan yang menerapkan prinsip *content-centric*, yaitu pengguna mendapatkan data berdasarkan penamaan konten, bukan berdasarkan *server* tertentu. Keefektifan dari prinsip ini membuat router NDN memiliki kemampuan untuk memenuhi permintaan pengguna, tanpa perlu merujuk ke server. Dalam skenario tersebut, pentingnya algoritma yang efisien untuk mengelola penghapusan konten dari *cache* menjadi sangat krusial. Untuk menangani masalah tersebut, penelitian ini mengadopsi algoritma *caching Least Frequently Used with Dynamic Aging* (LFUDA) dalam lingkungan arsitektur NDN. Simulasi yang digunakan dalam penelitian ini adalah ndnSIM, dan beberapa parameter utama diukur dalam skenario berbagai tingkat interest dan kapasitas penyimpanan konten. Hasil penelitian menunjukkan bahwa, dibandingkan dengan algoritma *caching* FIFO (*First In First Out*) dan LRU (*Least Recently Used*), LFUDA mampu mencapai rasio *hit rate* yang lebih tinggi dan nilai rata-rata *delay* dan *hop* yang lebih rendah. Spesifik untuk skenario dengan 10 *interest*, LFUDA menunjukkan keunggulan sebesar 4,24% dibandingkan FIFO dan 4,09% dibandingkan LRU. Sementara itu, pada skenario dengan kapasitas penyimpanan 50 paket, *delay* rata-rata LFUDA sekitar 61% lebih rendah dibandingkan FIFO dan LRU. Hasil ini mengindikasikan superioritas LFUDA sebagai algoritma manajemen *caching* di dalam konteks NDN, meskipun pengembangan lebih lanjut masih dibutuhkan untuk peningkatan performa dalam berbagai kondisi interest dan kapasitas penyimpanan.

Kata kunci: NDN, LFUDA, *caching*, ndnSIM

ABSTRACT

Named Data Networking (NDN) is a network architecture that applies a content-centric paradigm, where users request data based on content naming, rather than a specific server. This efficiency results in NDN routers' capability to fulfill user requests, eliminating the need for server-specific referrals. Consequently, the deployment of an efficient algorithm for content eviction from the cache becomes crucial. In addressing this issue, this research deploys the Least Frequently Used with Dynamic Aging (LFUDA) caching algorithm within the NDN architectural environment. The ndnSIM simulator was used in the research, with critical parameters measured across scenarios with varying levels of interest and content storage capacities. The findings reveal that, when compared with the FIFO (First In First Out) and LRU (Least Recently Used) caching algorithms, LFUDA achieves a higher hit rate ratio and lower average delay and hop values. Specifically, in a scenario with 10 interests, LFUDA demonstrated a superior performance of 4.24% compared to FIFO and 4.09% compared to LRU. Meanwhile, in a scenario with a storage capacity of 50 packets, LFUDA's average delay was about 61% lower than both FIFO and LRU. These results reinforce the prominence of LFUDA as a caching management algorithm within the NDN context, although further advancements are still necessary to improve performance under various interest and storage capacity conditions.

Keywords: *NDN, LFUDA, caching, ndnSIM*

1. PENDAHULUAN

Peningkatan penggunaan jaringan internet secara global telah menyebabkan lonjakan permintaan konten multimedia. Menurut survei Cisco, konsumsi konten video pada tahun 2017 mencakup 75% dari total lalu lintas internet di dunia. Pada tahun 2022, Cisco memprediksi bahwa konsumsi konten video akan mencapai 82% dari total lalu lintas internet [1]. Pertumbuhan ini dapat menghasilkan beban lalulintas internet yang lebih besar. Oleh karena itu, diperlukan solusi untuk mengurangi beban pada jaringan internet sehingga dapat beroperasi dengan lebih efisien dan cepat.

Naskah diterima tanggal 18 Juli 2023, dipresentasikan tanggal 25 Juli 2023
*E-mail korespondensi

Konsep baru dalam pengembangan internet, yaitu *Named Data Network* (NDN), telah muncul sebagai pengembangan dari *Content Centric Networking* (CCN) yang diperkenalkan oleh Van Jacobson [2]. Ini menawarkan alternatif dalam jaringan internet. Dalam arsitektur NDN, jaringan internet tidak lagi berdasarkan IP untuk meminta data atau memenuhi permintaan oleh server sebagai penyedia konten, tetapi berdasarkan nama konten. Hal ini dapat menghasilkan efisiensi yang lebih tinggi karena permintaan data tidak lagi ditujukan pada IP spesifik, tetapi ke lokasi di mana konten tersebut berada. Ini mengubah konsep dari 'di mana' data diperlukan menjadi 'apa' data yang diperlukan. Perbandingan performa antara jaringan NDN dan internet berbasis IP dapat dilihat di [3].

Pada NDN, router memiliki peran penting dalam menyimpan data yang diminta oleh pengguna. Data ini kemudian dapat digunakan untuk memenuhi permintaan dari pengguna lain, yang dapat mengurangi beban lalu lintas internet yang dihasilkan oleh permintaan konten dinamis. Namun, router NDN memiliki keterbatasan dalam kapasitas penyimpanan, sehingga diperlukan fungsi caching pada setiap router. Algoritma caching yang efisien dapat menentukan konten mana yang harus dihapus saat penyimpanan penuh, sehingga dapat mengurangi beban pada penyedia konten. Selain itu, caching yang efisien juga dapat mengurangi jumlah hop dan delay dalam pengiriman data.

Algoritma *caching* berdasarkan popularitas atau frekuensi, yaitu *Least Frequently Used with Dynamic Aging* (LFUDA), telah banyak diterapkan dalam sistem *cache memory* berbagai aplikasi, termasuk sistem operasi, sistem basis data, dan pengiriman konten multimedia [4]. Algoritma ini sering digunakan dalam lingkungan yang membutuhkan penanganan efisien atas permintaan data yang berubah-ubah. Penggunaan algoritma LFUDA pada NDN adalah suatu penelitian yang cukup baru dan belum banyak dilakukan [5]. Dalam penelitian ini, algoritma LFUDA diimplementasikan menggunakan simulator ndnSIM dengan bahasa pemrograman C++. Ada tiga parameter yang diuji dalam penelitian ini, yaitu rata-rata *delay*, *hit rate ratio*, dan rata-rata *hop* per pengiriman. Selain itu, algoritma ini juga dibandingkan dengan algoritma caching lain yang sudah ada pada simulator ndnSIM, yaitu *First In First Out* (FIFO) dan *Least Recently Used* (LRU) [6].

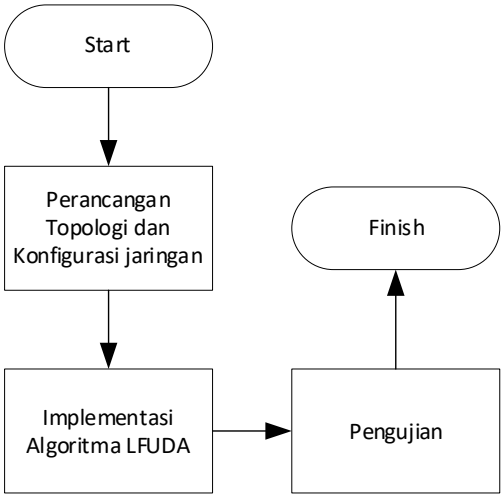
NDN semakin mendapat perhatian sebagai solusi potensial terhadap tantangan skala yang dialami oleh jaringan internet saat ini. Selain itu, penelitian sebelumnya juga telah menunjukkan bahwa pendekatan berbasis konten seperti NDN dapat memberikan peningkatan efisiensi dan kinerja jaringan, khususnya dalam hal penanganan konten multimedia dan layanan berbasis video [7]. Dengan desain yang memfokuskan pada 'apa' data yang diperlukan daripada 'di mana' data diperoleh, NDN memungkinkan penggunaan teknologi caching secara lebih efektif untuk mengurangi latensi dan beban jaringan. Algoritma *caching* seperti LFUDA, yang memprioritaskan penghapusan item yang paling jarang digunakan dengan memperhitungkan penuaan dinamis, dapat memanfaatkan struktur NDN untuk peningkatan kinerja lebih lanjut [8]. Penelitian ini bertujuan untuk mengimplementasikan algoritma LFUDA pada arsitektur NDN. Simulasi dalam penelitian ini menggunakan ndnSIM dan beberapa parameter utama diukur dalam skenario berbagai tingkat *interest* dan kapasitas penyimpanan konten.

2. METODE PENELITIAN

Penelitian ini mengikuti beberapa tahapan penting yang digambarkan dalam diagram alir Gambar 1. Langkah-langkah tersebut memiliki peran masing-masing dalam implementasi dan evaluasi algoritma LFUDA.

Pertama, langkah "Perancangan Topologi dan Konfigurasi Jaringan" berfungsi sebagai dasar dalam penelitian ini. Dalam tahapan ini, topologi Polban dibangun dengan menggunakan simulator ndnSIM, sebuah perangkat simulasi yang dibuat khusus untuk studi terkait NDN. Setiap node di dalam jaringan tersebut kemudian dikonfigurasi sebagai produsen atau konsumen data, dan parameter jaringan seperti tipe *routing* dan *forwarding* ditentukan. Tipe routing adalah strategi yang digunakan dalam menentukan jalur terbaik bagi data untuk bergerak antar titik di dalam jaringan, sementara tipe forwarding merujuk pada bagaimana data tersebut disalurkan melalui node-node jaringan.

Setelah pembuatan dan konfigurasi topologi jaringan, proses penelitian bergerak ke langkah "Implementasi Algoritma LFUDA". LFUDA, atau *Least Frequently Used with Dynamic Aging*, adalah strategi pengelolaan *cache* yang memprioritaskan penghapusan item yang jarang digunakan dan telah lama berada dalam *cache*. Implementasi algoritma ini pada topologi Polban melibatkan penyesuaian mekanisme *cache* simulator ndnSIM sehingga setiap interaksi data, baik permintaan maupun respon, akan mematuhi aturan yang ditentukan oleh algoritma LFUDA.



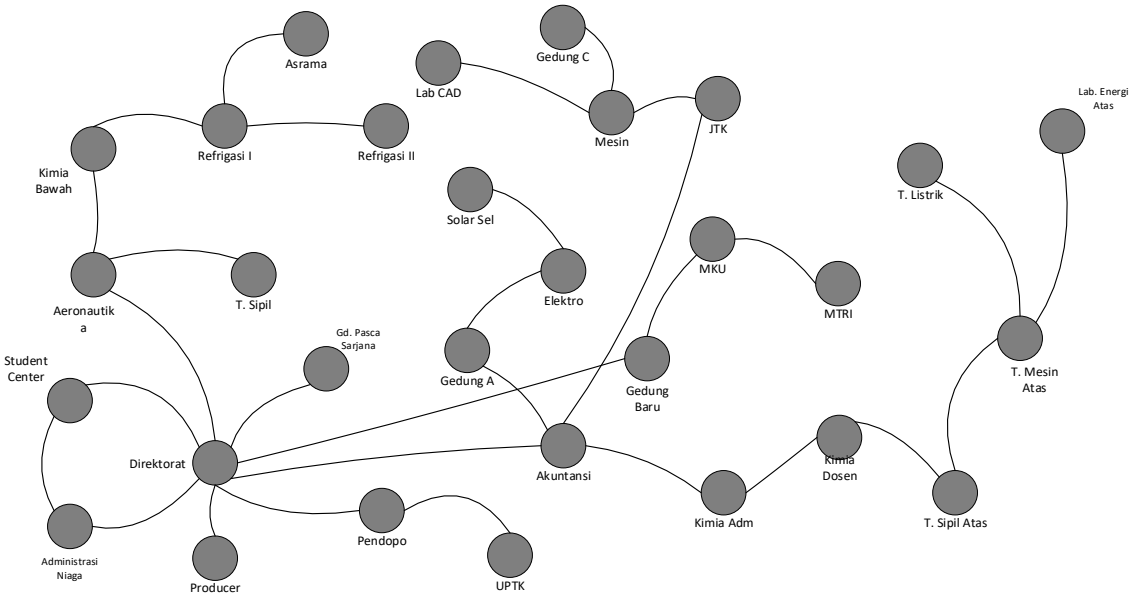
Gambar 1. Diagram alir penelitian

Tahap terakhir penelitian adalah "Pengujian". Di sini, sejumlah parameter penting diuji untuk mengevaluasi kinerja algoritma LFUDA dalam simulasi. Parameter-parameter tersebut meliputi rata-rata delay, yang merujuk pada waktu rata-rata yang diperlukan untuk memproses permintaan data; *hit rate ratio*, yang menunjukkan proporsi permintaan data yang dapat dipenuhi oleh *cache*; dan rata-rata hop per pengiriman, yang menggambarkan jumlah rata-rata langkah yang diperlukan oleh paket data untuk mencapai tujuan. Untuk membantu proses ini, simulator ndnSIM menyediakan *helper*, yakni alat yang digunakan untuk mengumpulkan dan menganalisis data selama simulasi berjalan.

Melalui penjabaran langkah-langkah metodologi penelitian ini, diharapkan dapat memberikan pemahaman yang lebih mendalam tentang bagaimana algoritma LFUDA diimplementasikan dan bagaimana kinerjanya diukur dalam konteks NDN.

2.1 Perancangan Topologi dan Konfigurasi Jaringan

Dalam penelitian ini, topologi jaringan yang dipilih adalah topologi Polban, seperti yang ditampilkan pada Gambar 2. Topologi Polban terdiri dari 30 *node* yang dibagi menjadi *node* konsumen dan *node* produsen. Terdapat 29 *node* konsumen yang berfungsi sebagai *router* NDN dan satu *node* produsen yang bertindak sebagai penyedia konten.



Gambar 2. Topologi Polban

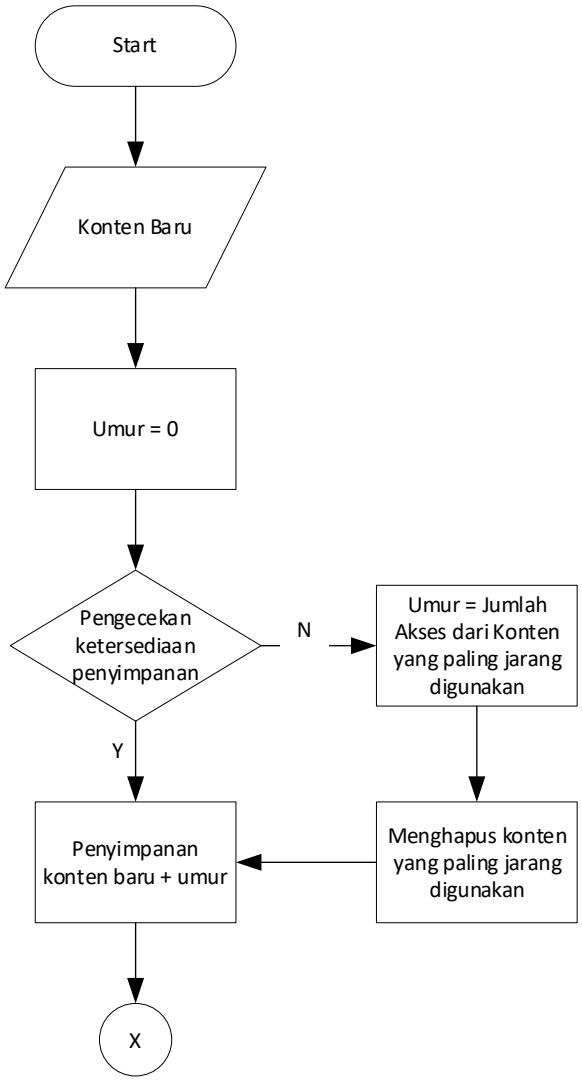
Selain itu, penelitian ini juga mengonfigurasi parameter, jenis *routing*, dan jenis *forwarding* yang akan digunakan dalam topologi Polban. Tabel 1 menyajikan parameter-parameter yang digunakan dalam simulasi.

Tabel 1. Parameter simulasi

No	Parameter	Nilai
1	*Data rate*	1 Mbps
2	*Payload*	1024 *bytes*
3	Waktu simulasi	100 detik
4	Tipe *forwarding*	*Best-route*
5	TIpe *routing*	*ndn Global routing*

2.2 Implementasi Algoritma LFUDA

Algoritma *caching* yang akan digunakan dalam penelitian ini adalah algoritma LFUDA. Algoritma LFUDA adalah perkembangan dari algoritma LFU, yang menghapus konten dalam penyimpanan yang sudah penuh berdasarkan frekuensi akses terendah. LFUDA berbeda dari LFU karena memiliki sistem 'umur' dinamis yang dirancang untuk mengatasi masalah polusi caching dalam algoritma LFU [5]. 'Umur' dalam algoritma ini mempengaruhi nilai awal konten baru yang akan disimpan, di mana nilainya tidak dimulai dari 1 seperti dalam LFU.



Gambar 3. Diagram alir algoritma caching LFUDA

Gambar 3 menampilkan diagram alir dari algoritma LFUDA. Implementasi algoritma ini dilakukan saat penyimpanan sudah penuh, dan diperlukan penghapusan konten. Di dalam algoritma LFUDA, ada variabel 'umur' dengan nilai yang dinamis. Selama penyimpanan belum penuh, variabel 'umur' ini akan tetap nol. Saat penyimpanan penuh, algoritma LFUDA akan menentukan konten dengan frekuensi akses terendah untuk dihapus. Frekuensi akses terendah ini menjadi 'umur' dari algoritma LFUDA. Kemudian, konten baru disimpan dalam penyimpanan dengan frekuensi yang ditambah dengan 'umur' algoritma.

2.3 Pengujian Algoritma LFUDA

Situasi pengujian dalam penelitian ini dibagi menjadi dua kondisi, yaitu kondisi di mana ukuran *interest* berubah-ubah dan kondisi di mana kapasitas penyimpanan konten berubah-ubah. Tabel 2 memberikan detail lebih lanjut mengenai kondisi pengujian ini.

Tabel 2. Kondisi pengujian parameter

Skenario	*Interest* (Permintaan)	Besar penyimpanan (paket)	Waktu simulasi (detik)
Kondisi 1	10	30	100
	50		
	100		
Kondisi 2	50	10	
		30	
		50	

2.4 Parameter Uji

Berikut merupakan parameter yang akan diuji untuk mengukur performansi algoritma LFUDA:

1. Rata-rata *delay*

$$rata - rata\ delay = \frac{jumlah\ delay}{jumlah\ pengiriman} \quad (1)$$

Parameter rata-rata delay didapatkan dari rentang waktu dari permintaan interest yang dilakukan oleh consumer sampai paket data tersebut diterima lagi oleh *consumer* dalam satuan waktu.

2. *Hit rate ratio*

$$hitrate\ ratio = \frac{hitrate}{hitrate+hitmiss} \times 100\% \quad (2)$$

Parameter *hit rate ratio* menunjukan permintaan konten yang dilayani dibandingkan dengan jumlah total permintaan.

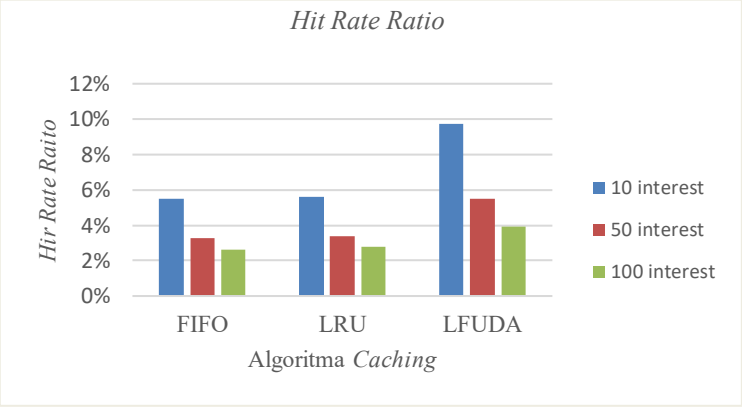
3. Rata-rata *hop*

$$rata - rata\ hop = \frac{jumlah\ hop}{jumlah\ pengiriman} \quad (3)$$

Parameter jumlah *hop* menghitung banyaknya node yang dilewati dari dikirimnya *interest* sampai diterima oleh consumer.

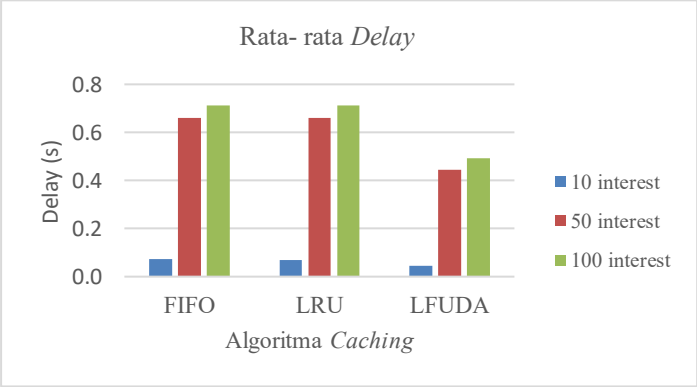
3. HASIL DAN PEMBAHASAN

Hasil pengujian yang dilakukan dalam dua kondisi yang berbeda memperlihatkan variasi dalam nilai rata-rata delay, *hit rate ratio*, dan rata-rata *hop* yang diperoleh untuk setiap skenario pengujian. Kedua kondisi pengujian yang diuji mencakup situasi di mana interest berubah-ubah dan situasi di mana penyimpanan konten berubah-ubah, untuk memberikan analisis yang komprehensif tentang kinerja algoritma dalam kondisi yang berbeda.



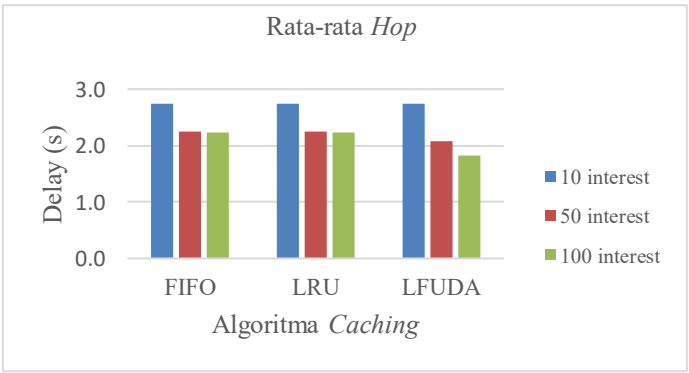
Gambar 4. Hasil pengujian *hit rate ratio* pada kondisi 1

Gambar 4 menunjukkan *hit rate ratio* algoritma LFUDA yang lebih besar dibandingkan dua algoritma lainnya di setiap skenario. Selisih paling signifikan terjadi pada skenario dengan 10 *interest*, di mana algoritma LFUDA unggul 4,24% dibandingkan algoritma FIFO dan 4,09% dibandingkan algoritma LRU. Semakin tinggi nilai *interest*, semakin kecil perbedaan antara algoritma LFUDA dan dua algoritma lainnya.



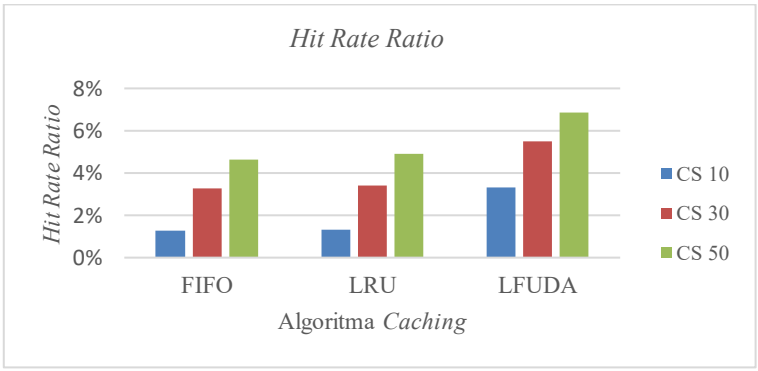
Gambar 5. Hasil pengujian rata-rata *delay* pada kondisi 1

Gambar 5 menunjukkan bahwa rata-rata *delay* untuk algoritma LFUDA lebih rendah dibandingkan dua algoritma lainnya di setiap skenario. Algoritma LFUDA tidak menunjukkan peningkatan rata-rata *delay* yang signifikan seperti algoritma LRU dan FIFO saat ukuran interest dinaikkan.



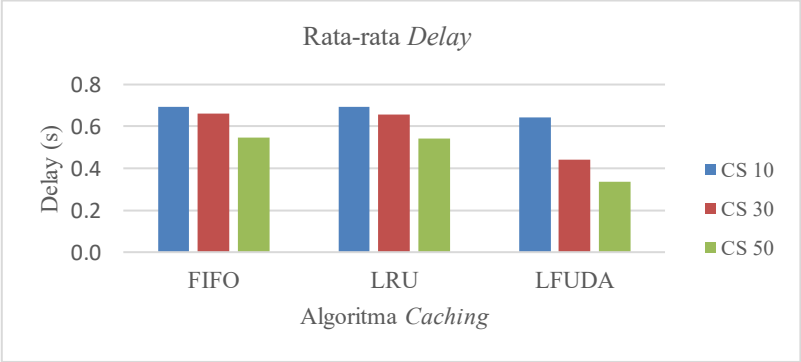
Gambar 6. Hasil pengujian rata-rata *hop* pada kondisi 1

Gambar 6 menunjukkan bahwa rata-rata *hop* algoritma LFUDA secara umum lebih rendah, kecuali pada saat interest 10. Pada interest 10, rata-rata hop LFUDA sekitar 0,03% lebih tinggi dibandingkan FIFO dan sama dengan LRU. Namun, saat ukuran interest dinaikkan, selisih rata-rata hop menjadi lebih signifikan.



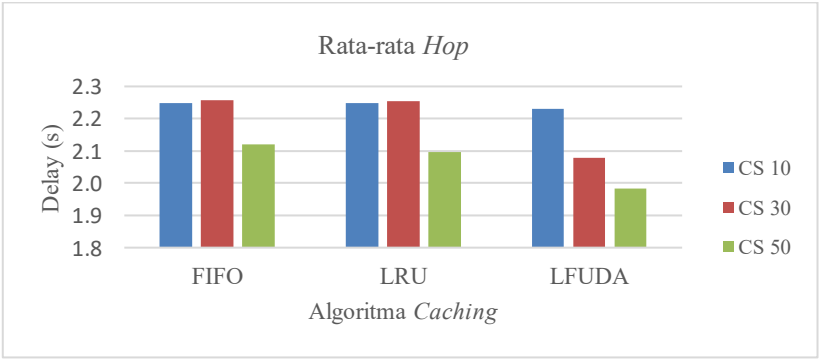
Gambar 7. Hasil pengujian *hit rate ratio* pada kondisi 2

Gambar 7 menunjukkan bahwa *hit rate ratio* algoritma LFUDA selalu lebih besar dibandingkan dua algoritma lainnya di setiap skenario. Selisih hit rate ratio LFUDA selalu sekitar 2% lebih besar dibandingkan FIFO dan LRU.



Gambar 8. Hasil pengujian rata-rata *delay* pada kondisi 2

Gambar 8 menunjukkan bahwa rata-rata *delay* algoritma LFUDA selalu lebih rendah dibandingkan dua algoritma lainnya di setiap skenario. Pada skenario ini, perbedaan terbesar terjadi saat kapasitas penyimpanan sebesar 50 paket, yaitu sekitar 61% lebih rendah dibandingkan FIFO dan LRU. Saat kapasitas penyimpanan lebih kecil, rata-rata delay menjadi lebih rendah.



Gambar 9. Hasil pengujian rata-rata *hop* pada kondisi 2

Gambar 9 menunjukkan nilai rata-rata *hop* untuk algoritma FIFO, LRU, dan LFUDA. Tidak ada perbedaan yang signifikan antara ketiga algoritma tersebut. Namun, algoritma LFUDA selalu memiliki rata-rata *hop* yang lebih rendah dibandingkan dua algoritma lainnya. Perbedaan rata-rata hop LFUDA paling besar terjadi saat ukuran interest 50, di mana LFUDA 8,53% lebih rendah dibandingkan FIFO dan 8,37% lebih rendah dibandingkan LRU.

4. KESIMPULAN

Berdasarkan serangkaian uji coba yang telah dilakukan dalam penelitian ini, dapat disimpulkan bahwa algoritma LFUDA memiliki kinerja yang superior dibandingkan dengan algoritma LRU dan FIFO di semua skenario yang diuji. LFUDA secara konsisten mencatat *hit rate ratio* yang lebih tinggi, menunjukkan efektivitasnya dalam memanfaatkan *cache*, terlepas dari perubahan *interest* dan kapasitas penyimpanan. Algoritma ini terbukti sangat efektif dalam menanggapi permintaan data yang fluktuatif, yang sangat penting dalam lingkungan jaringan yang dinamis. LFUDA juga mencatat rata-rata *delay* yang lebih rendah, menunjukkan efisiensi waktu respons yang lebih baik, terutama saat permintaan data tinggi atau kapasitas penyimpanan lebih besar. Dalam sebagian besar skenario yang diuji, LFUDA menunjukkan dominasi yang signifikan, dengan kinerja yang umumnya melebihi LRU dan FIFO lebih dari 40%. Namun, dalam satu skenario tertentu, keunggulan kinerja LFUDA berkurang menjadi sekitar 8%. Meskipun demikian, penelitian ini secara umum menunjukkan bahwa algoritma LFUDA memberikan peningkatan signifikan dalam hal efisiensi dan kecepatan respons dibandingkan dengan algoritma *caching* lainnya dalam konteks NDN.

REFERENSI

[1] Cisco, "Cisco Visual Networking Index: Forecast and Trends, 2017–2022," 2019. [Online]. Available: https://twiki.cern.ch/twiki/pub/HEPIX/TechwatchNetwork/HtwNetworkDocuments/white-paper-c11-741490.pdf. [Accessed 29 Januari 2022].

[2] V. Jacobson, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. L. Braynard, "Networking named content," in *Proceedings of the 5th international conference on Emerging networking experiments and technologies*, 2009.

[3] M. N. D. Satria, F. H. Ilma, and N. R. Syambas, "Performance comparison of named data networking and IP-based networking in palapa ring network," in *2017 3rd International Conference on Wireless and Telematics (ICWT)*, Palembang, 2017.

[4] L. V. Yovita and N. R. Syambas, "Caching on Named Data Network: a Survey," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 8, pp. 4456-4466, 2018.

[5] M. Luthfianza and D. Febriawan, "Analisa Perbandingan Web Proxy Sebagai Filtering Antara Mikrotik dan Suid Berdasarkan Hit Ratio dan Byte Hit Ratio," *JUTIKOM*, vol. 1, no. 1, 2022.

[6] F. W. Bramantyo, "Analisa Unjuk Kerja Proxy Server Menggunakan Algoritma LRU, LFU, dan GDSF," 21 April 2016. [Online]. Available: https://repository.usd.ac.id/9905/2/115314087_full.pdf. [Accessed 11 Juni 2022].

[7] B. Parli, "bparli.medium.com," 7 April 2020. [Online]. Available: https://bparli.medium.com/enhancing-least-frequently-used-caches-with-dynamic-aging-64dc973d5857. [Accessed 21 Januari 2020].

[8] A. Afanasyev, I. Moiseenko, and L. Zhang, "ndnSIM: NDN simulator for NS-3," University of California, Los Angeles, Tech. Rep. NDN-0005, 2012.

[9] S. Mastorakis, A. Afanasyev, I. Moiseenko, and L. Zhang, "ndnSIM 2.0: A new version of the NDN simulator for NS-3," University of California, Los Angeles, Tech. Rep. NDN-0028, 2016.

[10] H. Dai, J. Lu, Y. Wang, and B. Liu, "On Pending Interest Table in Named Data Networking", in *Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication (SIGCOMM '18)*, Budapest, Hungary, pp. 369-383, Aug. 2018.