マルチアイテム環境における コールドデータの効率的提供手法の提案

油田 健太郎[1]　山場 久昭[2]　朴 美娘[3]　岡崎 直宣[2]

概要：今日，車両を運転する様々な場面での事故や渋滞の軽減，またサービスアナウンス等の運転支援システムを目的にしたITS（高度道路交通システム）が注目されている．本論文では，路車間通信(V2I:Vehicle-to-Infrastructure)に着目し，路上に設置された無線端末のRSU(Road Side Unit)と車両が通信を行うシステムについて検討する．システムでは，予め路上に設置されているRSUにサーバから転送されたデータを蓄積し，車両がRSUの通信範囲に入った時にその情報を車両が要求するためにクエリを生成し，送信する．クエリにはシングルアイテムクエリとマルチアイテムクエリの2種類あり，前者は一つのクエリで一つのデータ項目を要求し，後者は一つのクエリで複数のデータ項目を要求する．マルチアイテムクエリにおけるデータ項目はアクセス頻度が高いものをホットデータ，低いものをコールドデータに分けるが，コールドデータの受信には場合によっては長い時間が掛かり，クエリの生存時間が切れる可能性があることが問題点として挙げられる．そこで，本論文では，既存アプローチに新たな選択方法のアルゴリズムを組み込むことで，コールドデータを含むマルチアイテムクエリの効率的提供方法を提案する．シミュレーションにより，大幅にデットラインミス率を向上できることを明らかにした．

キーワード：ITS（Intelligent Transport Systems）,V2I（Vehicle-to-Infrastructure），CQS（Cooperative Query Serving），RXW（Number of pending Request Multiply Waiting time）

1. はじめに

今日，車両を運転する様々な場面での事故や渋滞の軽減，またサービスアナウンス等の運転支援システムが開発されている[1]．このようなものをITS（高度道路交通システム）[2]と呼び，車両同士で通信を行う車々間通信(V2V:Vehicle-to-Vihicle)，路上に設置された端末と車両で通信を行う路車間通信(V2I:Vehicle-to-Infrastructure)，路車間で連携するシステム(V2X:Vehicle-to-X)に分けられる．本論文では，路上に設置された無線端末のRSU(Road Side Unit)と車両で通信を行うシステムについて検討するため，路車間通信に着目する．

路車間通信において，予め路上に設置されているRSUにサーバから転送されたデータを蓄積し，車両がRSUの通信範囲に入った時にその情報を車両が要求するためにクエリを生成・送信するシステムについて検討する．クエリにはシングルアイテムクエリとマルチアイテムクエリの2種類あるが，前者は一つのクエリで一つのデータ項目を要求し，後者は一つのクエリで複数のクエリを要求する[3][4]．マルチアイテムクエリにおけるデータはアクセス頻度が高いものをホットデータ，低いものをコールドデータに分けるが，コールドデータの受信には場合によっては長い時間が掛かる[5][6][7]．これが原因となりクエリの生存時間が切れる可能性があることが問題点として挙げられる．

これを解決するためにCQS（Cooperative Query Serving）アプローチと呼ばれる既存手法がある[8]．この手法では，RSUがコールドデータを含むクエリの提供に時間を掛け過ぎることを防止するために，提供クエリを到着順に送信するのでなく，3つの基準によって優先度を決める．その条件は，1.QL' 値（クエリの長さ），2.QLD(Query Local Deadline)，3. クエリの提供時間である．また，RSUが時間内に要求されたデータを提供できないと判断した場合，隣接RSUへクエリを転送することも可能である．

その他の既存手法について簡潔に説明する．まずFCFS(First Come First Service)があり，これは最もシンプルな手法で，RSUの受信キュー（待ち行列）におけるクエリの到着順に従ってクエリを選択する．そのためクエリに優先度はなく，連続したブロードキャストによってデータを提供する．2つ目はMRF（Most Requests First）で，クエリ内の要求データ項目数が最も大きいものを選択し，

[1] 大分工業高等専門学校
Oita National College of Technology
[2] 宮崎大学
University of Miyazaki
[3] 神奈川工科大学
Kanagawa Institute of Technology

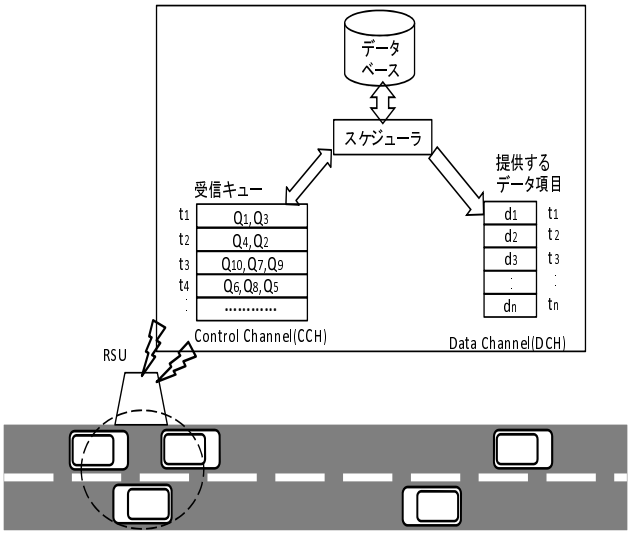

図 1 システムアーキテクチャ

ブロードキャストする手法である．そのため，データに優先度がある．

CQS，FCFS，MRF については文献 [8] で既に比較済みであり，デッドラインミス率を評価基準としている [8]．デッドラインミス率とは，RSU の受信キュー内にあるクエリ総数のうち，生存時間が切れたクエリの割合のことである．そして比較した結果は，優先度に関係なくクエリ選択を行う FCFS と，要求データ項目数が大きいクエリから選択する MRF に対し，完成させやすいクエリから選択する CQS が最も優れていた．これは，先に述べた CQS の手法により，完成までの時間が短いクエリから提供したために，転送に成功したクエリ数が多くなったからであると考えられる．しかし CQS の 3 つのクエリ選択条件では，不足している点もある．例えば複数のクエリの長さ，QLD 値，提供時間が同じであったとき，RSU は条件に従ってクエリ選択ができない．そこで我々は，CQS アプローチに RXW(Number of pending Request Multiply Waiting time)[9] と呼ばれるアルゴリズムを組み合わせる手法を提案する．これは，CQS の 1 から 3 の基準でクエリの優先度を判断できない場合に，RXW 値によって提供するクエリを選択するものである．RXW 値について説明すると，まず R は未解決データ項目の数 で，W はクエリが RSU の受信キュー内で待機した時間である．この 2 つを掛け合わせたものが RXW 値である．この値が最も大きいものから選択し，ブロードキャストするアルゴリズムが RXW である．これにより CQS のクエリ選択の補助を行うことで，デッドラインミス率をより改善することができる．また，RXW がこのシステムに有効なものであるかを検証するため，デッドラインミス率を評価するシミュレーションを行い，その結果から RXW と CQS を組み合わせることでどのような効果が得られるのかを検討する．

2. 概要

本章では，システムの概要とシステム内で用いる定義について説明する．

2.1 システムの概要

この手法のシステムアーキテクチャを図 1 に示す．我々が提案した路車間通信システムは，それぞれの RSU が狭域通信を CCH と 1 つの DCH に分けることで成り立っている．CCH を通じて RSU は定期的な safety message の送信と，次に送信する non-safety message の索引情報の送信を行う．本論文で述べる safety message とは，交差点での衝突回避を促す等の運転者の安全に関係する情報のことであり，non-safety message とは，通行料の徴収やサービスアナウンス等の安全とは関係のない情報のことである [10]．本論文では DCH の non-safety message のスケジューリングに焦点を当てている．RSU の範囲内に入ると，車両は自身の CCH を通じて non-safety message を要求するためのクエリを生成することができ，それは RSU の範囲外に出るまで DCH を介して応答を受信する．RSU が相互接続されているため，隣接 RSU に部分的または完全に満たされていないクエリを転送することができる．RSU は，次のサイクルで提供するためのクエリを選択するために，基礎となるスケジューリングアルゴリズムを呼び出す．

2.2 マルチアイテムクエリ

マルチアイテムクエリは一つのクエリの中に複数のデータ項目を含むことができるクエリのことである [3][4]．また，RSU の持つデータ項目は，割り当てられた ID によってホットデータとコールドデータと呼ばれるものに分類される．低い数字がホットデータ，高い数字がコールドデータである．ID はアクセス頻度が高いものほど低い数字が割り当てられる．問題点として，ホットデータは短時間で受信が完了するが，コールドデータは受信に長い時間を必要とする可能性がある [5]．そのような場合，クエリの生存時間が切れてしまい，データ項目をすべて受信することができない．

2.3 ホットデータとコールドデータ

アクセス頻度の高いデータはホットデータと呼ばれ，高速ストレージに保管されるが，高速ストレージは保管コストが大きい．それに対し，ほとんどアクセスされないデータはコールドデータと呼ばれ，コストの低い低速ストレージに保管される．ホットデータのアクセス頻度が低下することにより，そのデータを低速のストレージに保管することができる．低速ストレージは保管コストは小さいが，コールドデータの受信には長い時間が必要である [5][6][7]．

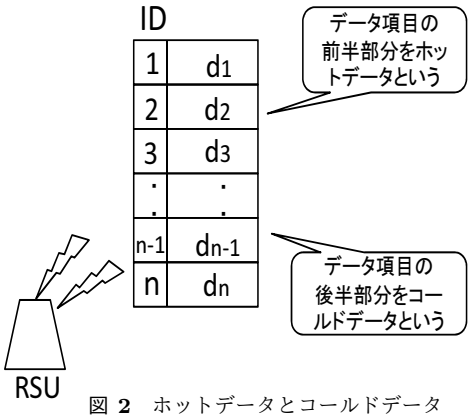

図 2　ホットデータとコールドデータ

3. 既存手法

本章では，既存のアルゴリズムについて説明する．

3.1 FCFS(First Come First Service)

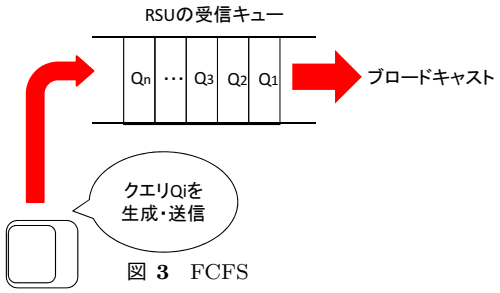

図 3　FCFS

このアルゴリズムは，RSU の受信キュー (待ち行列) において，クエリが到着した順序に従ってクエリを選択する．マルチアイテムクエリ内のすべての要求データ項目は，FCFS で等しい優先権を獲得し，それ故に，連続したブロードキャストによって提供される．

3.2 MRF(Most Request First)

MRF では，RSU 内のデータ項目のうち，受信キューの中で待機するデータ項目で，最も数が多いクエリを選択し，ブロードキャストする．マルチアイテムクエリにおいて異なる要求データ項目は，それぞれ違った人気度を持つ．故に，単一クエリ内の異なるデータ項目はこのアルゴリズムに従って，異なる働きを行う可能性がある．

4. CQS アプローチ

マルチアイテムクエリのコールドデータ受信の問題に対し，CQS アプローチと呼ばれる手法がある [11]．CQS アプローチに用いられる仮定とその具体的な手法について本

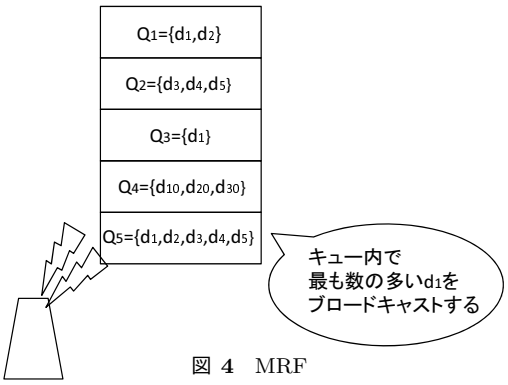

図 4　MRF

章で説明する．次に RXW と呼ばれるアルゴリズムを組み込む手法を検討する．

4.1 CQS アプローチで用いる仮定と表記法

提案手法で用いる仮定と表記法について図 5 を用いて説明する．また，それぞれの仮定で用いる計算式を各節で説明する．

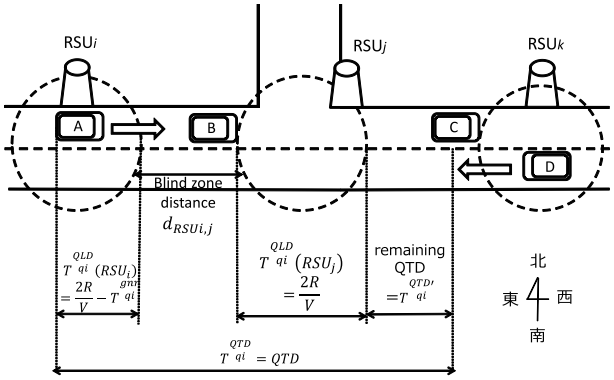

図 5　提案手法の説明

4.2 RSU で用いる仮定

4 章で用いる仮定と表記の意味を表 1 に示す．

表 1　RSU で用いる表記と仮定

表記	仮定
$DBSIZE$	RSU データベースの大きさ
d_i	クエリ内のデータ項目
R	RSU の通信半径
V	車両の平均速度
T^{dwell}_{max}	RSU の通信範囲内における車両滞在最大時間
$blindzone$	RSU の通信範囲外である RSU 間の距離
d_{avg}	$blindzone$ の平均距離
Vh_i	RSU_i の通信範囲内の車両
Q_i	車両 Vh_i が生成したクエリ

表 1 の定義に基づき，車両が RSU に滞在する最大時

間は，

$$T_{max}^{dwell} = \frac{2R}{V} \tag{1}$$

と表せる．

4.3 クエリにおける仮定

本章で用いる表記とその意味を表 2 に示す．また，それぞれの仮定で用いる計算式も本節と図 5 を用いて説明する．

表 2 クエリで用いる表記と仮定

名前	表記
$T_{q_i}^{gnr}$	クエリが生成された時間
$T_{q_i}^{QTD}$	クエリに割り当てられている生存時間
$HD(Vh_i)$	車両の進行方向で，北 (S)，南 (N)，東 (E)，西 (W) がある．
$T^{serv}(Q_i)$	RSU_i でクエリ Q_i が生成された時間
T_{qi}^{QLD}	クエリを RSU の通信範囲内でに送信可能な時間
$T_{q_i}^{QTD'}$	残りの生存時間
$T_{RSU_{i,j}}^{bz_{travel}}$	RSU_i と RSU_j 間の blind zone の車両合計移動時間
n_{hops}	クエリの生存時間内で送信可能な RSU の経路数
T_{hop}^{travel}	クエリを送信する RSU 間のホップ数
$HD(Vh_i)_{location}$	自車両の位置情報

車両 Vh_i が生成したクエリ Q_i に含まれる情報を

$$Q_i = \{Vh_i,\, d_i,\, T_{qi}^{gnr},\, T_{qi}^{QTD},\, HD(Vh_i),\, HD(Vh_i)_{location}\} \tag{2}$$

とする．

クエリが生成された時間の値は

$$T_{qi}^{gnr} = random\{0.0,\, T_{max}^{dwell}(Vh_i)\} \tag{3}$$

とする．これは，RSU の範囲内で，0 秒から，車両が RSU の通信範囲内に滞在できる最大の時間までの値で生成した時間が設定される．クエリは RSU 内のどのタイミングで生成されるかわからないため random と表記する．

生存時間 T_{qi}^{QTD} を過ぎたクエリは無効なクエリとなる．クエリ Q_i を生成した車両 Vh_i は，通信可能な最大時間 $T_{Vh_i}^{dwell}(RSU)$ の間，RSU 内に存在する．ここで，クエリ Q_i の生存時間は，車両が生成した RSU_i 内だけでなく，進行方向上に存在する複数の RSU にも送信可能な生存時間を設定する．このとき生成するクエリによって送信する RSU の範囲を決定する．したがって T_{qi}^{QTD} は，

$$T_{qi}^{QTD} \geq T_{Vh_i}^{dwell}(RSUi) \tag{4}$$

と表せる．

4.4 クエリの生存時間

クエリには 2 つの生存時間の定義がある．

（1） クエリ合計デッドライン
(QTD:Query Total Deadline)
T_{qi}^{QTD} のことで，ここでとる値を QTD 値と呼ぶ．

（2） クエリローカルデッドライン
(QLD:Query Local Deadline)
T_{qi}^{QTD} のことで，同様に QLD 値と呼ぶ．

QLD は，

$$T_{q_i}^{QLD}(RSU) = T_{Vh_i}^{dwell} = \frac{2R}{V} - T_{q_i}^{gnr} \tag{5}$$

で求められる．QLD は，車両が RSU に滞在できる最大時間からクエリが生成された時間を引いたものである．$T_{q_i}^{QLD}$ は時間の経過とともに減少し，Vh_i が RSU の通信範囲外に出たときゼロになる．$T_{q_i}^{QLD}(RSU_i)$ がクエリを生成した RSU_i 内で，RSU_i に送信する時間として足りない場合，他の隣接 RSU に転送する前に，そのクエリの QTD 値が十分に足りているかどうか，RSU が判断する．十分に転送可能な残りの QTD 値を持っていれば，隣接 RSU にクエリを転送できる．したがって，

$$T_{q_i}^{QTD'} = (T_{q_i}^{QTD} - t) \geq \{T_{q_i}^{QLD}(RSU_i) + (n_{hops} - 1) \times T_{max}^{dwell} + T_{RSU_{i,j}}^{bz_{travel}}\} \tag{6}$$

で表せる．

$T_{q_i}^{QTD'}$ は QTD 値から実際の経過時間 t を引いたものである．$T_{q_i}^{QLD}(RSU_i)$ は，クエリを生成した RSU での QLD 値である．$(n_{hops} - 1) \times T_{max}^{dwell}$ は，車両の RSU 最大滞在時間と，生存時間中にクエリが伝達可能な RSU の数を掛けたものである（クエリを生成した RSU での QLD 値を除くため，$n_{hops} - 1$ とおいている）．$T_{RSU_{i,j}}^{bz_{travel}}$ は，RSU 間の blind zone の移動距離のことである．これらを足したものより，クエリの残りの生存時間 $T_{q_i}^{QTD'}$ が大きければ，そのクエリは転送可能なクエリである．

$T_{RSU_{i,j}}^{bz\_travel}$ は

$$T_{qi}^{bz\_travel} = n_{hops} \times T_{hop}^{travel} \tag{7}$$

と表せる．

RSU_i の通信範囲内の車両 Vh_i で，RSU_i に送信できなかった不完全なクエリが，隣接する RSU_j に Vh_i によって転送された場合，クエリは RSU_j の通信範囲内で新しい QLD 値を得る．RSU_j の通信範囲内に車両 Vh_i が入ったあとに生成されたクエリが有効になる．このときの QLD 値は，クエリが持っている残りの QTD 値によって決まる．残りの QTD 値が最大値である $\frac{2R}{V}$ より小さい場合，クエリが生成された RSU の通信範囲内で決定した QLD 値が残りの QTD 値の最大値となる．したがって，転送先の RSU_j

での新しい QLD 値は，

$$T_{q_i}^{QLD}(RSU_j) = min\{\frac{2R}{V}, (T_{q_i}^{QTD} - t)\} \quad (8)$$

で求められる．

RSU_j の QLD 値である $T_{q_i}^{QLD}$ 内でクエリが RSU に送信できず，残りの QTD 値がクエリを送信するのに必要な送信時間より短い場合，クエリは送信不可能になり，

$$(T_{q_i}^{QTD} - t) < T^{serv}(Q_i) \quad (9)$$

と表せる．

CQS アプローチでは，3 つの情報を利用して QTD 値を決定する．自車両の進行方向，自車両の位置情報, 事故車両等の位置情報である．これらを用いて自車両と危険車両の位置情報を相対的に算出し，自車両に対して危険情報の重要性が高いと判断すると，QTD 値を長く設定する．ここで，図 5 より，QTD は，

$$T_{q_i}^{QTD} = \{T_{q_i}^{QLD} + d_{RSUi,j} + T_{q_i}^{QLD} + T_{q_i}^{QTD'}\} \quad (10)$$

と表すことができ，図 5 より $T_{q_i}^{QLD}$ と $d_{RSUi,j}$ を複数組み合わせたものが QTD 値と設定されている．重要性が高いと判断されたクエリは，QTD を設定する際に，この $T_{q_i}^{QLD}$ と $d_{RSUi,j}$ の組み合わせを増加することで QTD の設定を大きくし，より先の RSU まで情報を伝達することが可能となる．

4.5 CQS アプローチの手法

マルチアイテムクエリを取り扱うクエリ提供手法を要約すると以下の通りである．

（1） クエリの QLD 値を計算する
クエリの QLD 値の計算は，それが RSU 自身のクエリの場合は式 (2) を使い，他の RSU から受信したクエリの場合は式 (4) を使う．

（2） RSU が受け取ったクエリを緊急度によってサブキューに分ける
未解決クエリ Q_i は，時刻 t で十分な残り QTD 値を持つ場合，少なくとも 1 ホップ離れた隣接 RSU から完全なクエリにする機会を持つ．それは，

$$(T_{q_i}^{QTD}) \geq \{T_{q_i}^{QLD}(RSU_i) + T_{hop}^{travel} + T_{QL'(Q_i)}^{serv}\} \quad (11)$$

で表される．しかし，十分な残り QTD 値を持っていないクエリ，つまり式 (6) は，RSU_i においてそれらのクエリが完全に満たされるための最終手段である．そのようなクエリは RSU_i 自身のクエリか，他の RSU からその隣接 RSU に転送されたクエリである．したがって，このクエリは，式 (6) を満たした十分な残り QTD 値を持ったクエリよりも優先度が高くなければならない．時刻 t において 2 つのサブキューの中で我々が割り当てた RSU_i が受信したクエリは，
$Q_t^{NCQ}(RSU_i)$(Non-critical sub-queue)：
式 (6) を満たした時刻 t において QTD 値が残っているクエリである．
$Q^C Q_t(RSU_i)$(Critical sub-queue：
十分に式 (6) を満たした時刻 t において QTD 値が残っていないクエリである．

（3） クエリを選択する
時刻 T_t^{QLD} における残り QLD 値に基づいた重要なサブキュー $Q_t^{CQ}(RSU_i)$ から提供するためのクエリ選択は，提供されていないクエリの長さ $QL'(Q_i)$ と，提供時間 $T_{QL'(Q_i)}^{serv}$ に着目する．クエリ選択は以下の基準を用いる：
(1) 同じ QL' 値（未提供クエリの長さ）を持つ 2 つのクエリにおいては，低い残りの QLD 値を有する，つまり最も緊急のものを選択する．
(2) 同じ QLD 値を持つ 2 つのクエリにおいては，QL' 値が低い方を選択する．これは，最も未提供データ項目の数が少ないものが完全に提供されることを保証する．
(3) 同じ QLD 値の残りと QL' 値を持っている 2 つのクエリにおいては，クエリが必要としている提供時間 $T_{QL'(Q_i)}^{serv}$ が低いものを選択する．この手法によって，最も完成するのが容易なクエリから提供することで，コールドデータの提供に時間を掛け過ぎることを防ぎ，生存時間が切れるクエリを最小限に抑えることができる．従って，クエリ Q_i は時間 t において最も小さいクエリ選択値 $(Q_i(t))$ であるサブキュー $Q_t^{CQ}(RSU_i)$ から選択される，それは
$Query\_Selection\_Value(Q_i(t)) =$

$$\{T_{q_i}^{QLD}(RSU_i) - t + T_{QL'}^{serv}(Q_i)\} * QL'(Q_i) \quad (12)$$

で表される．

（4） 選ばれたクエリからデータ項目を選択する
次の提供サイクル中での提供のために，選択されたクエリ Q_i のデータ項目 d_i は時刻 t で最大の需要を持つ未提供の $d'(Q_i)$ のセットから選ばれる．

$$d_i = max\{P(d'_j(t))|\forall d'_j \in d'(Q_i)\} \quad (13)$$

選択されたデータ項目 d_i は提供されてないクエリ Q_i の間で最も需要があるとし， d_i はサブキューの中に存在する他のクエリによってもまた要求される．

従って，提供 d_i は帯域幅最大化問題に取り組む．

（5） クエリを満たし，サブキューから削除する
サブキュー $Q_t^{CQ}(RSU_i)$ か $Q_t^{NCQ}(RSU_i)$ の中に存在するクエリはデータ項目 d_i を要求し，未提供データ項目のセット $d'(Q_i)$ とクエリの長さ $QL'(Q_i)$ を更新する．クエリ Q_i の $QL'(Q_i)$ がゼロになったときに Q_i は完全に満たされたとし，それぞれのサブキューから除かれる．

（6） クエリの実現可能性をチェックする
クエリ実現可能性チェックは式 (5) に従って結果的に実行不可能なクエリを除く．$Q_t^{CQ}(RSU_i)$ サブキューが空ならば，クエリ選択（ステップ 3 より）とサブキュー $Q_t^{NCQ}(RSUi)$ より上記の過程を繰り返す．

（7） サブキュー $Q_t^{NCQ}(RSU_i)$ からのクエリ転送を行う
$Q_t^{NCQ}(RSU_i)$ 内のクエリ Q_i の QLD 値のクエリの期限が切れているが，クエリが $d'(Q_i)$ 内でいくつかの未提供データ項目をまだ持っているならば，クエリ転送過程が始まる．クエリ Q_i の転送は車両 Vh_i の方向と，Q_i の残り QTD 値，その方向設置されている利用可能な処理の負荷容積を持つ RSU に基づき行われる．このステップは次の通りに要約される．
1. もし RSU_j が，クエリ Q_i を提供できる方向に，RSU_i の隣接 RSU から見つけた場合，RSU_j へクエリを転送する．
2. そうでない場合，隣接した RSU_i のセット RSU から RSU_k を見つけ，その方向で最小の仕事量（最小の $N_{usr}(RSU_k)$），Vh_i がそこに到着した時，RSU_k に Q_i を転送する．

5. 提案手法

4 章で説明した CQS アプローチは，クエリの優先度を 3 つの基準によって判断するが，3 つの基準だけでは不足する可能性があると考えられる．例えば 2 つのクエリを比較した際に，クエリの生存時間，要求データ項目数，提供時間が同じであるか，またはすべて異なる場合である．これを解決するために CQS アプローチに既存アルゴリズムを組み合わせて，有効なクエリ選択を行う必要がある．

まず既存手法の FCFS であるが，これはクエリの優先度を考慮せずに連続したブロードキャストを行う手法であるから，有効であるとはいえない．2 つ目の手法の MRF は，デッドラインミスは最高でも約 30 %に抑えられているが，この手法はデータ項目の優先度を決め，ブロードキャストするデータ項目ベースの手法である．そのため，クエリベースの CQS アプローチと組み合わせるのは不可能である．そこで，我々は CQS アプローチに RXW と呼ばれるアルゴリズムを組み込む手法を提案する．RXW は，CQS アプローチで用いていない要素であるクエリの受信キュー内における待機時間を考慮するので，これらを組み合わせるとよりデッドラインミス率を抑えられることが期待できる．

データ項目	W=clock-1stALV	RXW値
Q_1={d_1,d_2}	W_1	2 × W_1
Q_2={d_4}	W_2	1 × W_2
Q_3={d_1,d_2,d_3,d_4}	W_3	4 × W_3
Q_4={d_4,d_5,d_6}	W_4	3 × W_4
Q_5={d_{10}}	W_5	1 × W_5

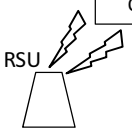

図 6 RXW

5.1 RXW

このセクションでは，CQS アプローチに組み込む RXW と呼ばれるアルゴリズムについて説明する．RXW はホットデータとコールドデータを効率良く提供するためのものである．RXW はホットとコールド両方のデータをブロードキャストする．RXW の構造は，未解決要求のそれぞれのデータのためのクエリから成る．未解決要求の数のカウントを REQcnt，未解決要求が送信された時間を 1stARV とする．このアルゴリズムは次のように動く．サーバに要求が届いたとき，要求データ項目のクエリを見つけるためにハッシュを調べる．もしクエリが見つかれば，REQcnt を 1 増やし，見つからなければ，REQcnt を 1 へ初期化する．それぞれのブロードキャストティックで，サーバは最も大きい (R × W) の値を持つデータ項目を選択する．R は REQcnt で，W は clock-1stARV で算出される．clock はブロードキャストティックの中での現在の時刻である．そのアルゴリズムは最大値を探すためにすべてのクエリを降順にソートし，上から順にブロードキャストすることによって，データ項目の最大値を見つける．それ故に，RXW アルゴリズムは他の提案手法の O(N) アルゴリズムのコストと類似している．しかしながら，RXW は未解決要求の通常の状態に基づいて決定し，データ項目のアクセス確立による要求をしないため，コールドデータの受信時間を短くすることができる．

5.2 シミュレーション評価

CQS アプローチに組み込む RXW と，FCFS をデッドラインミス率について比較し，評価を行う．デッドライン

ミス率は，(生成された総クエリ数 /デッドラインミスした総クエリ数) × 100 の式により算出する．また，1 つの RSU における純粋なデッドラインミス率を得るため，シミュレーションでは，RSU は 1 つだけであると仮定し，ランダム要素を必要最低限にするため，クエリの生存時間を一定値に定めた．

5.3 シミュレーション環境

本論文で行うシミュレーションは，一定時間ごとに生成されるクエリをどれだけ生存時間を切らすことなく処理できるかのスケジューリングを評価するためのものである．RXW と FCFS を比較する上でのシミュレーション環境について説明する．初期条件として，RSU の受信キュー内で 1ms につき 4 つのクエリが生成され，1 回のシミュレーション時間は 15s である．そして，1ms ごとに生成されるクエリを 1 個ずつ増やし，35 個になるまで繰り返す．RSU の通信範囲 50m を車両が日本国内での車両法定速度 60km/h で通過するものと考え，クエリの生存時間を 3 秒に設定した．車両が要求するデータがホットデータかコールドデータのいずれを選択するかは Zipf の法則 [12] に従い，そのパラメータは theta=0.8 である．

次に RXW についてのシミュレーション方法の説明を行う．まず 1ms に 4 つのクエリが RSU 内で生成されると，データ配信のスケジューリングは RXW の評価式を用いて RXW 値を算出し，その値に従ってクエリを降順にソートしたリストを作成する．そして，大きい RXW 値を持つものから順にそのクエリ内の未解決データ項目を提供する．このとき，DCH のチャンネル数を想定し，1ms に 4 つまでデータを配信する．そして，要求データ項目がすべて配信されたクエリは完全に満たされたとして削除する．また，要求データ項目をすべて提供する前にクエリの生存時間が切れた (デッドラインミス) クエリも削除する．

RXW と比較するための FCFS のシミュレーションも行った．FCFS は，クエリの優先度を決める処理を行わず，クエリが生成された順番に従い，クエリ内のデータ項目をブロードキャストする．

5.4 結果・評価

シミュレーションの結果について，RXW，FCFS のデッドラインミス率について図 7 に示す．横軸は 1ms で RSU が受信するクエリ数を示し，縦軸はデッドラインミス率を示す．RXW は，1ms 間に RSU が受信するクエリ数が 35 個のとき，デッドラインミス率が約 40 %である．FCFS は同じ条件でほぼ 100 %のデッドラインミス率を示している．この結果から，RXW がデッドラインミス率を抑えるという点で FCFS より優れていることがわかる．RXW の結果が単調増加になっていない点としては，RSU がデータ配信に使うことのできる帯域幅や，ホットデータとコールドデータの受信時間の違いなどランダムな要素によるものだと考えられる．また，同様のシミュレーションを複数回行ったが，いずれもこれと類似した結果が得られた．1ms 間に生成されるクエリ数が 30 個を超えた辺りから，デッドラインミス率は再び増加しているが，単純にクエリ数の増加に伴い RSU の 4 つのチャネルでは対応できなくなったと考えられる．なお今回行った実験は，クエリの生存時間は 3s に固定し，RSU に送信されるクエリ数も時間ごとに決まった数が送信される設定にしたため，この 2 点についてより詳細な設定を行う必要がある．

6. 考察

本論文の目的は，CQS アプローチの精度を改善するために RXW と呼ばれるアルゴリズムを CQS に組み合わせた場合，それが有効であるかを検証することである．実際に CQS と RXW を組み合わせたシミュレーションを行うことは，シミュレーション環境の規模を拡大することが必要となり，困難であったため，RXW と FCFS を比較するシミュレーションを行った．また，クエリの生存時間についても便宜上，本来の式を用いず RSU 内における車両の滞在時間を設定した．評価方法について，想定環境上に道路と RSU が設置されてあり，車両が RSU の通信範囲内に入ったときに送信したクエリが RSU の受信キュー内に蓄積し，その蓄積したクエリの総数に対し，生存時間を切らせてしまった割合を示すデッドラインミス率を用いる．

まず RXW のみの評価を行った結果として，「クエリ内の要求データ項目の数」と，「RSU の受信キュー内での待機時間」を考慮することが有効に働き，デッドラインミス率は，1ms に送信されるクエリ数が 35 個のときに 43 %となった．これに対し FCFS は 97 %となり，RXW の方が優れていることが分かる．他の研究で測定した FCFS，MRF，CQS のデッドラインミス率は，シミュレーション内で用いる車両数が 75 台のとき，それぞれ 70 %，65 %，35 %であった．シミュレーションのパラメータの根拠については

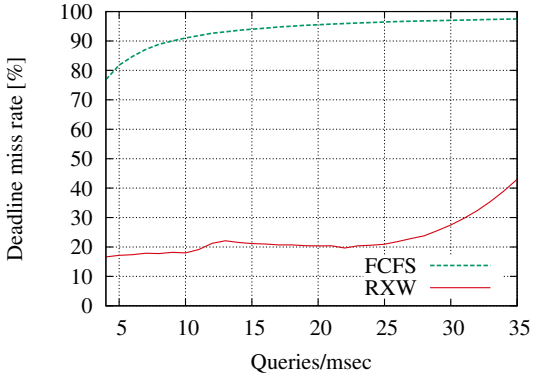

図 7 RXW と FCFS のデッドラインミス率

文献 [8] に記述されている．この文献で行われているシミュレーションと我々が行ったものは RSU に送信されるクエリ数が違う点である．我々は最高で 1ms 間に 35 個のクエリが生成される環境で行ったため，莫大な数のクエリを生存時間内に処理できるかを検証することができる． RXW はデッドラインミス率に着目した場合，いずれのアルゴリズムよりも良い結果が得られた．故に CQS のクエリ選択の判定基準にこのアルゴリズムを追加し，CQS アプローチが考慮していない要素であるクエリの受信キュー内における待機時間を考慮することにより，更に少ないデッドラインミス率が得られることが期待できる．

今後の課題としては，クエリの生存時間を実際の設定式に基づいたものを用いてシミュレーションを行うことと，実際に CQS と RXW を組み合わせた手法についてのシミュレーションを行うことが挙げられる．

参考文献

[1] 花井利通，“ITS による安全運転支援に関する将来展望”，ITS Japan(2011)．

[2] ITS Japan，“ ITS とは” http://www.its-jp.org/about/.

[3] Nitin Prabhu，Vijay Kumar，“Data Scheduling for Multi-item and Transactional Requests in On-demand Broadcast”，6th International Conference on Mobile Data Management (2005)，pp. 1-9.

[4] Jun Chen，Victor C.S. Lee，Kai Liu，“On the performance of real-time multi-item request scheduling in data broadcast environments”，The Journal of Systems and Software (2010)，pp. 1337-1345.

[5] “高速データ・アクセスのための Multi-Temperature Data Storage”，
http://pic.dhe.ibm.com/infocenter/db2luw/v10r1/index.

[6] Toru Makabe，
“Big より Hot な Cold Data”，
http://www.enterprisecioforum.com/.

[7] “Teradata Database 14”，
http://www.teradata-j.com/product/sw/teradata-14.html.

[8] G. G. Md. Nawaz Ali，Edward Chan，Wenzhong Li，“On Scheduling Real-time Multi-item Queries in Multi-RSU Vehicular Ad Hoc Networks (VANETs)”，IEEE International Conference of Advanced Information Networking and Applications Workshop (2013)，pp. 1-6.

[9] Kai Liu，Victor C.S. Lee，“On-demand broadcast for multiple-item requests in a multiple-channel environment”，Information Sciences 180 (2010)，pp. 4336-4352.

[10] Tony K. Mak，Kenneth P. Laberteaux，Raja Sengupta，“A Multi-channel VANET Providing Concurrent Safety and Commercial Services”，UCB-ITS-PWP (2005)，pp. 1-16.

[11] 細木詩織，“路車間連携による自動車の安全性向上手法の提案”，大分工業高等専門学校卒業論文 (2012)．

[12] “統計的テキスト解析 (5)〜統計法則と指標〜”，
http://www1.doshisha.ac.jp/~mjin/R/60/60.html.