Vol.2009-DPS-140 No.11
2009/9/11

頻繁なノード参加離脱に対応した 自己起点型 P2P モバイル音楽共有サービス

中川優里† 皆木沙織† 其田雅徳† 加藤由花†

ユーザの滞在する場によって，提供されるサービスの内容が異なる P2P モバイル音楽共有サービス JAMS の研究を進めている．これまで，音楽ファイルのキャッシュを利用した配信方式を検討してきたが，ノードの参加離脱が頻繁に発生する環境への適用が困難であるという問題があった．本稿では，自ノードを起点として柔軟にキャッシュ配送木を構築する手法を提案し，この問題の解決を目指す．

A Cache Management Method Adapting to Join and Leave of Nodes for P2P Mobile Music Delivery Services

YURI NAKAGAWA,† SAORI MINAGI,† MASANORI SONODA† and YUKA KATO†

We have been undertaking a research on JAMS (JAMais vu System), a music delivery system, that provides different kind of service according to location where users stay. So far, a cache management scheme have been focused on JAMS so as to provide music files efficiently to individual nodes, however, there are problems that nodes have difficulties of adapting to each location due to their mobilities. In order to overcome this problem, this paper proposes a cache management scheme that copes with their mobilities by centering each node in a network depending upon where it is.

1. はじめに

近年，iPhone や GooglePhone に代表されるようにモバイル端末の高機能化が進み，モバイルネットワークの広帯域化と相まって，モバイルネットワークサービスに対する注目が高まってきている．このような背景の下，我々は，携帯端末で構成されるアドホックネットワークによりサービスの局所性を実現する P2P 型音楽共有システム JAMS（JAMais vu System*1）の研究を進めている[1][2]．JAMS は，利用するたびに受けられる内容の異なるサービスの実現を目指しており，これにより日常生活に驚きや楽しみを感じてもらうことを目標としている．近年，ユーザの状況や嗜好にマッチしたサービスを提供するためのコンテキストアウェアネス技術に対する注目が高まってきているが[3][4]，JAMS の発想はこれとは逆であり，偶発的に変化する状況を利用したサービスの提供がその目的となっている．

JAMS ユーザは，あらかじめ自身の持つモバイル端末に音楽ファイルを蓄積しておき，通勤通学時やカフェでの待ち合わせ時間などに，同じ場に滞在する他の JAMS ユーザ（不特定多数のユーザ）との間で各自が持つ音楽ファイルを共有する．このとき，ユーザが滞在する場は時間とともに変化していくので，場を構成する JAMS ユーザも変化し，共有される音楽ファイルの種類も変化していく．その結果，滞在する場により受けられるサービスの内容が変化し，意外性，偶発性を持ったサービスの提供が実現する．

このように，モバイル端末から構成される P2P ネットワーク上で音楽配信を行うことを考えると，1 台の端末に負荷が集中することを避けるために，ネットワーク上にキャッシュを配布し，効率的な音楽配信を実現する必要が出てくる．JAMS においては，(i) 各ノードが自律的にキャッシュを管理すること，(ii) 場の変化によりキャッシュの入れ替えが発生すること，(iii) 統計的な手法ではなく適応的な手法を用いること，の 3 点を設計指針とし，キャッシュ配置の最適化を行わず，各ノードからの要求に従って適応的にキャッシュを配布する方式を提案した．これにより，JAMS に適したロバストなキャッシュ管理を実現したが，一方，ノードの参加離脱が頻繁にある状況（駅のホーム，通勤電車内など）ではキャッシュの配布が行われず，配信効率が悪化してしまうという問題があった．

そこで本稿では，自己の周辺のノードに対して，配信要求がなくてもキャッシュを能動的に配布することにより，参加離脱が頻繁に発生する状況においても効率的に音楽配信を可能となる方式を提案する．提案方式では，ファイルへのアクセス頻度などの場としての統計情報を利用せず，自己を起点としてキャッシュ配布を行うことによ

† 産業技術大学院大学
School of Industrial Technology, Advanced Institute of Industrial Technology

*1「未視感」のこと．deja vu（既視感）の逆の概念で，見慣れたはずのものが未知のものに感じられること．

り, JAMS の設計指針を満足する. キャッシュ配布によるトラフィック量の増大には, ノードのグループを形成し, グループ内にキャッシュを配布することにより対応する.

2. 関連研究

JAMS におけるキャッシュ配布の目的は，リソースの限られた携帯端末において配信負荷を軽減することにある．方式への要求条件としては，キャッシュ配布方式が場の変化に影響を与えないこと，場が頻繁に変わっても短時間でキャッシュ配布が可能であることがあげられる．本稿ではさらに，ノードの参加離脱が頻繁に発生する状況においても，効率的にキャッシュを配布する方式を考察対象としている．

P2P ネットワークやモバイルネットワーク上でのキャッシュ管理に関しては，これまでも多くの研究が行われている．代表的なものには，Winny[5]におけるキャッシュ管理方式がある．ここでは，ネットワークリソースやマシンパワー等が潤沢にあるノードにキャッシュを配布することにより，負荷分散を実現しているが，携帯端末を利用し，モビリティの高い JAMS にこのような方式を適用することは困難である．

場の変化に対応して短時間でキャッシュ配布が可能な方式としては，キャッシュの更新をある構造に従って伝搬する方式が参考になる．チェイン構造を用いた更新伝播方式[6]や，木構造を用いた方式[7]などが提案されており，これらの方式では，キャッシュの伝播によるトラフィック量削減に対する様々な工夫がなされている．しかし，リソースの限られたモバイル端末に適用することを前提にはしておらず，JAMS への適用は困難であると考えられる．

また，ノードの参加離脱が頻繁に発生する状況におけるキャッシュ配布方式に関しては，アドホックネットワークにおける頻繁なネットワーク分断に適応することを目的としたキャッシュ配布方式に関する一連の研究がある[8][9]. ここでの目的はデータの一貫性保持であり，データ更新時の再配布方法や，更新頻度の高いデータを配布する等が考察対象となっている. JAMS ではデータの一貫性に対する考慮は必要ないが, ノードの出入りの激しい状況を前提とすることから，これらの手法が参考になる．ただし，滞在する場が常に変化し，データのアクセス頻度等を予測できないことから，適応的なキャッシュ配布を実現する方式を利用する必要がある．

3. JAMS の概要

提案方式の説明の前に，本稿で考察対象とする JAMS の概要を説明する．JAMS では，コンテンツの局所性を利用したサービスを提供するので，本章ではまず,「場」による「コンテンツ」の違いについて説明する．その後，システムの構成を概観することにより，JAMS の機能を明らかにする．

3.1 コンテンツの局所性

JAMS では，コンテンツの局所性を「場」という概念で表現する．ここでは，特定の範囲内にいるノード同士がネットワークを形成し（局所性), そのネットワークのことを「場」と定義している．JAMS では，この局所性を，アドホックネットワークにより実現する．ある限られた空間の中（電車の中，大学の教室など）で，各自が所有する携帯端末によってアドホックネットワークを構築し，そのネットワーク内で音楽ファイルを共有する．ここでの考察対象は，アプリケーションレイヤであり，下位レイヤのネットワークは IEEE802.11g 等により既に構築されているものと仮定する．

このとき，ネットワーク内のノード数は常に変化するので，変化が起こるたびに「場」の特徴（ネットワークの構成要員やその特徴）は変化していく．JAMS では，刻一刻と変わっていくこの場の特徴を利用して，意外性のあるサービスを実現していくので，場の特徴が変化していく様子をユーザに提示する必要がある．ここでは，各ユーザが持つ音楽ファイルの情報を利用して場の特徴量を定義し，これをユーザに提示する．ネットワークの構成要員が入れ替われば場の特徴量も変化する．また，構成要員に変化が無くても，各々が持つ音楽ファイルが変化すればやはり特徴量は変化する．音楽ジャンルを利用した特徴量の提示例を図 1 に示す.

図 1　特徴量の提示例

3.2 システムの構成

次にシステムの構成について説明する．JAMS のシステム構成を図 2 に示す.

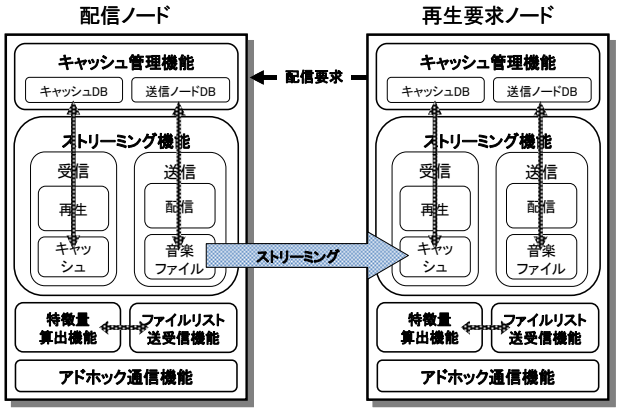

図 2　システムの構成

JAMS はピュア P2P 型のネットワークサービスであるため，全ての端末上に同一のアプリケーションが実装され，全ての端末が同等の立場で通信を行う．そのため，1 台の端末上に，音楽ファイルの送信機能と受信機能の両方が実装される（ストリーミング機能）．JAMS はまた，各端末が保有している音楽ファイルからノードの特徴量を算出し，ユーザに提示する機能，および同一の場に存在する端末から場の特徴量を算出する機能を有している．各ノードには，これらを実現するための機能（特徴量算出機能）が実装される（詳細については[10]を参照）．

キャッシュ管理方式としては，オーバレイネットワークである JAMS ネットワーク上に音楽ファイルのキャッシュを配布する機能を有している. そのため, 各端末には，自身が配布したキャッシュファイルを管理する機能（送信ノード DB）と，自身が保持するキャッシュファイルを管理する機能（キャッシュ DB）が実装される．

3.3 キャッシュ管理方式

JAMS の特徴は，携帯端末による効率的な音楽配信を実現するために，JAMS ネットワーク上に音楽ファイルのキャッシュを配布することにある. ここでの設計指針は，（i）各ノードが自律的にキャッシュを管理すること（携帯端末での利用を前提に負荷が集中しない方式を採用する），（ii）場の変化によりキャッシュの入れ替えが発生すること（キャッシュの配布が場の特徴量に影響を与えない方式を採用する），（iii）統計的な手法ではなく適応的な手法を用いること（場を構成しているファイルの種類は時間，場所とともに変化するので，ファイルのアクセス頻度等を事前に推測できない）の 3 点である．

これらの指針に基づき設計した JAMS におけるキャッシュ管理方式の特徴を以下に示す．

- キャッシュの配布 : 能動的なキャッシュ配布を行わず，他のノードから音楽ファイルの配信を受けた場合にのみ，そのファイルをキャッシュとして蓄積する．
- キャッシュの消去 : あらかじめ設定されたタイムアウト時間を超えるまで配信要求がなかった場合は，自動的にキャッシュを消去する．
- キャッシュの置き換え : タイムアウトまでの時間が短いファイルから順にキャッシュを置きかえる（LRU）．
- キャッシュの管理 : ある音楽ファイルのキャッシュを保持するノードは，オリジナルファイルを持つノードを起点とした木構造を取り，これをオリジナルファイルを持つノードが管理する．
- キャッシュの選択 : あるノードへの配信要求数がしきい値を超えた場合に，要求ノードに対して自ノードが管理するキャッシュを紹介し，アクセスを分散させる．キャッシュを利用する順番は，オリジナルファイルを持つノードからの木構造の階層が少ない順とする．

3.4 既存方式の問題点

既存方式では，キャッシュの配布に統計的な手法を用いないことにより，場の変化に影響を受けないロバストな方式を実現した．しかし，駅のホームなど，ノードの出入りが頻繁に発生する「場」においては，以下の問題が発生した．

- 能動的なキャッシュ配布を行わないため，ノードの参加離脱が激しい場ではキャッシュの配布がほとんど行われない．その結果，人気のある音楽ファイルにアクセスが集中した場合に配信に失敗する確率が高くなる．
- キャッシュの管理は，オリジナルデータを保持するノードがキャッシュ配送木を保持することで実現している. そのため, オリジナルノードへの負荷が高くなる．

本稿ではこれらの問題の解決を目指し，自己起点型のキャッシュ配布方式を提案する．

4. 自己起点型キャッシュ配布方式

既存方式の問題点を解決するために，本稿では自己起点型のキャッシュ配布方式を提案する．以下，提案方式について詳述する．

4.1 方式の概要

方式の提案にあたっては，以下の設計指針を策定した．

- ノードの参加離脱が頻繁に発生する状況においてもキャッシュの配布が行われること．
- ファイルへのアクセス頻度等，場としての統計情報が不要であること．
- キャッシュ配布のためのトラフィック量を抑制すること．

これらの指針を満たすために，ネットワーク的に自己の周辺に存在する複数ノードから成るグループを形成し，そのグループ内でキャッシュの配布を行う方式を設計した．このとき，グループ内でのキャッシュ配布のためのトラフィック量を削減するために，グループごとに親ノードを決定し，各ノードが自己を起点として，可能な限り視聴要求の高いキャッシュを配信することとした．その他，キャッシュの消去，親ノードによるキャッシュ管理方法等は既存方式と同様とする．

4.2 ノードのグループ

まず，提案手法の一番の特徴である，ノードグループについて説明する．JAMS に参加するノードは，キャッシュを共有するためのグループの構成要員となり，グループ内でキャッシュを共有することになる．そのため，全てのノードはそれぞれに個別の ID（UID）とグループの ID（GID）を保有することとした．携帯電話の場合は電話番号，その他の端末の場合はネットワークアダプタのシリアル番号や MAC アドレス等を UID として利用することができる．さらに各端末は，UID に，各自が所有している音楽ファイルの情報やタイムスタンプなどを持たせている．具体的には，グループは JAMS ネットワークにおいて 1 ホップ以内にあるノードをメンバーとして構成され，グループを構成するノード数を 6 個以内としている．グループには親ノードが存在するが，これはグループが生成された後にノードの保有する属性値を比較することにより決定する．

4.3 処理の流れ

次に，グループ生成時の処理，グループ参加時の処理，ストリーム送受信時の処理について説明する．

4.3.1 グループ生成時の処理

ある場にノード A を含む複数のノードが存在する場合のグループ生成時の処理を図 3 に示す．具体的な処理の流れは以下のようになる．

1. グループ生成に必要なノードの UID の取得
   ノード A は周囲のグループに所属していないノードに UID を要求する．要求を受けたノードは UID を送信，ノード A は UID を取得する．
2. 親ノードの決定と GID の生成
   ノード A は取得した UID と自分の UID を合わせ，その中で音楽情報数が最も多く，タイムスタンプが最も新しい UID を親ノードとして決定する．さらにそれぞれの UID と親ノード情報を持たせた GID を生成する．
3. 親ノードへ GID を送信
   GID を親ノードへ送信する．ノード A が取得した UID などの情報は削除する．
4. 子ノードのキャッシュ情報を親ノードが収集，管理
   親ノードは子ノードにキャッシュを要求し，子ノードはキャッシュを親ノードに送信する．親ノードは子ノードと自ノードのキャッシュを管理する．
5. キャッシュの配布
   親ノードは子ノードから収集したキャッシュをグループ内のノードの特徴量に応じてそれぞれの子ノードへ配布する．キャッシュはそれぞれの子ノードが持っているキャッシュ以外のものが配布される．

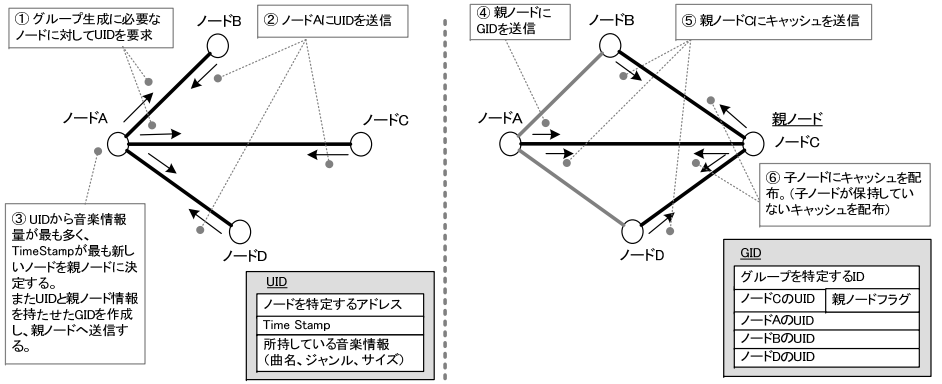

図 3　グループ生成時の処理

4.3.2 グループ参加時の処理

新たにグループに参加するときの処理を図 4 に示す．具体的な処理の流れは以下のようになる．

1. 近隣のノードからグループの親ノード情報を取得
   ノード A は近隣のノードから参加しているグループの親ノード情報を要求する．要求された近隣のノードはノード A にグループの親ノード情報を送信し，ノード A はグループの親ノード情報を取得する．
2. 親ノードへのグループ参加の許可を要求
   ノード A は取得したグループの親ノード情報を用いてそのグループの親ノー

ドにグループに参加の許可を要求する．親ノードは要求を受信し，現在のグループ内のノード数から参加可能か判断し，その結果をノード A に送信する．

3. グループへの参加
ノード A はグループに参加できない場合は再度, 他のグループに参加しているノードに対してグループの親ノード情報を要求し，1．2．と同じ処理を行う．グループへ参加できる場合は親ノードに UID を送信．UID を受信した親ノードはそれを GID の UID テーブルに追加する．
4. キャッシュの配布
親ノードはノード A に対してキャッシュを要求し，ノード A は親ノードにキャッシュを送信する．親ノードがノード A のキャッシュを取得すると，他の子ノードに対してノード A のキャッシュを配布する．ノード A にはノード A 以外のキャッシュはグループ内のノードの特徴量に応じて配布される．

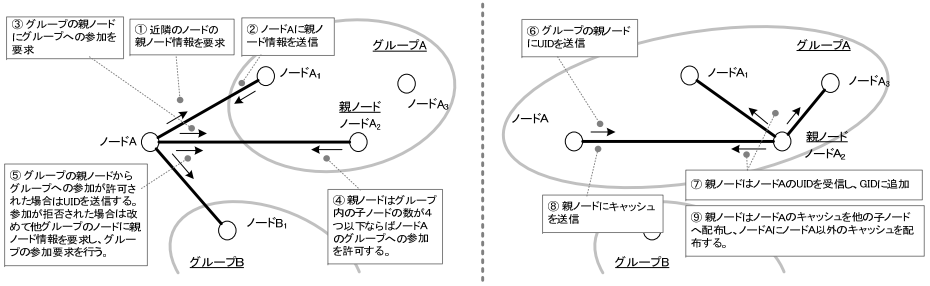

図 4　グループ参加時の処理

4.3.3 キャッシュ管理情報を収集する処理

グループ間でキャッシュの管理情報を交換するときの処理を図 5 に示す．具体的な処理の流れは以下のようになる．

1. 他のグループのノードから親ノード情報を取得
グループ A の親ノードはグループ A の外にいるノードの親ノード情報を要求する．要求されたノードは自分が参加しているグループ B の親ノード情報を送信し，グループ A の親ノードはグループ B の親ノード情報を取得する．
2. グループ B の親ノードに GID を要求
グループ A の親ノードはグループ B の親ノードに GID を要求する. 要求されたグループ B の親ノードはグループ A の親ノードへ GID を送信し，グループ A の親ノードはグループ B の GID を取得する．
3. グループ B の親ノードへグループ A の GID を送信
グループ A の親ノードはグループ B の親ノードへグループ A の GID を送信し，グループ B の親ノードはグループ A の GID を取得する．

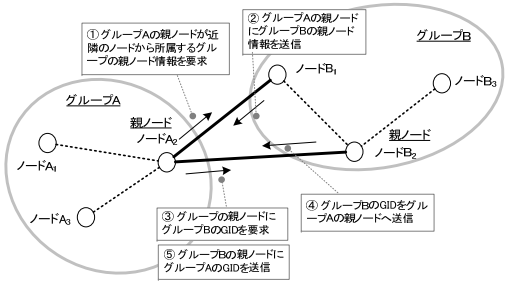

図 5　キャッシュ管理情報を収集する処理

4.3.4 親ノード離脱時の処理

親ノードが離脱し，代わりのノードが親ノードになるときの処理を図 6 に示す．具体的な処理の流れは以下のようになる．

1. 親ノードの離脱時に GID を更新，次の親ノードへ送信する
親ノードはグループから離脱する際に GID から自ノードの UID を削除する．また音楽情報数が最も多くタイムスタンプが最も新しい UID を次の親ノードとし GID の内容を親ノードの UID に親ノードフラグを持たせて更新する．
更新した GID は次の親ノードに送信される．
2. 子ノードは親ノード情報を更新
新しい親ノードは子ノードに新しい親ノード情報を送信する．子ノードは親ノード情報を更新し，更新完了したことを親ノードに通知する．
3. 子ノードのキャッシュ情報を親ノードが収集，管理
親ノードは GID を用いて親ノードが所有していないキャッシュを子ノードに要求する．子ノードは要求されたキャッシュを親ノードに送信する．親ノードはグループ内の子ノードと自分のキャッシュを合わせて管理する．
4. キャッシュの配布
子ノードから収集したキャッシュはグループ内のノードの特徴量に応じてそれぞれの子ノードへ配布される．キャッシュの配布はそれぞれの子ノードが持っているキャッシュ以外のものが配布される．

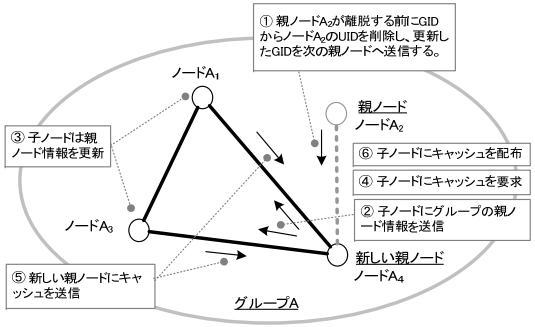

図 6　親ノード離脱時の処理

4.3.5 子ノード離脱時の処理

子ノードが離脱するときの処理の流れは以下のようになる．

1. 子ノードがグループを離脱
   子ノードはグループを離脱する際に，親ノードに離脱することを通知する．
2. 親ノードは GID を更新
   子ノードがグループを離脱する通知を受けた親ノードは GID から子ノードの UID を削除する．
3. 親ノードはグループ外の親ノードに GID の更新データを送信
   親ノードは離脱した子ノードの UID を削除した GID の更新データをグループ外の親ノードに送信する．
4. グループ外の親ノードはグループ A の GID を更新
   グループ A の GID の更新データを受信したグループ B の親ノードは，保持していたグループ A の GID に更新データを使って GID を更新する．

4.3.6 ストリーム送受信時の処理

ストリームを送受信するときの処理の流れは以下のようになる．

1. 子ノードが親ノードに音楽ファイルを要求
   グループ A の子ノードは親ノードに指定した音楽ファイルを持っているノードの所在を要求する．
2. 親ノードは GID から音楽ファイルを持つノードを検索
   グループ A の親ノードは要求された音楽ファイルを持っているノードを特定するため，自グループの GID と他グループの GID から該当する音楽ファイル名を持つ UID を検索する．さらに検索した UID からノードを特定する．
3. 親ノードは子ノードにノードを紹介
   親ノードは音楽ファイルを所有するノードを子ノードに紹介する．
4. 音楽ファイルを所有するノードへストリーム送信を要求
   子ノードは音楽ファイルを持つノードへストリーミングの開始を要求する．
5. 要求先のノードはストリーム送信できるか要求元へ通知
   要求先のノードは要求された音楽ファイルを所有しているか確認する．所有している場合はストリームを要求元のノードへ送信する．所有していない場合は要求元のノードへストリーム送信不可能である通知を送信する．
6. 要求元のノードは要求先のノードから要求の結果を受信
   要求先のノードのストリーム送信が可能ならば送信元はストリームを受信する．
   要求先のノードのストリーム送信が不可能ならば送信元は送信不可能であるという通知を受信する．
   要求先が送信不可能である場合や，一定時間経ってもストリーミングが開始されない場合は再度，グループ A の親ノードに音楽ファイルを要求する．
7. ストリームの受信完了後，親ノードに受信完了の通知を行う
   ストリーム受信が完了するとそのノードは親ノードに受信完了の通知を行う．
8. GID を更新
   親ノードは子ノードが新たに所有したキャッシュとタイムスタンプを GID に記録し更新する．GID の更新データは他のグループの親ノードに送信される．

5. 評価

ここでは，キャッシュ配布の安定性について，JAMS における既存方式と自己起点型キャッシュ配布方式を比較する．評価尺度しては，対象ノードが所有するキャッシュを他のノードがストリームとして要求できる平均時間 Tc[sec]を用いる．本章では，対象となる場にノードが滞在する平均時間を Ts[sec]，キャッシュのタイムアウトの平均時間を Tout[sec]とする．

5.1 既存方式

既存方式では，音楽ファイルを所有するノードが親ノードとなり，音楽ファイルのストリーム配信、キャッシュの管理やキャッシュの配布先ノードの紹介などを行っている．この親ノードが場から離脱すると新たに他のノードがキャッシュのストリーム受信を要求することができなくなる．従って，他ノードがキャッシュのストリーム受信を要求できる平均時間 Tc[sec]は以下のように示される．

$$Tc[\mathrm{sec}] = Ts[\mathrm{sec}] \quad \cdots \ (1)$$

5.2 自己起点型キャッシュ配布方式の場合

自己起点型キャッシュ配布方式では，グループを生成し，グループ内の親ノードがキャッシュの管理を GID で行っている．親ノードが離脱する時には GID を次の親ノードに渡すことで，引き続きキャッシュのストリーム受信の要求を受け適切にノードを紹介することができる．また予めグループ内でキャッシュを共有しているので，音楽ファイルを所有するノードが離脱してもその親ノードを通して他の子ノードにキャッシュのストリーム受信の要求を行うことができる．また、キャッシュのストリーム受信の要求ができる平均時間 Tc[sec]はグループ内でキャッシュが共有される前に音楽ファイルを所有するノードが離脱してしまうまでの平均時間 Ts[sec]、またはキャッシュがグループ内で共有された後、他のノードからキャッシュに対する受信要求がなくキャッシュがタイムアウトを迎えて消滅するまでの平均時間 Tout[sec]が考えられる．従って，他のノードがキャッシュのストリーム受信を要求できる平均時間 Tc[sec]は以下のように示される．

$$Ts[\mathrm{sec}] \le Tc[\mathrm{sec}] \le Tout[\mathrm{sec}] \quad \cdots (2)$$

5.3 考察

以上より，他ノードがキャッシュをストリーム受信できる時間は，既存方式では式(1)，自己起点型キャッシュ配布方式では式(2)となる．これは他ノードが受信要求できる時間が既存方式では最大でも音楽ファイルを所有するノードが離脱するまでの時間であるのに対し，自己起点型キャッシュ配布方式では音楽ファイルを所有するノードが離脱した後もキャッシュのストリーム受信の要求がなくなりタイムアウトして消滅する時間までであることを意味している．つまり，提案方式では音楽ファイルのキャッシュをもつノードが存在する限り，他ノードはキャッシュをストリーム受信することが可能である．

キャッシュのストリーム受信を要求できる時間が増すことは，他ノードに対してより安定してストリーム配信を提供できることを意味する．従って，既存方式でユーザの配信要求に対して失敗率の高かったノードの参加離脱が頻繁に行われる状況でも，自己起点型キャッシュ管理方式では，より安定したストリーム配信を提供できると考えられる．

6. おわりに

本稿では，自己を起点として柔軟にキャッシュ配送木を構築し，配信要求がなくてもキャッシュを積極的に配布することにより，参加離脱が頻繁に発生する状況においても効率的な音楽共有が可能となる手法を提案した．今後，シミュレーション実験より，提案手法の有効性を検証していく予定である．また，実環境を想定したサービスの提案を行うために，より柔軟なキャッシュ管理方式を検討していく予定である．

参考文献

1) 柴田浩明，富澤智，遠藤博樹，加藤由花．コンテンツの局所性に着目した P2P 型音楽配信システム．情報処理学会 DPS ワークショップ 2007, pp.37-42, 2007.
2) 柴田浩明，富澤智，遠藤博樹，渡部寿基，大城裕史,加藤由花．場の特徴量を利用した意外性のあるモバイル音楽共有システム．情報処理学会 DPS 研究会, Vol. DPS-133, pp.25-30, 2007.
3) D.J. Corbett and D. Cutting, "AD LOC: Collaborative Location-based Annotation," IPSJ Journal, Vol.48, No. 6, pp.2052-2064, 2007.
4) Y. Nakanishi, K. Takahashi, T. Tsuji and K. Hakozaki, "iCAMS: A Mobile Communication Tool using Location and Schedule Information," IEEE Pervasive Computing, Vol.3, No.1, pp.82-88, 2004.
5) 金子勇（編）, "Winny の技術," アスキー, 2005.
6) Z. Wang, S.K. Das, M. Kumar and H. Shen, "An Efficient Peer-to-Peer Scheme Algoritm for P2P Systems, "Computer Communications, pp.1106-1115, 2007.
7) 渡辺俊貴，原隆浩，木戸裕樹，中通実，西尾章治郎, "P2P ネットワークにおける木構造に基づく複製更新伝播法," 情処論, Vol.48, No.2, pp.527-538, 2007.
8) 林秀樹，原隆浩，西尾章治郎, "アドホックネットワークにおけるトポロジ変化に適応した複製の再配置, " 情処論, Vol.47, No.1, pp.2-14, 2006.
9) L. Yin and G. Cao, "Supporting Cooperative Caching in Ad Hoc Networks," IEEE Trans. on Mobile Computing, Vol.5, No.1, pp.77-89, 2006.
10) 富澤智，大城裕史，加藤由花, "モバイル音楽共有システム JAMS における特徴量表示方式," 情報処理学会 DPS 研究会, DPS-134, pp.145-150, 2007.