Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer e-ISSN: 2548-964X
Vol. 2, No. 9, September 2018, hlm. 2654-2660 http://j-ptiik.ub.ac.id

Analisis Kinerja *On-Path Caching* Dan *Off-Path Caching* Pada *Information-Centric Networking*

Muhamad Rizka Maulana[1], Achmad Basuki[2], Kasyful Amron[3]

Program Studi Teknik Informatika, Fakultas Ilmu Komputer, Universitas Brawijaya
Email: [1]muhamaul@gmail.com, [2]abazh@ub.ac.id, [3]kasyful@ub.ac.id

Abstrak

Peningkatan lalu lintas data di Internet yang tinggi saat ini tidak didukung oleh arsitektur Internet yang masih bersifat *host-centric*. Paradigma *Information-centric Networking* (ICN) berusaha untuk mengatasi masalah tersebut dengan model komunikasi yang bersifat *content-centric* atau *information-centric*. Beberapa konsep arsitektur baru seperti CCN, DONA, dan NetInf telah diusulkan dan umumnya memiliki tiga kesamaan konsep kerja, yaitu operasi *Publish-Subscribe*, *In-network Caching*, dan *Content-oriented Security*. *In-network caching* memainkan peran penting dalam meningkatkan kinerja jaringan dengan cara menyimpan konten di *cache* yang tersebar di jaringan. Penelitian ini bertujuan untuk membandingkan kinerja *on-path caching* dan *off-path caching* dengan menggunakan topologi jaringan Biznet pada simulator Icarus. Hasil pengujian menunjukkan bahwa dalam aspek *cache hit ratio*, mekanisme *off-path caching* memiliki CHR 12,45% lebih tinggi dibandingkan mekanisme *on-path caching*. Sementara dalam aspek *link load*, mekanisme *on-path caching* memiliki *link load* 63,14% lebih rendah dibandingkan dengan mekanisme *off-path caching*. Dan dalam aspek latensi, mekanisme *on-path caching* memiliki latensi 42,07% lebih rendah dibandingkan dengan mekanisme *off-path caching*.

Kata kunci: *Information-centric networking, in-network caching, analisis kinerja, simulator Icarus*

Abstract

The massively increasing Internet traffic is not supported by the current Internet architecture which is still based on host-centric communication. Information-centric networking (ICN) paradigm has been proposed to resolve that problem with content-centric communication. Some ICN architectures have been proposed, such as CCN, DONA, and NetInf. Those proposed architectures generally have three main concepts, which are publish-subscribe operation, in-network caching, and content-oriented security. In-network caching plays a very important role in ICN paradigm. It can improve network performance by saving content in caches that are spread in the network. This research aims to compare on-path caching and off-path caching strategies using Biznet topology on Icarus simulator. The result shows that off-path caching has 12.45% higher cache hit ratio than on-path caching, and on-path caching has 63.14% lower link load and 42,07% *lower latency than off-path caching. In addition to that, it is worth noted that bigger cache capacity and α value results in higher cache hit ratio, lower link load, and lower latency.*

Keywords: *Information-centric networking, in-network caching, performance analysis, Icarus simulator*

1. PENDAHULUAN

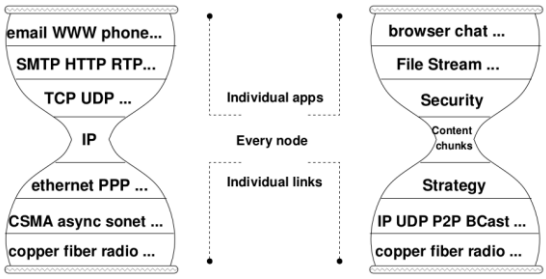
Teknologi Internet yang berkembang dengan pesat sejak tahun 1960-an telah menyebabkan peningkatan jumlah konten dan lalu lintas data yang sangat signifikan. Berdasarkan prediksi Cisco Visual Networking Index (2016), pada tahun 2020 lalu lintas IP secara global akan mencapai 2,3 ZB per tahunnya, atau 194 EB per bulan. Lalu lintas data yang sangat besar tersebut tidak didukung oleh arsitektur komunikasi Internet saat ini yang masih bersifat *host-centric* dan dirancang hanya untuk berbagi sumber daya. Paradigma *Information-Centric Networking* (ICN) diusulkan untuk mengatasi masalah tersebut dengan merancang ulang arsitektur Internet yang sebelumnya *host-centric* menjadi *information-centric* atau *content-centric*. Beberapa konsep arsitektur baru seperti CCN, DONA, dan NetInf telah diusulkan dan umumnya memiliki tiga

kesamaan konsep kerja, yaitu operasi *Publish-Subscribe*, *In-network Caching*, dan *Content-oriented Security*. Secara umum, mekanisme *in-network caching* dapat dibedakan menjadi dua jenis, yaitu *on-path caching* dan *off-path caching*. *In-network caching* memainkan peran penting dalam meningkatkan kinerja jaringan dengan cara menyimpan konten di *cache* yang tersebar di jaringan. Saat metode *caching* diterapkan di dalam jaringan, akan terjadi penurunan lalu lintas data dan latensi karena konten akan tersebar. Namun, menurut Wang (2015), metode *caching* yang sudah dipakai pada arsitektur Internet saat ini tidak banyak membantu dalam konteks arsitektur ICN karena tidak tersebar di dalam jaringan. Perbedaan arsitektur ICN dan Internet saat ini mengharuskan metode *caching* diteliti kembali agar dapat bekerja dengan baik pada arsitektur ICN (Rossi dan Rossini, 2011).

Penelitian ini bertujuan untuk menganalisis dan membandingkan kinerja mekanisme *on-path* dan *off-path caching*, khususnya dalam aspek *cache hit ratio*, *link load*, dan latensi. Pengujian kinerja dilakukan melalui sebuah simulasi pada simulator Icarus menggunakan topologi jaringan salah satu ISP di Indonesia, Biznet

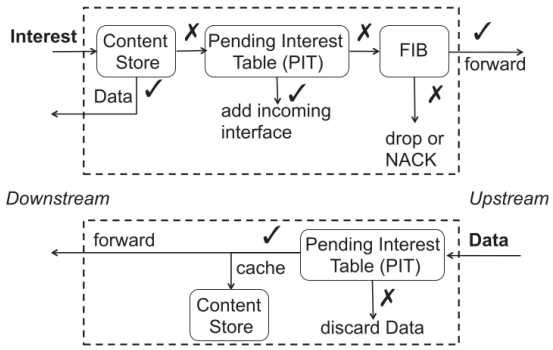
2. *INFORMATION-CENTRIC NETWORKING*

Information-centric Networking (ICN) adalah paradigma baru dalam arsitektur Internet yang diusulkan oleh Van Jacobson (2009) untuk menggantikan arsitektur Internet saat ini. Gambar 1 adalah perbandingan layer pada Internet dan ICN. Pada gambar tersebut, komunikasi pada Internet saat ini yang berbasis IP yang bersifat *host-centric* diganti dengan berbasis *content chunks* yang bersifat *content-centric*.



Gambar 1. Perbandingan layer Internet dan ICN
Sumber: Van Jacobson et al. (2009)

Arsitektur ICN memiliki dua jenis paket, yaitu Interest dan Data. Paket Interest digunakan untuk meminta suatu konten dengan cara mengirimkannya secara *broadcast* pada jaringan. Sedangkan paket Data adalah balasan yang berisi konten yang diminta. Pada penulisan penelitian ini, penggunaan istilah paket Interest dapat dipertukarkan dengan paket *request* atau permintaan, dan paket Data dapat dipertukarkan dengan paket *content* atau konten.



Gambar 2. Alur pemrosesan paket ICN
Sumber: Yi, et al. (2013)

Selain itu, terdapat tiga struktur utama, yaitu *Content Store*, *Pending Interest Table* (PIT), dan *Forwarding Information Base* (FIB), seperti pada Gambar 2. *Content Store* ialah tempat penyimpanan konten sementara, PIT berfungsi untuk mencatat setiap paket Interest yang masuk, sedangkan FIB berfungsi untuk meneruskan paket Interest ke *node* lain. Saat suatu paket Interest masuk ke suatu *node*, maka *node* tersebut pertama kali akan memeriksa apakah ia memiliki konten yang diminta di *Content Store*. Apabila ada, maka konten tersebut akan diteruskan ke *face* asal dari paket Interest tersebut. Apabila tidak ditemukan, maka paket Interest akan diperiksa di PIT.

Jika di PIT sudah ada paket Interest untuk konten yang sama, maka *face* asal paket Interest tersebut akan ditambahkan pada PIT dan paket Interest akan dihapus karena sudah ada paket Interest untuk konten yang sama. Jika di PIT tidak ada catatan paket Interest untuk konten yang sama, maka paket Interest akan diperiksa di FIB apakah memungkinkan untuk diteruskan ke *node* lain berdasarkan tabel *forwarding* di FIB. *Prefix* paket Interest akan dicocokkan dengan *prefix* di FIB dan bila cocok akan diteruskan ke *node* lain atau langsung ke *content source* lalu *face* asal Interest akan ditambahkan di PIT. Bila *prefix* tidak ditemukan di FIB, maka paket Interest akan dihapus.

Saat suatu *node* menerima paket Data, *node* tersebut akan memeriksa apakah ia memiliki catatan untuk paket Data tersebut di PIT

miliknya. Jika cocok, *node* akan menyimpan Data tersebut di *Content Store* dan meneruskannya ke *face* yang tercantum di PIT lalu menghapus *face* tersebut setelah diteruskan. Bila tidak ada paket Interest untuk Data tersebut di dalam PIT, maka paket Data akan dihapus.

3. *IN-NETWORK CACHING*

In-network caching bertujuan untuk menurunkan lalu lintas data dan latensi pengiriman dengan cara menempatkan *cache* di dalam jaringan. Apabila suatu konten ditemukan pada suatu *cache*, maka peristiwa tersebut dinamakan *cache hit*. Sebaliknya, jika konten tidak ditemukan, dinamakan *cache miss*. Secara umum, terdapat dua mekanisme yang saling melengkapi dalam *in-network caching*, yaitu *content placement* dan *content replacement*. *Content placement* mengatur bagaimana cara menyimpan konten dalam *cache* tersebar di jaringan, sedangkan *content replacement* mengatur penghapusan konten saat *cache* penuh. Strategi yang termasuk *content replacement* adalah LRU, LFU, dan FIFO. Mekanisme *content placement* masih dapat dikategorikan menjadi dua, yaitu *on-path caching* dan *off-path caching*.

Metrik yang dapat digunakan untuk mengukur kinerja *in-network caching* antara lain:

- *Cache Hit Ratio* (CHR) : Perbandingan jumlah permintaan yang berhasil dilayani (*cache hit*) oleh *cache* dengan total permintaan yang dikirimkan. Persamaan 1 adalah persamaan untuk menghitung CHR.

$$CHR = \frac{n(hit)}{n(total)} \quad (1)$$

- Latensi : Waktu yang dibutuhkan suatu *node* untuk mendapatkan sebuah konten sejak node tersebut mengirimkan permintaan sampai mendapatkan konten.
- *Link Load* : Beban kerja yang dialami suatu *link* karena lalu lintas data yang terjadi di dalamnya. Semakin tinggi *link load* berarti semakin tinggi lalu lintas data yang melalui jaringan.

3.1. *On-path Caching*

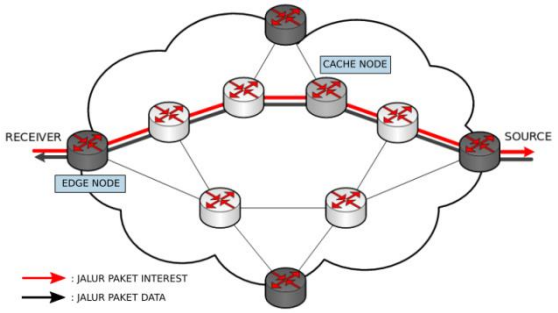
Dalam mekanisme *on-path caching*, konten akan disimpan di jalur atau *cache* yang dilalui oleh konten. Beberapa strategi yang termasuk *on-path caching* antara lain:

- **Leave Copy Everywhere (LCE):** Dalam strategi LCE, konten akan disimpan di seluruh *cache* yang dilalui oleh konten.
- **Random:** Strategi Random akan menyimpan suatu konten pada satu *cache* yang dipilih secara acak di jalur pengiriman konten.
- **ProbCache:** Strategi ProbCache diusulkan oleh Psaras, et al. (2012). Strategi ini akan menyimpan konten pada satu *cache* terbaik di sepanjang jalur pengiriman berdasarkan persamaan probabilitas. Tujuan dari ProbCache adalah mengurangi *cache reduncancy* sehingga konten yang disimpan di jaringan *cache* lebih bervariasi.

3.2. *Off-path Caching*

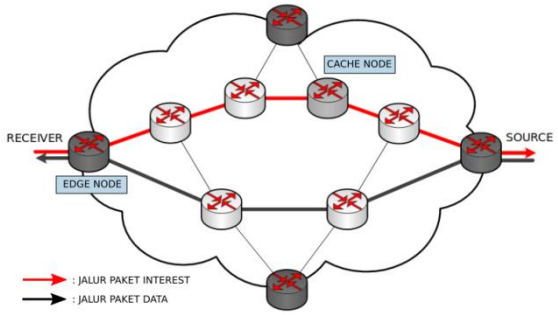
Dalam mekanisme *off-path caching*, konten akan disimpan pada *cache* yang dipilih berdasarkan aturan yang sudah ditentukan sebelumnya, misalnya dengan metode Hash-routing (Saino, Psaras, & Pavlou, 2013). Dengan Hash-routing, sebuah paket permintaan atau konten akan diteruskan ke suatu *cache* berdasarkan fungsi *hash.* Jenis-jenis strategi *off-path caching* yang berbasis Hash-routing (HR) sebagai berikut:

- **Hash-routing Symmetry:** HR Symmetry adalah strategi *off-path caching* di mana jalur pengiriman yang dilalui oleh konten sama dengan jalur yang dilalui oleh paket permintaan. Saat menerima paket Permintaan dari *receiver*, *edge node* akan melakukan komputasi fungsi *hash* untuk memetakan *cache node* lalu meneruskan paket Permintaan ke *cache node* tujuan. Apabila memiliki konten yang diminta (*cache hit*), *cache node* tujuan akan mengirim paket Data ke peminta melalui jalur yang dilalui oleh paket Permintaan.
  Apabila tidak memiliki konten yang diminta (*cache miss*), *cache node* tujuan akan meneruskan paket Permintaan ke *content source*. *Content source* akan mengirimkan paket Data ke *receiver* melalui jalur yang dilewati paket Permintaan dan *cache node* menyimpan replika konten, seperti pada Gambar 3.



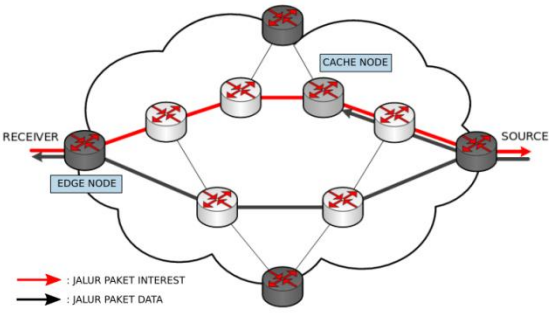
Gambar 3. *Cache miss* pada HR Symmetry

- **Hash-routing Asymmetry:** Perbedaan HR Symmetry dan HR Asymmetry hanya terletak pada saat *cache miss*, di mana konten akan dikirim melalui jalur terpendek menuju *receiver*, tanpa ada proses penyimpanan di *cache*, seperti pada Gambar 4.



Gambar 4. *Cache miss* pada HR Asymmetry

- **Hash-routing Multicast:** HR Multicast merupakan kombinasi dari HR Symmetry dan HR Asymmetry. Saat terjadi *cache miss*, HR Multicast dapat mengirimkan konten melaui jalur terpendek ke *receiver* sekaligus menyimpannya di *cache*, seperti pada Gambar 5.



Gambar 5. *Cache miss* pada HR Multicast

4. POPULARITAS KONTEN

Popularitas konten adalah jumlah permintaan untuk suatu konten, yang mana semakin tinggi jumlah permintaan maka konten tersebut semakin populer. Studi tentang popularitas objek di bidang ilmu komputer pernah dilakukan oleh Breslau, et al. (1999) yang menyatakan bahwa permintaan halaman web mengikuti distribusi *Zipf-like* atau *power law*. Persamaan 3 adalah persamaan peluang untuk permintaan suatu konten dengan peringkat popularitas ke- i dalam distribusi *Zipf-like*,

$$P(i) = \frac{C}{i^{\alpha}} \qquad (3)$$

di mana C adalah $\left(\sum_{i=1}^{n} \frac{1}{i^{\alpha}}\right)^{-1}$ dari n sejumlah konten dan α adalah parameter *skewness* dan mengindikasikan derajat konsentrasi permintaan (Laoutaris, Syntila, & Stavrakakis, 2004).

5. METODOLOGI PENGUJIAN

Pengujian dilakukan menggunakan simulator Icarus dengan lingkungan simulasi yang disesuaikan semirip mungkin dengan kondisi riil Internet. Sebelum memulai simulasi, sebuah berkas Graph Modelling Language (GML) yang berisi topologi Biznet harus dibuat terlebih dahulu. Berkas tersebut lalu diintegrasikan dengan Icarus agar dapat disimulasikan. Proses integrasi tersebut juga berfungsi untuk memberi tugas pada node di dalam topologi untuk berfungsi sebagai *content source*, *cache router*, *receiver*, atau *router* biasa, seperti pada topologi pengujian di Gambar 6.

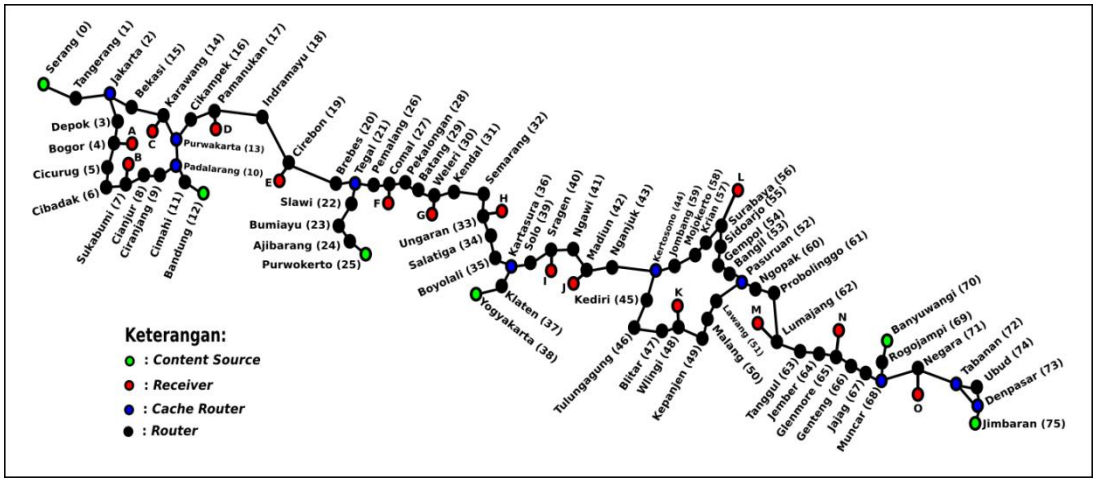
Setelah proses integrasi selesai, parameter-parameter pengujian di dalam Icarus perlu dikonfigurasi. Parameter-parameter tersebut dapat dilihat pada Tabel 1. Proses simulasi pada Icarus akan berjalan sebanyak jumlah kombinasi dari strategi yang diuji, *content placement*, nilai α, dan kapasitas *cache*.

6. HASIL DAN PEMBAHASAN

Hasil pengujian menunjukkan bahwa setiap kenaikan nilai α diikuti oleh semakin baiknya kinerja *in-network caching* dan nilai α 0,9 memiliki pengaruh paling baik terhadap CHR, *link load*, dan latensi (semakin tinggi CHR serta semakin rendah *link load* dan latensi) dibandingkan dengan nilai α 0,7 dan 0,8. Nilai α 0,9 akan digunakan sebagai acuan untuk menganalisis dan membandingkan *on-path caching* dan *off-path caching*.

6.1. *Cache Hit Ratio*

Berdasarkan hasil pengujian CHR *on-path caching* dan *off-path caching*, terjadi peningkatan CHR seiring dengan meningkatnya kapasitas *cache*. Saat kapasitas *cache* 1%,



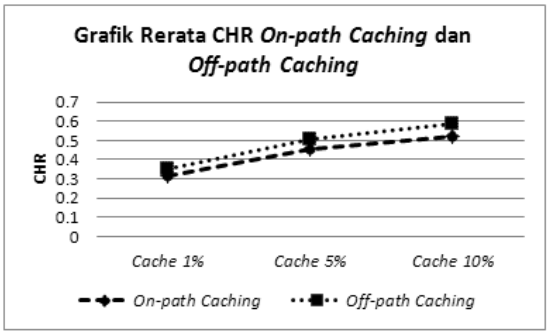
Gambar 6. Topologi pengujian

rerata CHR mekanisme *on-path caching* adalah 0,313, sementara kapasitas cache 5%, CHR naik 45,04%. Sementara saat kapasitas cache 10%, rerata CHR *on-path caching* naik 14,09% menjadi 0,518. Sedangkan untuk *off-path caching*, saat kapasitas *cache* 1%, rerata CHR adalah 0,352, dan saat kapasitas *cache* naik menjadi 5%, CHR *off-path caching* naik 44,22%. Dan saat kapasitas cache 10%, rerata CHR *on-path caching* naik 15,29%. Meningkatnya kapasitas *cache* berarti semakin banyak konten yang dapat ditampung, sehingga peluang terjadinya *cache hit* semakin besar.

Tabel 1. Parameter Pengujian

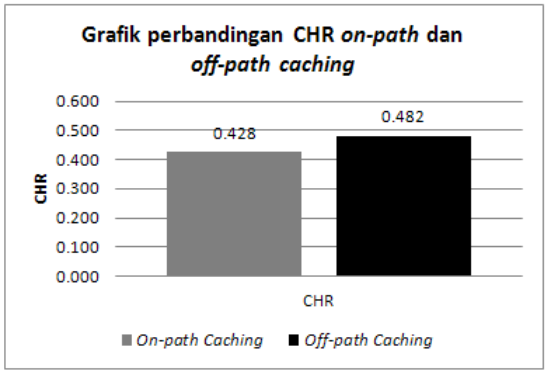
Parameter	Nilai
Strategi yang diuji	LCE, Random, ProbCache, Hash-routing Symmetry, Hash-routing Asymmetry, Hash-routing Multicast
Content replacement	LRU
Nilai α	0,7, 0,8, 0,9
Kapasitas *Cache*	1%, 5%, dan 10% dari jumlah keseluruhan konten
Kolektor Data	*Cache Hit Ratio*, Latensi, *Link Load*
Jumlah Konten	252255 Konten
Request Rate	15 permintaan per detik
Warm-up Request	252255 Permintaan
Measured Request	504510 Permintaan

Gambar 7 menunjukkan grafik kenaikan rerata CHR pada *on-path caching* dan *off-path caching*. Strategi *in-network caching* yang memiliki rerata CHR tertinggi adalah HR Multicast dan HR Symmetry.



Gambar 7. Grafik rerata CHR *on-path caching* dan *off-path caching*

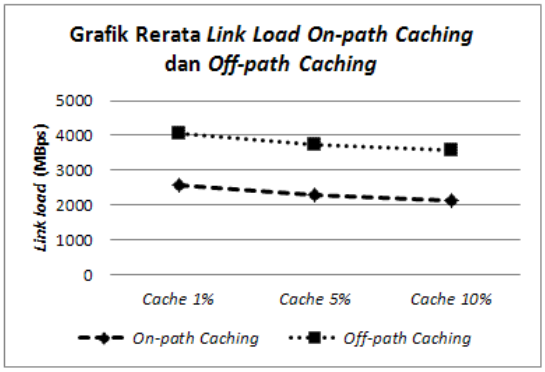
Untuk mengetahui hasil perbandingan secara menyeluruh, maka rerata tiap hasil pengujian dikalkulasikan lagi reratanya agar dapat mewakili kinerja *on-path caching* dan *off-path caching*. Berdasarkan hasil rerata tersebut, CHR *off-path caching* 12,45% lebih tinggi dibandingkan dengan CHR *on-path caching*. Hal tersebut cukup beralasan mengingat mekanisme tersebut menyimpan konten di *cache* yang sudah ditentukan saat terjadi *cache miss*, sehingga memperbesar peluang terjadinya *cache hit* pada permintaan-permintaan selanjutnya. Sedangkan *on-path caching* menyimpan konten di cache yang dilalui oleh paket konten. Gambar 7 adalah grafik perbandingan CHR *on-path caching* dan *off-path caching*. Rerata CHR 0,428 untuk *on-path caching* dan 0,482 untuk *off-path caching* menunjukkan bahwa rata-rata 42,8% permintaan konten berhasil dilayani oleh *cache* pada *on-path caching* dan 48,2% permintaan konten berhasil dilayani oleh *cache* pada *off-path caching*. Gambar 8 adalah grafik perbandingan CHR *on-path caching* dan *off-path caching*.



Gambar 8. Grafik perbandingan CHR *on-path* dan *off-path caching*

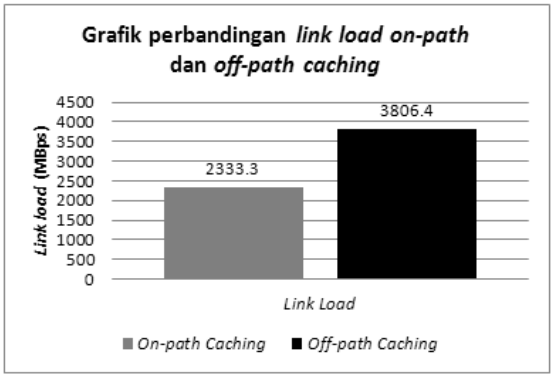
6.2. *Link Load*

Pada hasil pengujian *link load*, rerata *link load on-path caching* dan *off-path caching* mengalami penurunan seiring dengan meningkatnya kapasitas *cache*. Pada saat kapasitas *cache* 1%, rerata *link load on-path caching* adalah 2584,7 MBps. Saat kapasitas *cache* naik menjadi 5%, rerata *link load* turun 11,87%. Dan saat kapasitas *cache* naik menjadi 10%, rerata *link load on-path caching* menurun 6,14%. Sementara untuk *off-path caching*, saat kapasitas *cache* 1%, rerata *link load* adalah 4067 MBps. Saat kapasitas *cache* naik menjadi 5%, rerata *link load* turun 7,66%. Dan saat kapasitas *cache* naik menjadi 10%, rerata *link load off-path caching* menurun lagi 4,21%. Gambar 9 adalah grafik rerata *link load* pada *on-path caching* dan *off-path caching*.



Gambar 9. Grafik rerata *link load on-path caching* dan *off-path caching*

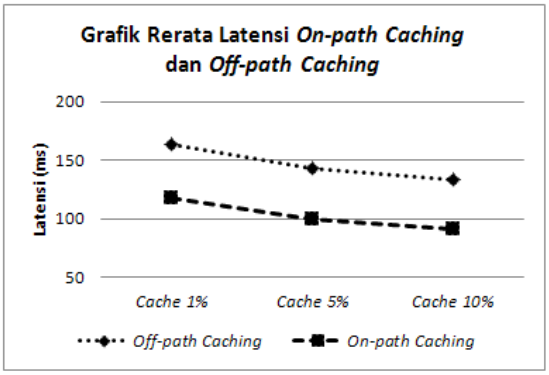
Untuk mengetahui hasil perbandingan secara menyeluruh, maka rerata tiap hasil pengujian juga dikalkulasikan lagi reratanya agar dapat mewakili kinerja *on-path caching* dan *off-path caching*. Berdasarkan rerata tersebut, *link load* mekanisme *on-path caching* 63,14% lebih rendah dibandingkan dengan *link load* mekanisme *off-path caching*. Hal tersebut karena pada mekanisme *on-path caching* hanya terjadi satu kali pengiriman konten melewati jalur yang simetris bila terjadi *cache hit* atau *cache miss*, sedangkan mekanisme *off-path caching* harus mengirim dua kali saat terjadi *cache miss* (ke *node receiver* dan ke *cache* untuk menyimpan konten) yang menyebabkan bertambahnya lalu lintas data dalam jaringan. Selain itu, permintaan akan diteruskan ke *cache* hasil fungsi *hash* yang letaknya belum tentu dekat dengan *receiver*. Strategi *in-network caching* yang memiliki rerata *link load* terendah adalah ProbCache. Gambar 10 adalah grafik perbandingan *link load on-path caching* dan *off-path caching*.



Gambar 10. Grafik perbandingan *link load on-path* dan *off-path caching*

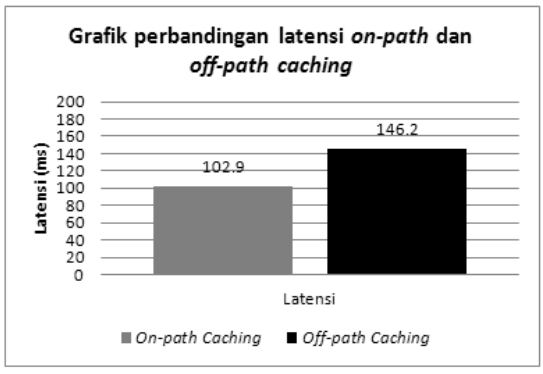
6.3. Latensi

Pada pengujian latensi *on-path caching* dan *off-path caching*, hasil pengujian menunjukkan penurunan latensi seiring dengan meningkatnya kapasitas *cache*. Saat kapasitas *cache* 1%, rerata latensi mekanisme *on-path caching* adalah 117,6 ms. Sementara saat kapasitas *cache* naik menjadi 5%, rerata latensi turun 15,27%. Sedangkan saat kapasitas *cache* 10%, rerata latensi *on-path caching* turun 8,26%. Dan untuk *off-path caching*, saat kapasitas *cache* 1%, rerata latensi mekanisme *off-path caching* adalah 162,9 ms. Sementara saat kapasitas *cache* naik menjadi 5%, rerata latensi turun 12,38%. Sedangkan saat kapasitas *cache* 10%, rerata latensi mekanisme *off-path caching* turun 6,84%. Gambar 11 adalah grafik rerata latensi pada *on-path caching* dan *off-path caching*.



Gambar 11. Grafik rerata latensi *on-path caching* dan *off-path caching*

Rerata tiap hasil pengujian juga dikalkulasikan lagi reratanya agar dapat mewakili kinerja *on-path caching* dan *off-path caching*. Berdasarkan rerata tersebut, mekanisme *on-path caching* memiliki latensi 42,07% lebih rendah dibandingkan dengan mekanisme *off-path caching*. Hal ini karena mekanisme *off-path caching* menyimpan konten di *cache* yang seringkali tidak dekat (karena sudah ditentukan) dengan *receiver* sehingga membutuhkan waktu yang lebih lama dibandingkan dengan *on-path caching* yang menyimpan konten pada *cache* yang dilalui oleh konten. Selain itu, apabila terjadi *cache miss*, mekanisme *off-path caching* juga akan meneruskan paket *request* ke *content source*, sehingga memperpanjang waktu yang dibutuhkan dalam satu kali sesi permintaan. Strategi *in-network caching* yang memiliki rerata latensi terendah adalah ProbCache. Gambar 12 adalah grafik perbandingan latensi mekanisme *on-path caching* dan *off-path caching*.



Gambar 12. Grafik perbandingan latensi *on-path* dan *off-path caching*

7. KESIMPULAN

Hasil pengujian menunjukkan bahwa dalam aspek *cache hit ratio*, mekanisme *off-path caching* memiliki CHR 12,45% lebih tinggi dibandingkan mekanisme *on-path caching*. Sementara dalam aspek *link load*, mekanisme *on-path caching* memiliki *link load* 63,14% lebih rendah dibandingkan dengan mekanisme *off-path caching*. Dan dalam aspek latensi, mekanisme *on-path caching* memiliki latensi 42,07% lebih rendah dibandingkan dengan mekanisme *off-path caching*. Strategi *in-network caching* yang cocok pada topologi Biznet adalah ProbCache yang memiliki latensi dan *link load* paling rendah serta CHR cukup tinggi yang hampir sama dengan mekanisme *off-path caching.*

8. DAFTAR PUSTAKA

Breslau, L., Cao, P., Fan, L., Phillips, G., & Shenker, S. (1999). Web Caching and Zipf-like Distribution: Evidence and Implications. *INFOCOM '99*. IEEE.

Cisco Visual Networking Index: Forecast and Methodology, 2015-2020.*White paper* (2016) Cisco.

Jacobson, V., Smetters, D. K., Thornton, J. D., Plass, M. F., Briggs, N. H., & Braynard, R. L. (2009). Networking Named Content. *CoNEXT '09* (pp. 1-12). New York, NY: ACM.

Laoutaris, N., Syntia, S., & Stavrakakis, I. (2004). Meta Algorithms for Hierarchical Web Caches. *IEEE Conference on Performance, Computing, and Communication 2004.* Phoenix, AZ: IEEE.

Psaras, I., Chai, W. K., & Pavlou, G. (2012). Probabilistic In-Network Caching for Information Centric Networks. *ACM ICN '12* (pp. 55–60). Helsinki: ACM.

Saino, L., Psaras, I., & Pavlou, G. (2013). Hash-routing Schemes for Information Centric Networking. *ACM SIGCOMM (ICN'13).* Hong Kong: ACM.

Saino, L., Psaras, I., & Pavlou, G. (2014). Icarus: a Caching Simulator for Information Centric Networking (ICN). *SIMUTOOLS'14.* Lisbon: ACM.

Yi, C., Afanasyev, A., Moiseenko, I., Wang, L., Zhang, B., & Zhang, L. (2013). A Case for Stateful Forwarding Plane. *Computer Communications*, 779-791.

Wang, L. (2015). *Content, Topology and Cooperation in In-network Caching.* Helsinki: University of Helsinki.