

令和5年度 春学期 卒業研究3 (BI)
学士論文

題目 低信頼なエッジキャッシュを
用いた高信頼キャッシュ法

指導教員 上山憲昭 教授

立命館大学 情報理工学部 セキュリティ・ネットワークコース

学籍番号 2600200469-0

吉田開

令和6年1月31日

概要

Mobile Edge Cache (MEC) は、人気のあるコンテンツをユーザの近くの ES にキャッシュをすることで、検索待ち時間、ネットワークの混雑、リモートコンテンツプロバイダへのリクエスト数を削減することが期待されている。一般的に MEC では、コストの低い低信頼な ES が用いられており、障害による不稼働率の増加が課題となっている。低信頼な ES を想定した研究では、erasure coding を用いて、キャッシュコンテンツの可用性を高めているが、コンテンツを事前に ES へ配置することを想定している。そこで本稿では、低信頼な ES において LRU を用いた置換制御を想定した ES の不稼働を考慮した上で、ES からのコンテンツ取得可能性を向上させる erasure coding を用いた ES 挿入法を提案する。また、目標ヒット率 $\hat{h}_m$ と実際のヒット率 h_m の比較により、キャッシュ挿入確率 f_m を適切に設定できていることを確認した。また、全コンテンツ m の平均取得成功率 (ES からのチャンク取得のみでコンテンツを復元できた割合) を提案方式を用いてキャッシュ挿入確率 f_m を設定した場合と、キャッシュミス時に常にコンテンツをキャッシュした場合について数値評価により比較をした。その結果、提案方式は単にコンテンツを ES にキャッシュする場合と比較して平均取得成功率を大きく改善することを確認した。

目次

概要	1
第 1 章 序論	3
1.1 研究の背景	3
1.2 研究の目的	3
第 2 章 関連研究	5
第 3 章 既存方式	7
3.1 低信頼なキャッシュ資源を持つ分散協調キャッシングシステム	7
3.2 DEC システムにおけるキャッシュ配置問題 (CP)	9
第 4 章 提案方式	12
4.1 キャッシュ挿入確率 f_m 算出	12
4.2 最適化目標	12
4.3 アルゴリズム	13
第 5 章 性能評価	14
5.1 評価条件	14
5.2 目標ヒット率と実際のヒット率の比較	15
5.3 各 ES の稼働率 r に対する平均取得成功率の変化	16
5.4 異なるリクエストレートでの平均取得成功率の変化	17
第 6 章 まとめ	18
謝辞	19
学会発表リスト	21

第1章 序論

1.1 研究の背景

5G では, スマートフォン等の移動端末で動画を視聴する機会が増加しており, モバイルネットワークの負荷が急増し, 通信品質の劣化やネットワークコストの増大が課題となっている. そこで 5G では, 無線の基地局にキャッシュサーバ (ES: edge Server) を設置し, ユーザに近い ES からコンテンツを配信する Mobile Edge Cache (MEC) が用いられている. MEC は, 近接して配置された ES で協調的なエッジキャッシュネットワークを形成し [2], 人気のあるコンテンツをユーザの近くの ES にキャッシュをすることで, ユーザが要求したコンテンツを取得する際の遅延を大幅に削減することが可能である. また, MEC はバックホールリンクを介したデータ伝送を回避するため, バックホールトラフィックを削減することが可能である. MEC については, 様々な目的や用途で数多くの研究が行われているが, これらの研究のほとんどは, キャッシュの障害を考慮せず, 高信頼な ES を前提としている. しかし, 高信頼なエッジキャッシュ資源は限られており [3], 非常に高価である [4]. 実際の MEC では, ES の設置数は膨大であるため, コストの低い低信頼な ES が用いられ, 障害による不稼働率の増加が課題となっている. また, 従来のキャッシュ制御では, 全ての ES が常時稼働し, 想定したキャッシュ容量が常に利用できることを想定しているが, 低信頼な ES を想定したキャッシュ制御が必要である.

また近年, 高いストレージ効率と信頼性を提供するために, 複数のサーバに冗長性を持たせた分散データストレージとして erasure coding がストレージシステムで広く利用されている [5] [6] [7]. Maximum distance separable (MDS) code 等の Reed-Solomon (RS) 符号 [9] は, ストレージ容量効率の点で最適であり, ストレージシステムで広く使用されている [5] [6] [10] [11]. 本稿では, ES の信頼性の低さに対処するために一般的なファイルを erasure coding を用いて冗長性を持たせキャッシュしコンテンツの可用性を高める.

1.2 研究の目的

低信頼な ES を想定した既存研究 [12] では, エッジ資源の信頼性の低さに対処するために, erasure coding を用いて ES の可用性を高める方式を提案しているが, コンテンツを ES 上に事前配置することを想定している. しかし, 実際の ES では LRU (Least Recently Used) 等を用いた置換による動的制御が一般的である.

そこで本稿では, 低信頼な ES において LRU を用いた置換制御を想定した ES の不稼働を考慮した上で, ES からのコンテンツ取得可能性を向上させる erasure coding を用いた ES 挿入法を提案する. 数値評価により, 提案方式を用いてコンテンツをキャッシュした場

合と、キャッシュミス時に常にコンテンツをキャッシュした場合で全コンテンツの平均取得成功率（ES からのチャンク取得のみでコンテンツを復元できた割合）の比較を行う。

第2章 関連研究

MECでは、近接して配置されたESで協調的なエッジキャッシュネットワークを形成し、人気のあるコンテンツをユーザの近くのESにキャッシュすることで、検索待ち時間やネットワークトラフィックを削減することが可能である。[14]でRamaswamyらは、効果的で効率的な協調エッジキャッシュネットワークを生成するために与えられたエッジキャッシュネットワークのサーバを指定された数の協調キャッシュグループに分割する問題を研究した。[2]ではRamaswamyらが、動的コンテンツをキャッシュするための協調的なエッジネットワークを設計する際の課題を研究し、大規模なエッジキャッシュネットワークにおける協調のための枠組みとして新たにキャッシュクラウドを提案した。しかし、MECに関する上記の全ての研究では、キャッシュ障害を考慮せず、高信頼なESを前提としている。

一方、信頼性を向上させ、冗長性を備えた分散データストレージシステムを使用することが注目されている[5]。レプリケーションは、冗長性を備えた分散データストレージの最も単純な形式である[8]。レプリケーションでは、オリジナルデータの同一のコピーをレプリカとして別々のストレージにコピーをし、冗長性を提供する。あるノードで障害が発生したり、データが失われた場合に、別のノードに複製されたコピーを使用して情報を復元することが可能である。しかし、レプリケーションでは元のデータの完全なコピーを各ノードに保存する必要があるため、より多くのストレージ容量が必要となる。そこで、信頼性とストレージ効率を向上させる erasure coding がストレージシステムで広く利用されている。erasure coding では、データを断片に分解し、消去符号と呼ばれる追加の断片を生成する。生成した断片は複数のノードに分散され、一部の断片が破損、消失したとしても元のデータを復元することを可能にする。erasure coding をストレージシステムで用いることで、障害により一定数のストレージノードが利用できない場合や、データの損失が発生した場合に元のデータを再構築できるためシステムの耐障害性やデータの信頼性、耐久性を向上させることが可能となる。また、データを完全に複製することなく冗長性を実現するため、ストレージ効率を向上することが可能となる。

そこで、低信頼なESに対処するために、erasure coding を活用してキャッシュするファイルの信頼性を高める方法が考えられるが、信頼性の低いESから取得できるデータサイズを最大化することは以下の理由により困難である。

- ESのキャッシュ容量には限りがあるため、キャッシュされるファイルの信頼性とESがキャッシュできるファイル数の間にはトレードオフがある。
- 異なるファイルには異なる人気があるため、より人気のあるファイルの存続は、人気のないファイルよりもユーザーに利益をもたらす可能性がある。
- ESの故障確率は異なり、各ファイルの信頼性はチャンクの場所によって変わる。

このような課題を解決するために, 容量制約の下で各ファイルに追加される冗長性, 各ファイルのチャンク数やチャンクの場所を適切に設定する必要がある. そこで, 信頼性の低いリソースを持つ分散協調エッジキャッシングシステムのためのキャッシュ配置アルゴリズムが提案されている [12]. 次節では, 本方式の概要を述べる.

第3章 既存方式

3.1 低信頼なキャッシュ資源を持つ分散協調キャッシングシステム

[12] で Li らは、低信頼なキャッシュ資源を持つ分散協調キャッシングシステム (DEC) を提案した。DEC システムで利用する記号と定義を以下の表 3.1 で示す。DEC では各ファイルの長さは全て M ビットと仮定し [1], 各ファイルを M/k の k 個のチャンクに分割し, RS 符号などの MDS 符号 [2] によって, k 個のチャンクから x 個の冗長チャンクを生成し, これらを N 個の ES に分散配置する。そして ES から任意の k 個以上のチャンクが取得できれば, 元のファイルを復元可能である。ユーザの要求したコンテンツのチャンクが利用可能な ES 上に k 個以上キャッシュされていない場合は, コアネットワークを介し, コンテンツプロバイダーから取得する必要がある。各ファイルのチャンクに冗長性を追加し, 異なる ES にキャッシュすることで, ES が故障しやすい環境においても k 個以上のチャンクを ES から取得できる可能性が高まるため, ファイル取得の待ち時間, コアネットワークの輻輳, リモートサーバの負荷を軽減する。[12] では ES から取得できるデータの総量が最大化するよう, 各コンテンツの静的な要求頻度に対し, 各 ES に配置するチャンクを最適設計している。

表 3.1: DEC システムに用いる記号と定義

記号	定義
$[T] = \{1, \dots, T\}$	タイムスロット時間
$\mathcal{I} = \{1, \dots, I\}$	人気ファイルの集合
h_i	ファイル i の要求レート
k	ファイル回復に必要なチャンクの下限值
$\mathcal{N} = \{1, \dots, N\}$	ES の集合
p_n	ES n の信頼性パラメタ
$f_{n,t}$	サーバ n がタイムスロット t で使用できる確率
$a_{i,t}$	タイムスロット t の間のファイル i の可用性
a_t	ファイル i の時間累積可用性
$x_{i,n}$	サーバ n にキャッシュされたファイル i のチャンク数
$\mathbf{x}_i$	$n \in N$ に対する $x_{i,n}$ の集合
$\mathbf{x}$	$i \in I, n \in N$ に対する $x_{i,n}$ の集合
$y_i = \sum_{n \in N} x_{i,n}$	ファイル i のチャンク数

3.2 DEC システムにおけるキャッシュ配置問題 (CP)

[12] では, DEC システムにおけるキャッシュ配置問題 (CP) を離散最適化問題として定式化し, *HomoCP* (Homogeneous Version of CP) と呼ばれる CP のアルゴリズムを設計している. $x_{i,n} \in \{0, 1\}$ を, ファイル i のチャンクが ES n に配置されているかどうかを表す変数とする. *HomoCP* では, ES は同じ容量と障害確率を持つため, $s = s_n, p = p_n$ である. ES にキャッシュされたファイル i のチャンクの総数を y_i で表す. すなわち, $y_i = \sum_{n \in \mathcal{N}} x_{i,n}$ で表される. *HomoCP* では, ファイル i の可用性 (ファイル i の使用可能なチャンクによってファイル i が回復できる確率) は y_i によって決定されるため, 最適な $y_i, i \in [N]$ を見つけることに焦点を当てている. ファイル i は, ファイル i の任意の k 個以上のチャンクによって回復できるため, ファイル i の可用性は, ファイル i の k 個以上のチャンクが利用可能な確率に等しい. よって与えられた y_i に対して, タイムスロット t におけるファイル i の期待可用性は次の式で表される.

$$A_t(y_i) = \begin{cases} \sum_{j=k}^{y_i} \binom{y_i}{j} (f_{n,t})^j (1 - f_{n,t})^{y_i-j} & \text{if } y_i \geq k, \\ 0 & \text{if otherwise.} \end{cases} \quad (3.1)$$

ここで, $\binom{y_i}{j}$ は, y_i 個から j 個選ぶ組み合わせ数である.

DEC システムの下では, ES 上に k 個以上チャンクがない場合, ユーザは混雑したコアネットワークを介してデータをダウンロードしなければならないため, 時間がかかる. そこで, DEC システムの目的は, ES からダウンロードされるファイルの期待サイズを最大化することである. *HomoCP* の下で, ES からダウンロードされるデータサイズを以下に示す.

$$g(\mathbf{y}) = \sum_{i \in \mathcal{I}} h_i a_i = \sum_{i \in \mathcal{I}} h_i \sum_{t \in [T]} A_t(y_i) = \sum_{i \in \mathcal{I}} h_i A(y_i) \quad (3.2)$$

ここで $A(y_i)$ は, ファイル i の時間累積可用性である.

HomoCP は, 数学的に以下のように記述される.

$$\begin{aligned} \max_{y_i, i \in \mathcal{I}} \sum_{i \in \mathcal{I}} h_i a_i &= \sum_{i \in \mathcal{I}} h_i A(y_i) \\ \text{s.t.} \quad \sum_{i \in \mathcal{I}} y_i &\leq s \cdot N \\ y_i &\in \{0, 1, \dots, N\}, \forall i \in \mathcal{I} \end{aligned} \quad (\text{HomoCP})$$

[12] では, 動的計画法 (DP) に基づき, Algorithm for *HomoCP* (AHM) と呼ばれるアルゴリズムの設計をしている. $w(z, j)$ を P1 の最適な目的値とする.

$$\begin{aligned} \max_{y_i, i \in [j]} \sum_{i \in [j]} h_i A_{y_i} \\ \text{s.t.} \quad \sum_{i \in [j]} y_i &\leq z \\ y_i &\in \{0, 1, \dots, N\}, \forall i \in [j] \end{aligned} \quad (\text{P1})$$

$w(z, j)$ は, j 個のファイルが存在し, ES の合計容量が z チャンクである場合の *HomoCP* の最適な目的値である. また, $r(z, j)$ は, j 個のファイルが存在し, ES の合計容量が z チャンクである場合の最適な選択肢を記録する. $w(z, j)$ の定義に基づくと, *HomoCP* の最適な目的値は, $w(sN, I)$ に等しくなる. $w(z, j)$ と $r(z, j)$ の初期化 ($j = 1$ の場合) を以下に示す.

$$w(z, 1) = \begin{cases} h_1 A(z) & \text{if } k \leq z \leq N \\ 0 & \text{if otherwise.} \end{cases} \quad (3.3)$$

$$r(z, 1) = \begin{cases} z & \text{if } k \leq z \leq N \\ N.A. & \text{if otherwise.} \end{cases} \quad (3.4)$$

ファイルを回復できるのは, チャンクが k 個以上存在しキャッシュできるチャンクの最大数が N の場合であるため $r(z, 1)$ は, $z < K, z > N$ の場合, N.A. である. 各 $j \in \{2, 3, \dots, I\}, z = \{0, 1, \dots, sN\}$ に対して, $w(z, j)$ と $r(z, j)$ は次のようになる.

$$w(z, j) = \max_{y_i} \{w(z - y_i, j - 1) + h_j A(y_i)\}, \quad (3.5)$$

$$r(z, j) = \operatorname{argmax}_{y_i} \{w(z - y_i, j - 1) + h_j A(y_i)\} \quad (3.6)$$

最適な目的値である, $w(sN, I)$ を取得した後, $r(z, j), 0 \leq z \leq sN, j \in [I]$ の記録をバックトラックすることで, 最適な選択を得ることができる. *HomoCP* の DP ベースのアルゴリズムは, Algorithm1 で記述されている. 本稿では, AHM アルゴリズムをもとに, 目標チャンク数 T_m の設定を行う.

Algorithm 1 *AHM for HomoCP*

```

1: Set  $w(z, 1)$  and  $r(z, 1)$  based on (5) and (6);
2: for  $j = \{2, 3, \dots, I\}$  do
3:   for  $z = \{0, 1, \dots, sN\}$  do
4:     Set  $w(z, j)$  based on (7);
5:     Set  $r(z, j)$  based on (8);
6:   end for
7: end for
8: Set  $z = sN$ ;
9: for  $i = \{I, I - 1, \dots, 1\}$  do
10:   Set  $y_i = r(z, j)$ ;
11:   Set  $z = z - r(z, i)$ ;
12: end for
13: Return  $y_i, i \in \mathcal{I}$  as the optimal solution of HomoCP;

```

第4章 提案方式

既存研究では、コンテンツの人気度は一定と想定し ES へコンテンツの事前配置を行っていた。実際の ES では、コンテンツの人気度は動的に変化するため LRU 等を用いた置換による動的制御が一般的に行われている。そこで本節では、低信頼な ES において LRU を用いた置換制御を想定した ES の不稼働を考慮した上で、ES からのコンテンツ取得可能性を向上させる erasure coding を用いた ES 挿入法を提案する。

4.1 キャッシュ挿入確率 f_m 算出

あるエリアに存在する N 個の ES へのチャンクキャッシュを想定し、置換方式は LRU を使用する。各 ES は、同じ容量 s と同じ稼働率 r を設定する。各コンテンツ m のチャンクは、各 ES に 1 つまでキャッシュすることが可能であり、合計で sN 個のチャンクをキャッシュすることが可能である。各コンテンツ m の、 N 個の ES 上にキャッシュされるチャンク数の期待値 T_m が [12] の設計値に一致するよう、コンテンツ m のキャッシュへの挿入確率 f_m を事前に設定する。各コンテンツ m の ES 上のキャッシュ数の期待値が T_m となる、ES の目標ヒット率 $\hat{h}_m$ は次式で与えられる。

$$T_m = Nr\hat{h}_m \quad (4.1)$$

4.2 最適化目標

目標ヒット率 $\hat{h}_m$ と、実際のヒット率 h_m の差が最小となるようキャッシュ挿入確率 f_m を最適設計する。提案方式で用いるパラメタを表 4.1 に示す。

要求が多い高人気コンテンツほど目標ヒット率 $\hat{h}_m$ と実際のヒット率 h_m の差異を小さくしたいため最小化を目的とする目的関数を次式で与える。

$$\min \sum q_m (h_m - \hat{h}_m)^2 \quad (4.2)$$

ここでの実際のヒット率 h_m は、文献 [13] で提案されている、次式で与えられる近似式を用いる。

$$h_m = 1 - e^{-f_m q_m t_C} \quad (4.3)$$

ここでの t_C は、 s をキャッシュ容量として次式を解くことで得られる。

$$\sum_{m \in M} 1 - e^{-f_m q_m t_C} = s \quad (4.4)$$

表 4.1: 提案方式に用いる記号と定義

記号	定義
N	ES の個数
s	ES の容量
r	ES の稼働率
m	コンテンツ数
q_m	コンテンツ m の要求レート
T_m	各コンテンツ m のキャッシュされるチャンク数の期待値
h_m	各コンテンツ m のヒット率
$\hat{h}_m$	各コンテンツ m の目標ヒット率
f_m	各コンテンツ m のキャッシュ挿入確率
t_C	Characteristic time

4.3 アルゴリズム

以下の遺伝的アルゴリズム (GA) で本最適化問題の近似解を得る.

1. キャッシュ挿入確率 f_m として離散的な区間を考え, 全コンテンツに対して, 各コンテンツ m に対して設定する f_m の区間の値を要素としてもつベクトルを各遺伝子として考え, 初期の複数の遺伝子をランダムに生成
2. 各遺伝子の適応度を (2) 式の逆数で算出し, 適応度の高いものをトーナメント選択で選択
3. 選択した遺伝子をもとに次世代の遺伝子を生成
4. 一定の確率で, 遺伝子内の f_m に対して交叉または突然変異を適用
5. step2~step4 を G 世代まで反復
6. step5 で最後の世代まで反復し, 生成遺伝子の中で最も適応度の高い遺伝子をキャッシュ挿入確率 f_m として決定

第5章 性能評価

5.1 評価条件

性能評価で使用するパラメタを表 5.1 にまとめる.

表 5.1: 評価に用いたパラメタ

記号	値
N	20, 30
s	6
r	0.7, 0.75, 0.8, 0.85, 0.9
k	5
m	50
θ	0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0, 1.1, 1.2

コンテンツ数 m を 50, コンテンツの復元に必要なチャンク数の下限値 k を 5, 各 ES の稼働率 r を 0.7~0.9 までの 0.05 刻みで設定する. あるエリアに存在する ES の個数 N が 20 の時, 各 ES の容量 s が 6 より, キャッシュできるチャンクの個数は最大 120 個となる. パラメタ θ の Zipf 分布に従いコンテンツを要求させ LRU によってキャッシュの置換を行った. また, 遺伝的アルゴリズムのパラメタとして, 個体数 200, 世代数 200, 交叉率 0.7, 突然変異率 0.1, トーナメントサイズを 5 に設定した.

本節では, 上記のパラメタを入力値として, 各コンテンツ m の目標ヒット率 $\hat{h}_m$ と実際のヒット率 h_m の比較をし, 提案方式を用いることで適切にキャッシュ挿入確率 f_m が設定できていることを確認する. また, N の二つの場合について, 平均取得成功率 (ES からのチャンク取得のみでコンテンツを復元できた割合) を提案方式を用いて f_m を設定した場合 ($f_m(GA)$) と, キャッシュミス時に常にコンテンツをキャッシュした場合 ($f_m(1.0)$) について, 各 ES の稼働率 r を変化させた場合と, Zipf のパラメタ θ を変化させて異なる要求レートを与えた場合で比較を行う.

5.2 目標ヒット率と実際のヒット率の比較

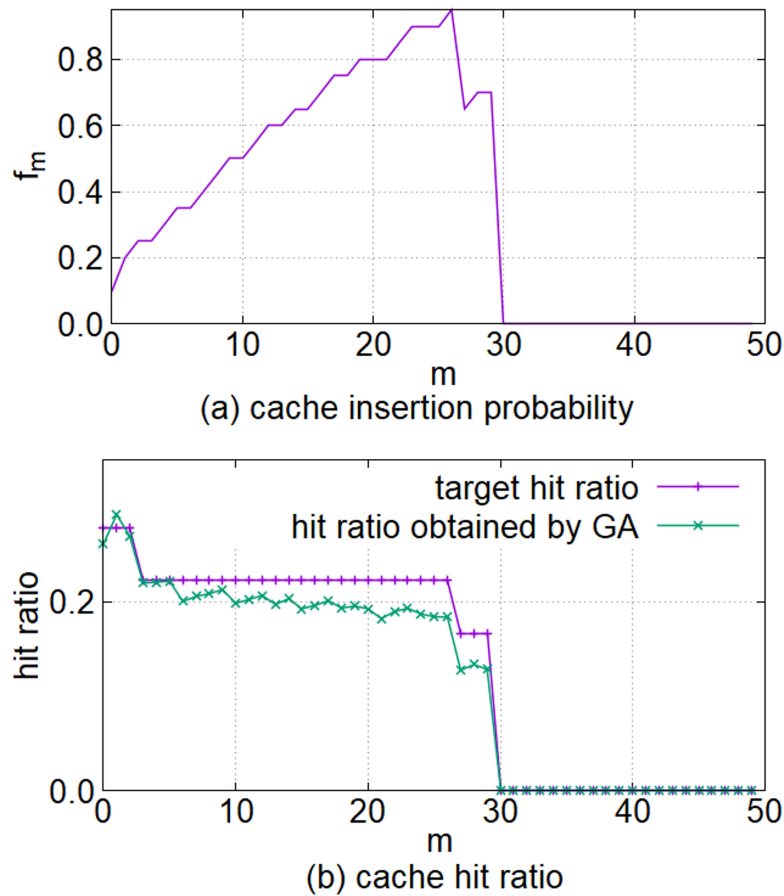


図 5.1: (a) 各コンテンツ m のキャッシュ挿入確率 f_m , (b) 目標ヒット率 $\hat{h}_m$ と提案方式を用いた場合のキャッシュヒット率 h_m の比較

図 5.1 の (a) は、各コンテンツ m の GA で算出された最適キャッシュ挿入確率 f_m を、また図 5.1 の (b) は GA で算出された f_m を用いたときのコンテンツ m のヒット率 h_m と目標ヒット率 $\hat{h}_m$ を、各々 m に対してプロットしたグラフである。 m は、人気順を示しており、値が小さいほど人気度が高くなる。キャッシュ挿入確率 f_m を提案方式で最適化することで、実際のヒット率 h_m と目標ヒット率 $\hat{h}_m$ の差異は小さく、適切にキャッシュ挿入確率 f_m を設定できることが確認できる。

5.3 各 ES の稼働率 r に対する平均取得成功率の変化

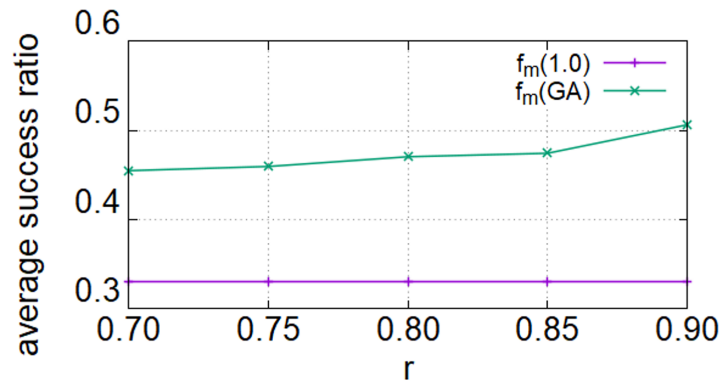
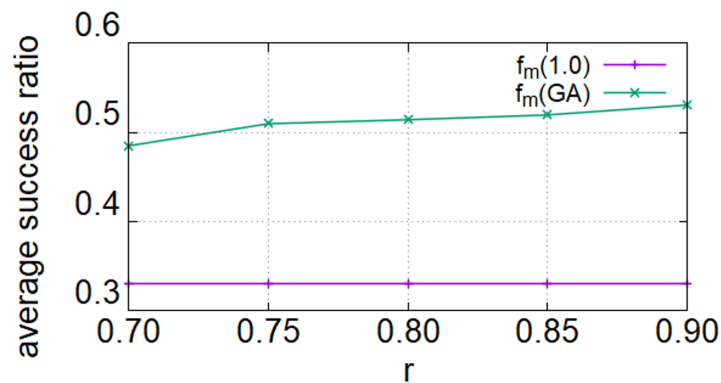
(a) $N=20$ (b) $N=30$ 図 5.2: 各 ES の稼働率 r に対する平均取得成功率

図 5.2 は, $N = 20, N = 30$, の場合について各々, 全コンテンツの平均取得成功率 (ES からのチャンク取得のみでコンテンツを復元できた割合) を, 提案方式を用いて f_m を設定した場合 ($f_m(GA)$) と, キャッシュミス時に常にコンテンツをキャッシュした場合 ($f_m(1.0)$) について, 各々, 各 ES の稼働率 r に対してプロットしたグラフである. グラフより, 各 ES の稼働率 r が増加するにつれて, ES の故障確率は減少し, 平均取得成功率はどちらの手法も増加する. 提案方式を用いることで, 単にコンテンツをすべて ES にキャッシュする場合と比較して, 平均取得成功率を大きく改善できることが確認できる.

5.4 異なるリクエストレートでの平均取得成功率の変化

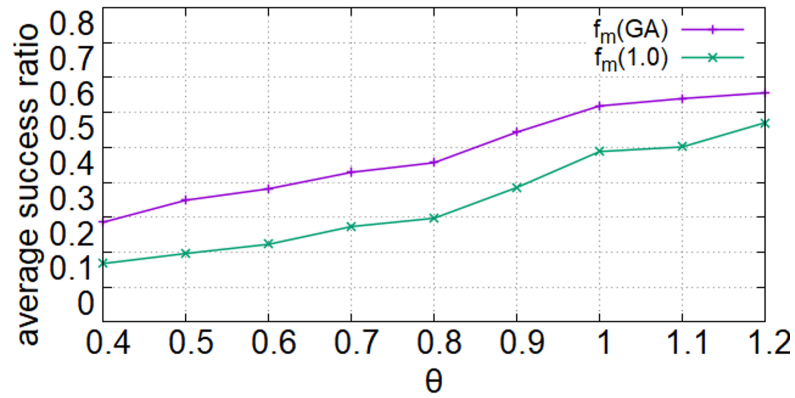
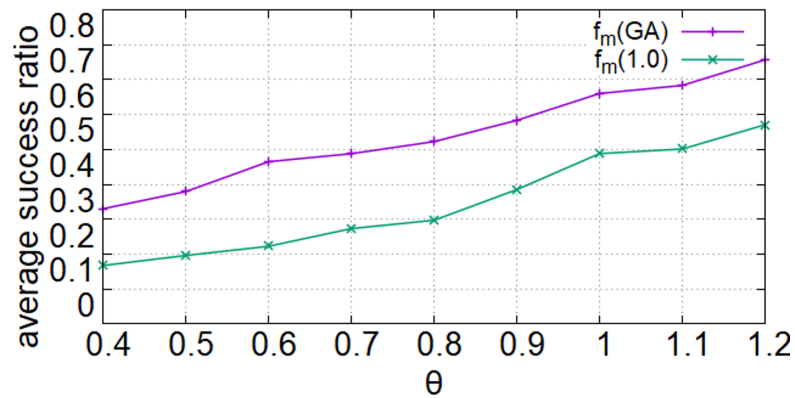
(a) $N=20$ (b) $N=30$

図 5.3: 異なるリクエストレートでの平均取得成功率

図 5.3 は, $N = 20$, $N = 30$ の場合について各々, 全コンテンツの平均取得成功率を, 提案方式を用いて f_m を設定した場合 ($f_m(GA)$) と, キャッシュミス時に常にコンテンツをキャッシュした場合 ($f_m(1.0)$) について, 各々, 異なる Zipf パラメタ θ に対してプロットしたグラフである. どちらの方式も, Zipf 分布の θ の値が大きくなると, 高人気コンテンツの要求比率が高くなると平均取得成功率が増加している, これは, LRU を用いた置換制御を行ってため, 人気の低いコンテンツのチャンクを優先的にキャッシュから削除するためである. また, 提案方式では, 既存研究をもとに目標チャンク数 T_m を設定し, 設計値に近づけるようにキャッシュ挿入確率 f_m を最適設計しているため, 平均取得成功率を大きく改善できることが確認できる.

第6章 まとめ

5G では、トラフィックを削減するために無線の基地局にキャッシュサーバを設置しユーザに近い ES からコンテンツを配信する MEC が用いられている。しかし ES の設置数が膨大であることからコストの低い低信頼な ES が用いられており、故障発生による不稼働率の増加が課題である。低信頼な ES を想定した研究 [12] では、ES の信頼性の低さに対処するために、erasure coding を用いて ES の可用性を高める方式を提案しているが、コンテンツを ES 上に事前配置している。しかし、実際の ES では LRU 等を用いた動的制御が一般的である。そこで、本稿では、低信頼な ES において LRU を用いた置換制御を想定した ES の不稼働率を考慮した上で、ES からのコンテンツ取得可能性を向上させる erasure coding を用いた ES へのチャンク挿入アルゴリズムを提案をした。また、性能評価から以下のことを確認した。

- キャッシュ挿入確率 f_m を提案方式で最適化することで、実際のヒット率 h_m と目標ヒット率 $\hat{h}_m$ の差異は小さく、適切に f_m を設定できていることを確認した。
- キャッシュ挿入確率 f_m を提案方式を用いて設定した場合 ($f_m(GA)$) と、キャッシュミス時に常にコンテンツをキャッシュした場合 ($f_m(1.0)$) で、平均取得成功率の比較を行い、提案方式を用いることで平均取得成功率を大きく改善することを確認した。

謝辞

本研究を行うに当たり，ご指導を頂いた上山教授に感謝します．また日常，有益な議論をして頂いた研究室の皆様にも感謝します．

参考文献

- [1] Y. Zeng, Y. Huang, J. Liu, and Y. Yang, "Privacy-preserving distributed edge caching for mobile data offloading in 5g networks," in 2020 IEEE 40th International Conference on Distributed Computing Systems (ICDCS), 2020, pp. 541–551.
- [2] L. Ramaswamy, L. Liu, and A. Iyengar, "Cache clouds: Cooperative caching of dynamic documents in edge networks," in 25th IEEE Conference on Distributed Computing Systems (ICDCS'05), 2005, pp. 229–238.
- [3] Y. Liu, X. Shang, and Y. Yang, "Joint sfc deployment and resource management in heterogeneous edge for latency minimization," IEEE Transactions on Parallel and Distributed Systems, vol. 32, no. 8, pp. 2131–2143, 2021.
- [4] X. Shang, Z. Liu, and Y. Yang, "Online service function chain placement for cost-effectiveness and network congestion control," IEEE Transactions on Computers, pp. 1–1, 2020.
- [5] A. G. Dimakis, P. B. Godfrey, Y. Wu, M. J. Wainwright, and K. Ramchandran, "Network coding for distributed storage systems," IEEE Transactions on Information Theory, vol. 56, no. 9, pp. 4539–4551, 2010.
- [6] V. R. Cadambe, S. A. Jafar, H. Maleki, K. Ramchandran, and C. Suh, "Asymptotic interference alignment for optimal repair of mds codes in distributed storage," IEEE Transactions on Information Theory, vol. 59, no. 5, pp. 2974–2987, 2013.
- [7] D. Bindel, Y. Chen, P. Eaton, D. Geels, R. Gummadi, S. Rhea, H. Weatherspoon, W. Weimer, W. Weimer, C. Wells et al., "Oceanstore: An extremely wide-area storage system," in Proceedings of the 9th International Conference on Architectural Support for Programming Languages and Operating Systems. Citeseer, 2000, pp. 190–201.
- [8] Q. Wei, B. Veeravalli, B. Gong, L. Zeng, and D. Feng, "Cdrm: A cost-effective dynamic replication management scheme for cloud storage cluster," in 2010 IEEE International Conference on Cluster Computing, 2010, pp. 188–196.
- [9] A. Soro and J. Lacan, "Fnt-based reed-solomon erasure codes," in 2010 7th IEEE Consumer Communications and Networking Conference, 2010, pp. 1–5.
- [10] S. Ghemawat, H. Gobioff, and S.-T. Leung, "The google file system," in Proceedings of the nineteenth ACM symposium on Operating systems principles, 2003, pp. 29–43.
- [11] D. Ford, F. Labelle, F. Popovici, M. Stokely, V.-A. Truong, L. Barroso, C. Grimes, and S. Quinlan, "Availability in globally distributed storage systems," 2010.
- [12] Y. Liu, et al., Distributed Cooperative Caching in Unreliable Edge Environments, IEEE INFOCOM 2022
- [13] H. Che, et al., "Hierarchical Web Caching Systems: Modeling, Design and Experimental Results," IEEE J. Selected Areas of Commun., vol. 20, no. 7, Sep. 2002
- [14] L. Ramaswamy, L. Liu, and J. Zhang, "Efficient formation of edge cache groups for dynamic content delivery," in 26th IEEE International Conference on Distributed Computing Systems (ICDCS '06). IEEE, 2006, pp. 43–43.

学会発表リスト

- 吉田 開, 上山 憲昭, ”低信頼な ES を想定した高信頼キャッシュ法”, 電子情報通信学会総合大会, 広島, 2024 年 3 月
- 吉田 開, 上山 憲昭, ”低信頼な ES を想定した高信頼キャッシュ法”, 電子情報通信学会インターネットアーキテクチャ(IA) 研究会, 宮古島, 2024 年 3 月