2008－AVM－61（15）
2008 / 7 / 24

ユビキタス環境におけるキャッシュ一貫性管理プロトコル

上西　裕†　早川　裕志†　田頭　茂明††　中西　恒夫††　福田　晃††

† 九州大学大学院システム情報科学府　〒819-0395 福岡市西区元岡 744 番地
†† 九州大学大学院システム情報科学研究院　〒819-0395 福岡市西区元岡 744 番地
E-mail: †{kaminishi,h2,shigeaki,tun,fukuda}@f.csce.kyushu-u.ac.jp

あらまし　本稿では，ユビキタス環境における端末の省電力化技術について検討する．特に，端末の省電力化のアプローチとしてキャッシュ技術に着目しており，キャッシュを用いて不必要な通信を削減することでネットワークデバイスの省電力化を試みる．キャッシュ技術の利用にはキャッシュデータの一貫性が管理されている必要がある．これまでにキャッシュデータの一貫性を管理する数多くの既存手法が提案されており，本稿では既存手法の調査とまとめを行う．さらに我々の想定するユビキタス環境を紹介し，キャッシュの確率的な一貫性プロトコルを提案する．

キーワード　省電力，キャッシュ，一貫性管理プロトコル，ユビキタス環境

A Cache Consistency Protocol for Ubiquitous Environment

Yutaka KAMINISHI†, Hiroshi HAYAKAWA†, Shigeaki TAGASHIRA††, Tsuneo NAKANISHI††, and Akira FUKUDA††

† Graduate School of Information Science and Electrical Engineering, Kyushu University.　744 Motooka Nishi-ku, Fukuoka 819-0395, Japan
†† Faculty of Information Science and Electrical Engineering, Kyushu University.　744 Motooka Nishi-ku, Fukuoka 819-0395, Japan
E-mail: †{kaminishi,h2,shigeaki,tun,fukuda}@f.csce.kyushu-u.ac.jp

Abstract　In this paper, we consider power-saving techniques for mobile terminals in ubiquitous environment. As a means for realizing power-saving, we focus on caching techniques and attempt to save the power consumption required for wireless network devices by reducing meaningless transmissions using local caches. The use of caching techniques is required to manage the consistency between original data and their caches. Many existing protocols for keeping the cache consistency have been extensively developed during the past decade so far, and we investigate and summarize these existing techniques in this paper. Finally, we describe our target ubiquitous environment and propose a probabilistic cache consistency protocol.

Key words　Power-saving, Caching, Consistency protocol, Ubiquitous environment

1. はじめに

無線ネットワークの普及に伴い，屋内だけでなく屋外においても，様々なネットワークサービスを利用できる環境が整いつつある．特に，携帯電話端末や小型携帯端末などのモバイル端末においては，屋外向けのネットワークサービスが拡充されており，外出時において有益な各種情報へのアクセスが容易になっている．これらのサービスは一般的に，インストールや設定が不要で，開発が容易にできるという点から，WEB をベースにしたものが多い．すなわち，これらのサービスはモバイル端末がオンライン状態でないと利用できず，サービス利用中はネットワークへのアクセスが必要となる．

一方で，半導体技術やパッケージング技術の飛躍的な進歩に伴い，モバイル端末の著しい高速化および小型化が進められている．しかしながら，これらの進歩に対して，バッテリ技術の性能向上が追いついておらず，現在のモバイル端末における可搬性のボトルネックはバッテリの持続時間であるといえる．特に，CPU や記憶装置などは半導体技術のさらなる進歩によりその省電力化を期待できるが，無線ネットワークデバイスに関しては，電波の送受信に電力を消費することが一般的であり，その省電力化は困難であることが予測できる．また，先にも述べたがネットワークサービス利用中はネットワークへのアクセ

スが必要であるため，無線ネットワークデバイスの省電力化が，ユビキタス環境の実用化へ向けたもっとも重要な研究課題であるといえる．本研究ではモバイル端末の省電力化を実現する為に，キャッシュ技術を利用してサービス利用中のネットワークアクセスの削減を試みる．

これまでにモバイル端末の省電力化の為に，キャッシュ技術を利用するアプローチが注目されてきた．頻繁にアクセスされるデータをキャッシュしておくことで，モバイル端末からネットワークへアクセスする必要が少なくなり，通信量の削減に効果があることが示されている [1]- [10]．キャッシュデータの利用には，キャッシュデータの一貫性が管理されている必要がある．本稿では，これまでに提案されたキャッシュデータの一貫性を管理するプロトコルを紹介する．特に，それらのプロトコルを想定しているネットワークモデルと，アクセスモデルの観点から分類し，それぞれの特徴について述べる．また，我々が想定しているユビキタス環境を紹介し，その環境でのキャッシュ一貫性プロトコルを提案する．提案手法は，Bloom フィルタを利用した確率的なキャッシュの一貫性プロトコルであり，以下の二つの特徴を持つ．

- キャッシュデータの更新リストを Bloom フィルタで表現することで通信量を削減する．
- ソースノードが個々のモバイル端末のキャッシュの状態を把握しなくても (Stateless に)，モバイル端末に個別の更新リストを作成できるようにし，一貫性管理に必要なコストを削減する．

以上の一貫性管理プロトコルを実現する具体的なアルゴリズムについても記述する．

本稿の以降の構成は第 2 章で既存のキャッシュデータの一貫性管理アルゴリズムについてまとめる．また第 3 章で我々の想定するユビキタス環境を紹介し，モバイル端末の省電力化に適したキャッシュデータの一貫性の管理方針を検討する．最後に第 4 章で本稿のまとめと今後の課題について述べる．

2. 関連研究

キャッシュデータの一貫性の管理には，キャッシュデータが適切に無効化されて，キャッシュデータの更新を促すアルゴリズムが必要である．キャッシュデータを無効化するアルゴリズムについては様々な研究が進められてきた．無線ネットワーク環境でのネットワークモデルは，大きくインフラストラクチャベースのモデル (図 1) とアドホックベースのモデル (図 2) の二つに分けることができる [6]．図 1，2 において，ネットワークに固定された端末のことを Mobile Support Station(以下，MSS) と呼ぶ．インフラストラクチャベースのモデルでは，MSS にモバイル端末が 1 ホップで接続し，MSS がモバイル端末の送受信するメッセージを転送する状況を想定する．一方でアドホックベースのモデルでは，MSS に 1 ホップで接続できないモバイル端末は MSS に接続可能なモバイル端末を経由して MSS に接続する．この場合，MSS はアクセスポイントの役割を果たす状況を想定している．この結果，それぞれのネットワークモデルでモバイル端末のキャッシュデータの一貫性を管理する方針が異なる．

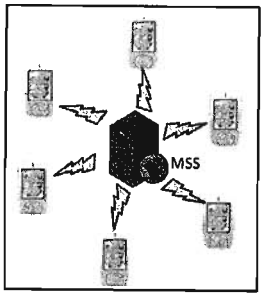

図 1　インフラストラクチャベースモデル

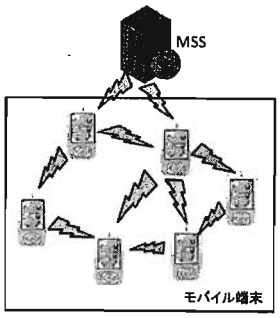

図 2　アドホックベースモデル

キャッシュデータを無効化するアルゴリズムには大きく Push 型と Pull 型の二つのアルゴリズムが存在する．Push 型のキャッシュデータの無効化では，MSS が能動的にモバイル端末の持つキャッシュデータの無効化を行う．モバイル端末は MSS からのキャッシュデータの無効化の通知を待つだけでよい為，モバイル端末の省電力化に貢献しやすい．しかし，オリジナルデータが MSS 側でしか更新されないと仮定すると，モバイル端末がキャッシュデータの一貫性を確認するには MSS による次の無効化の通知を待たなければならない．一方，Pull 型のキャッシュデータの無効化では，モバイル端末が自身の持つキャッシュデータの一貫性の確認を MSS に要求する．この場合，キャッシュデータの更新の有無に関わらず，モバイル端末はキャッシュデータを利用するたびに更新を確認しなければならない為，モバイル端末のバッテリーを消費しやすい傾向がある．以下では，インフラストラクチャベースモデルとアドホックベースモデルにおけるキャッシュデータの具体的な無効化アルゴリズムについて記述する．

2.1 インフラストラクチャベースモデルにおけるアルゴリズム

インフラストラクチャベースモデルにおけるキャッシュデータの無効化には MSS が Stateful にモバイル端末を管理する方法と Stateless に管理する方法の二つのアプローチがある．Stateless なアプローチとして，タイムスタンプを用いたプロトコルである TS 法が提案された [1]．TS 法は，MSS が周期 L でキャッシュデータの無効化情報 (IR: Invalidation Report) を MSS に接続しているモバイル端末にブロードキャストする

Push 型のアルゴリズムである．TS 法の概要を図 3 に示す．TS 法には次の周期の IR まで待たないとモバイル端末が IR を受信できないという問題 (受信遅延) がある．また，この IR には $w(w > L)$ 秒の間に更新されたオリジナルデータの更新時間が記録されている．モバイル端末が w 秒以上 MSS に接続していない状況が発生すると，モバイル端末は自身の持つキャッシュデータをすべて無効にしなければならない．

TS 法で問題になったモバイル端末の受信遅延を軽減する手法として，Updated Invalidation Report(以下，UIR) を用いる手法が提案された [10]．この手法は TS 法と同様に Stateless なアプローチを採用した Push 型のアルゴリズムである．UIR には，最後にブロードキャストした IR 以降に更新されたデータの更新時間が記録されている．UIR を IR の周期の間に複数回送信することで，モバイル端末の受信遅延を軽減している (図 4 参照)．しかしながら，データの更新が頻繁に起こると，ブロードキャストするメッセージサイズが増大し，ネットワーク帯域を圧迫してしまう問題や，IR や UIR といったメッセージを受信する回数が増えるためモバイル端末のバッテリを消費しやすいという問題もある．

このような Stateless なアプローチとは対称的に，MSS が Stateful にモバイル端末を管理し，IR が非同期 (非周期的) に送信される Push 型のキャッシュデータの無効化アルゴリズム Asynchronous Stateful プロトコル (AS 法) が提案された [4]．文献 [4] では，MSS がモバイル端末のキャッシュ状況をすべて把握しているため，オリジナルデータの更新があった場合に，MSS は特定のモバイル端末にのみ即座に IR を送信することができる．これにより IR の無駄なブロードキャストを避けることができる上に，エラーフリーな環境ではキャッシュデータの一貫性が常に保たれる．さらに，TS 法や UIR を用いた手法で問題となった一定時間以上 MSS に接続しない状況が発生しても，MSS がモバイル端末のキャッシュ状況を把握しているため，すべてのキャッシュデータを無効にする必要がない．しかしながら，MSS に接続する端末数が増えることで，MSS でのキャッシュ状況の管理が困難になりスケーラビリティに欠けるという問題がある．また，ネットワークトポロジーが動的に変化しやすい無線通信環境では端末を管理すること自体が困難である．

そこで，Stateless と Stateful のハイブリッド型である Push 型のアルゴリズムとして Scalable Asynchronous Cache Consistency Scheme (SACCS 法) が提案された [7]．SACCS 法における MSS の管理負担を軽減するアイデアとして以下の三つの方法が提案されている．

- フラグビットの利用
- TTL の利用
- ID の利用

SACCS 法では，MSS でフラグビットを利用することで，不要な IR と MSS の管理負担を削減している．すなわち，モバイル端末が MSS からオリジナルデータをキャッシュした時，キャッシュされたオリジナルデータのフラグビットが MSS でセットされる．MSS はフラグビットがセットされたオリジナルデータが更新されたら即座に IR をブロードキャストしてフラグビットをリセットする．

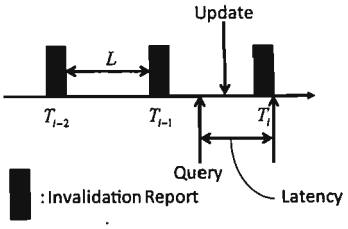

図 3　TS 法の概要

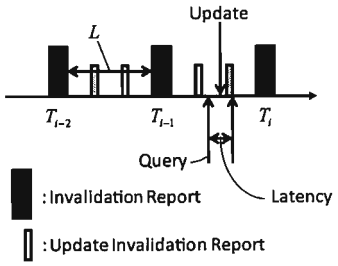

図 4　UIR による受信遅延の軽減

このように IR を非同期 (非周期的) に送信することで，TS 法のように不要な IR を周期的にブロードキャストする必要がなくなる．また，IR をブロードキャストする対象となるオリジナルデータをフラグビットのみで判断するため，AS 法よりも MSS の管理負担を削減している．

MSS は各オリジナルデータにそれぞれのデータの更新履歴から，Time-To-Live(以下，TTL) を設定する．TTL はキャッシュデータと共にモバイル端末にキャッシュされる．モバイル端末上で TTL が切れたキャッシュデータは SACCS 法では uncertain 状態 (キャッシュデータの一貫性が明白でない状態) とする．uncertain 状態を導入することで，IR を一定時間以上受け取っていない状況でも，古いキャッシュデータが利用されることがなくなる．uncertain 状態のキャッシュデータを利用する時は，モバイル端末は MSS にキャッシュデータの一貫性を確認する必要がある．また，IR を受けて無効化されたキャッシュデータの ID だけを保持している状態を SACCS 法では ID-only という状態としている．ID-only 状態のキャッシュデータは新たに MSS からオリジナルデータをキャッシュする必要がある．図 5 にキャッシュデータの状態遷移の様子を示す．

2.2　アドホックベースモデルにおけるアルゴリズム

アドホックベースモデルでもキャッシュデータの無効化方式として，Stateless と Stateful な方式が考えられるが，MSS からマルチホップ先のモバイル端末のキャッシュ状況まで把握する Stateful な方式は現実的ではないと考えられる．そこで，MSS が Stateless と Stateful のハイブリッド型で，かつ Push 型と Pull 型のハイブリッド型アルゴリズムである Relay-Peer-based Cache Consistency(RPCC 法) が提案された [6]．RPCC 法では，オリジナルデータを持つモバイル端末 (ソースノード) とキャッシュデータを持つモバイル端末 (キャッシュノード) を中継

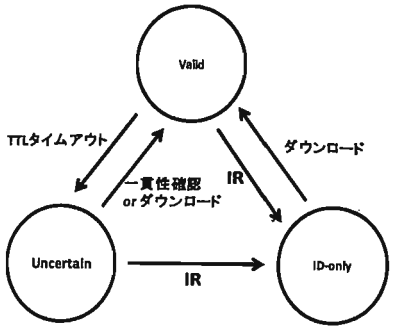

図 5　SACCS 法におけるキャッシュデータの状態遷移の様子

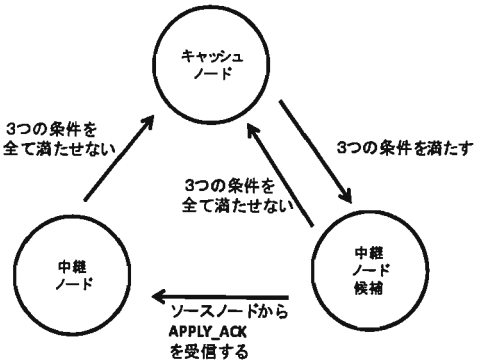

図 6　キャッシュノードから Relay peer への遷移の様子

するモバイル端末 (中継ノード) を適切に選択することで，ネットワークトラフィックと受信遅延の削減に成功している．すなわち，中継ノードを導入することで，ソースノードが中継ノードに IR を定期的に送信すること (Push 型) と，キャッシュノードが中継ノードからキャッシュデータの一貫性の確認をすること (Pull 型) の二つの手順を非同期に行い，ネットワークトラフィックと受信遅延を削減できる．また，キャッシュノードが一定期間ネットワークに接続していない場合，ソースノードにポーリングすることなく，近隣の中継ノードにポーリングすることでキャッシュデータの一貫性を確認することができる．さらに，RPCC 法では，ソースノードは中継ノードまでのキャッシュ状況を把握すればよい為，キャッシュ状態の管理が容易である．具体的には RPCC 法において，中継ノードは以下の三つの基準から選択される．

- ソースノードへのアクセスレート
- 安定性
- バッテリー残量

RPCC 法におけるモバイル端末は，キャッシュノード，中継ノードの候補ノード，中継ノードと三つの状態を持つ．図 6 にキャッシュノードから中継ノードへの遷移の様子を示す．図 6 より，キャッシュノードは上記の三つの基準に関してある閾値を全て満たすと中継ノードの候補ノードとなる．中継ノードの候補ノードは，ソースノードからキャッシュデータの IR を受け取ることができれば，中継ノードとなる．

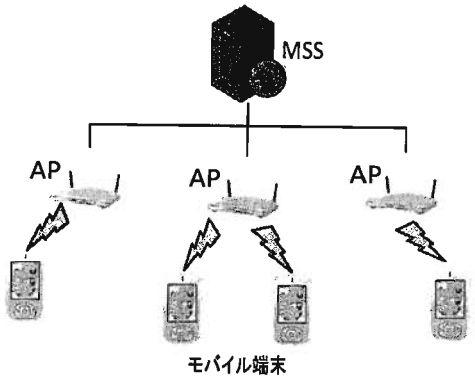

図 7　想定環境

2.3　その他のキャッシュ管理技術

文献 [11] では，構造型・非構造型 Peer-to-Peer ネットワークのそれぞれにおいて，各ノードがもつキャッシュ情報の管理技術についての性能評価が行われている．特にキャッシュ情報の検索技術に焦点が当てられており，以下の三つの方式が評価の対象となっている．

- Bloom フィルタを利用した DHT による検索技術
- super-peer ネットワークを利用した検索技術
- random-walk 探索を利用した検索技術

ここでは，特に Bloom フィルタを利用した DHT による検索技術 [12] に注目する．文献 [12] では，各モバイル端末が保持しているデータを転置リスト (モバイル端末が管理しているキャッシュデータのリスト) の形式で DHT に格納する．テキストの検索を行う時は，DHT 上の転置リストから対象となるキャッシュデータを所有しているモバイル端末を探索する．ネットワーク上のキャッシュデータ数が増大すると，転置リストのサイズが増大する．このためモバイル端末間で転置リストの転送を行うと，ネットワーク帯域を圧迫してしまうという問題がある．そこで文献 [12] では，このようなネットワークトラフィックの増大を抑える為に，転置リストを Bloom フィルタで表現することで，ネットワークトラフィックの削減に効果があったことが示されている．しかしながら，Bloom フィルタの偽陽性から，検索遅延が増大してしまうという問題がある．この問題に対しては，偽陽性を排除した結果をキャッシュして利用することで，ネットワークトラフィックと検索遅延を削減している．

3.　確率的一貫性管理方式

3.1　設計方針

本稿では，Stateless な Pull 型のキャッシュ情報の確率的一貫性管理方式を提案する．我々の想定する無線通信環境は，インフラストラクチャベースモデルであり，サーバ（ソースノード）がオリジナルのデータを保持することを前提としている．加えて，モバイル端末は MSS に 1 ホップで接続するが，MSS がアクセスポイントの役割しか果たさないことを前提としている (図 7 参照)．これは，無線 LAN を用いてモバイル端末が WEB サービスを受信する一般的な環境をモデル化している．

本研究の目的はモバイル端末の省電力化であるが，我々の想定する環境では，MSS が単なる AP の機能しか持たないため，MSS からモバイル端末に IR を送信することができず，キャッシュデータの無効化には Push 型の IR を利用したアルゴリズムを適用することができない．Pull 型のアルゴリズムでは，キャッシュデータを利用するたびにキャッシュデータの一貫性を確認しなければならず，モバイル端末の省電力化には適していない．そこで，我々は Bloom フィルタを利用して，Pull 型をベースにしたキャッシュデータの一貫性の確認に不必要な通信を削減する仕組みを提案する．特に，我々が提案するキャッシュデータの一貫性管理のポイントは以下の二点である．

- キャッシュデータの更新リストを Bloom フィルタで表現することで通信量を削減する．
- ソースノードが個々のモバイル端末のキャッシュの状態を把握しなくても (Stateless に)，モバイル端末に個別の更新リストを作成できる．

前者のポイントは，Bloom フィルタで表現された更新リストを取得するだけで，モバイル端末が保持するキャッシュデータの更新を確認することができる．すなわち，その確認に関してはネットワークを利用しないため省電力化を実現することができる．また後者では，ネットワーク上に存在するモバイル端末が保持するキャッシュデータは様々であることが予想される．すなわち，端末間で異なるデータがキャッシュされている，また同一のデータがキャッシュされているとしても，そのキャッシュした時刻は異なることが予想できる．この為，キャッシュデータの更新情報は端末間で個別に用意する必要がある．提案方式では，Bloom フィルタの更新リストを取得する際に，各モバイル端末が持つキャッシュデータを表現する Bloom フィルタをソースノードに送信し，そのモバイル端末に応じた更新リストを取得するようにしている．また，キャッシュ時刻に関しては，モバイル端末のキャッシュデータを時間で区切って，時間帯ごとの Bloom フィルタを作成し，Bloom フィルタに対して時間の概念を導入した．

3.2 キャッシュデータの無効化処理

モバイル端末が時間帯別のキャッシュデータの集合に対して作成した Bloom フィルタを利用して，ソースノードへキャッシュデータの一貫性を確認する様子を以下で説明する．はじめに，ソースノードへ送信する Bloom フィルタは二種類ある．一つは時間帯別に作成した Bloom フィルタと，もう一つはモバイル端末の持つ全てのキャッシュデータの Bloom フィルタである．時間帯別に作成した Bloom フィルタの例を図 8 に示す．それぞれの時間帯でキャッシュしたデータの集合の Bloom フィルタにタイムスタンプが付与されていることが分かる．

キャッシュデータの一貫性の確認が必要である時，図 8 に示すような時間帯別に作成した Bloom フィルタをソースノードへ送信する．また，モバイル端末が持つ全てのキャッシュデータの集合の Bloom フィルタもソースノードへ送信することで，ソースノードはモバイル端末が持つキャッシュデータの集合をある程度絞り込むことができ，ソースノード内の全てのソースデータについて一貫性を確認する必要がなくなる．

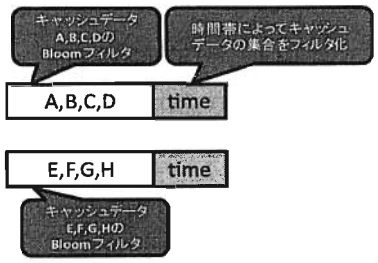

図 8　キャッシュデータの時間帯別 Bloom フィルタ 1

表 1　キャッシュデータのハッシュ値表

キャッシュデータ	ハッシュ値
A	10000000
B	00100000
C	00001000
D	00000010
E	01000000
F	00010000
G	00000100
H	00000001

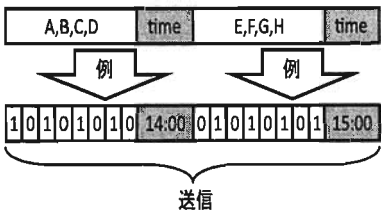

図 9　キャッシュデータの時間帯 Bloom フィルタの例 2

ソースノードで，モバイル端末から送られてきた時間帯別の Bloom フィルタから，キャッシュデータに更新が必要かどうかを確認する様子を図 10 に示す．今モバイル端末のキャッシュデータのハッシュ値を表 1 のように仮定する．よって図 8 と表 1 の例に従うとすると，図 9 のような時間帯別の Bloom フィルタがソースノードに送信されてくることになる (タイムスタンプは簡単のため，それぞれの Bloom フィルタに対して 14 時・15 時とした)．また，ソースノードとモバイル端末では同じ一つのハッシュ関数を利用するものとする．

図 10 において，キャッシュデータに A が含まれているとソースノードが判断したと仮定して，オリジナルデータ A と時間帯別の Bloom フィルタを比較した結果，A が 14 時以降にソースノード上で更新されている可能性があることが分かる．

そこで，時間帯別の Bloom フィルタのタイムスタンプとオリジナルデータ A のタイムスタンプの比較を行い，時間帯別の Bloom フィルタのタイムスタンプがオリジナルデータのタイムスタンプよりも古ければ，更新が必要なキャッシュデータ用の Bloom フィルタに新たに A の要素が追加される．

ソースノードで作られた更新が必要なキャッシュデータ用の Bloom フィルタを受け取ったモバイル端末は，更新が必要なキャッシュデータを無効化し，ソースノードにアクセスして最新のオリジナルデータをキャッシュする．受け取った Bloom フィ

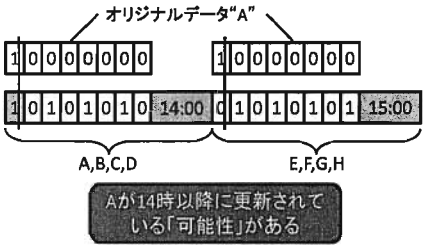

図 10　キャッシュデータの更新の可能性

ルタには一定の期限があり，期限が切れると最初の手順に戻り，再度キャッシュデータの一貫性の確認を行う．このように，モバイル端末は更新が必要なキャッシュデータ用の Bloom フィルタを受け取ることでキャッシュデータの無効化を行う．

4. まとめと今後の課題

本稿で我々はインフラストラクチャベースモデルとアドホックベースモデルのそれぞれの無線ネットワーク環境におけるキャッシュデータの一貫性管理を行う既存アルゴリズムについてまとめた．また，我々は調査した結果をもとに，我々の想定する環境でモバイル端末の省電力化に適したキャッシュデータの一貫性管理について，Bloom フィルタを利用するキャッシュデータの一貫性管理方式について述べた．

今後の課題は，第 3 章で提案した仕組みを実装し，Bloom フィルタがネットワークトラフィックと受信遅延にどのような影響を与えるか調査することである．また，提案手法において，キャッシュデータの Bloom フィルタを作成するタイムスタンプの頻度が，キャッシュデータの一貫性へどのような影響を与えるか調査することである．

文　献

[1] Daniel Barbara, Tomasz Imielinski, "Sleepers and workaholics: caching strategies in mobile environments," ACM SIGMOD Conference on Monagement of Data, pp.1-12, June 1994.

[2] Sunho Lim, Wang-Chien Lee, Guohong Cao, Das, C.R., "Performance comparison of cache invalidation strategies for Internet-based mobile ad hoc networks," Mobile Ad-hoc and Sensor Systems, 2004 IEEE International Conference, 25-27 Oct. 2004, pp.104- 113.

[3] Sunho Lim, Wang-Chien Lee, Guohong Cao, Chita R.DAS, "Cache invalidation strategies for internet-based mobile ad hoc netwoks," Computer Communications Volume 30, Issue 8, 8 June 2007, pp.1854-1869.

[4] A. Kahol, S. Khurana, S.K.S. Gupta, and P.K. Srimani, "A Strategy to Manage Cache Consistency in a Distributed Mobile Wireless Environment," IEEE Trans. Parallel and Distributed Systems, vol. 12, no. 7, pp. 686-700, July 2001.

[5] Yu Huang, Jiannong Cao, Zhijun Wang, Beihong Jin, Yulin Feng, "Achieving Flexible Cache Consistency for Pervasive Internet Access," Pervasive Computing and Communications, PerCom '07. Fifth Annual IEEE International Conference, 2007, pp.239-250.

[6] Jiannong Cao, Yang Zhang, Guohong Cao, Li Xie, "Data Consistency for Cooperative Caching in Mobile Environments," The flagship magazine of the IEEE Computer Society April 2007(Vol.40, No.4), pp.60-66.

[7] Zhijun Wang , Sajal Das , Hao Che , Mohan Kumar, "A Scalable Asynchronous Cache Consistency Scheme (SACCS) for Mobile Environments," IEEE Transaction on parallel and distributed systems, Vol. 15, No. 11, November 2004, pp.983-995.

[8] Zhijun Wang, Mohan Kumar, Sajal K Das, and Huaping Shen, "Dynamic Cache Consistency Schemes for Wireless Celluar Networks," IEEE Transaction On Wireless Communications, Vol.5, No.2, February 2006, pp.366-376.

[9] Chang, Yeim-Kuan Ting, I-Wei Lin, Tai-Hong, "Dynamic Cache Invalidation Scheme in IR-Based Wireless Environments," Advanced Information Networking and Applications, 2008. AINA 2008. 22nd International Conference, pp.697-704.

[10] G. Cao, "A scalable low-latency cache invalidation strategy for mobile environments," IEEE Transactions on Knowledge and Data Engineering 15 (5) 2003 pp.1251-1265.

[11] Yong Yang, Rocky Dunloap, Michael Rexroad, Brian F. Cooper, "Performance of Full Text Search in Structured and Unstructured Peer-to-Peer Systems," in Proc. IEEE INFOCOM, 2006.

[12] P.Reynolds and A.Valdet, "Efficient peer-to-peer keyword searching," in Proc. International Middleware Conference, 2003.