トレーサブルなP2Pレコード交換システムにおける物理構成方式に関する検討

李　峰栄† 飯田　卓也† 石川　佳治††

† 名古屋大学大学院情報科学研究科
†† 名古屋大学情報連携基盤センター
E-mail: †{lifr,iida}@db.itc.nagoya-u.ac.jp, ††ishikawa@itc.nagoya-u.ac.jp

あらまし P2P 環境において交換されるデータの信頼性を確保するために，我々はトレーサブルな P2P レコード交換システムのアーキテクチャを提案し，その開発を進めている [10]．提案したシステムの枠組みは，トレース処理を実現可能とする点で基本的な要求を満たすものであったが，信頼性や効率の面で改善の余地が大きい．そのため，本論文では，システムの物理構成方式に着目し，その拡張を行うことでそれらの問題の解決を図る．

キーワード P2P データベース，レコード交換，トレーサビリティ，物理構成方式

On Physical Organization of a P2P Record Exchange System

Fengrong LI †, Takuya IIDA†, and Yoshiharu ISHIKAWA††

† Graduate School of Information Science, Nagoya University
†† Information Technology Center, Nagoya University
E-mail: †{lifr,iida}@db.itc.nagoya-u.ac.jp, ††ishikawa@itc.nagoya-u.ac.jp

Abstract To ensure reliability among the data exchanged in P2P networks, we proposed a record exchange system based on database technologies [10]. While the proposed system framework satisfies the basic requirements by supporting tracing facilities, it still has reliability and performance problems. In this paper, we extend its phyisical organization scheme to solve the problems.

Key words P2P databases, record exchange, traceability, physical organization

1. はじめに

近年，ピア・ツー・ピア (P2P) 技術はデータ交換にとって重要な技術の一つとなっている．P2P ネットワークはサーバとなる計算機を必要とせず，従来のサーバ・クライアントモデルのネットワークと比べて利便性などの点で優れており，その普及が急速に進んでいる．例えば，Freenet，Gnutella，Napster，ICQ，Seti@home，LOCKSS など多くのシステムは P2P 技術を使用している [12]．

一方，科学技術分野などにおける情報交換においては，情報の信頼性は極めて重要である [3]．P2P ネットワークは特定多数のコンピュータ（ピア）が相互に接続され，直接ファイルなどの情報を送受信するインターネットの利用形態であるため，データの複製や変更が情報交換・流通中にネットワークの各所で発生する．そのため，ネットワークを介して入手した情報の信頼性を保障するのは容易ではない．そのため，P2P ネットワーク上の情報交換においても，データの出所の追跡や保障をする**系統管理**（data provenance, lineage tracing）は重要な課題であるといえる．バイオインフォマティクスなどの科学技術分野のデータ共有・注釈のみならず，ディジタルコンテンツの流通など，さまざまな応用が考えられ，信頼性の確保のために，トレーサビリティの機構は不可欠である．

このような背景から，我々の研究グループでは，P2P ネットワークにおける情報交換により得られたデータの信頼性を確保するための基盤技術の確立を目指して，流通するデータの**トレーサビリティ**（traceability）を実現するためのレコード交換システムのアーキテクチャを提案している [10]．各ピアでは，レコード（タプル）形式のデータの集合が管理され，他のピアとの交換が自律的に行われる．レコードの交換・変更の履歴は，データとともに各ピア上のリレーショナルデータベースシステムを用いて分散管理される．トレーサビリティに関する追跡処理が発生した際には，再帰的問合せを P2P ネットワーク上で分散実行して情報を収集するという方式で実現を図る [7], [8]．

この提案したシステムの枠組みは，P2P ネットワーク内のピアの参加・退出などを考慮してはいないが，トレース処理を実現可能とするための基本的な要求を満たすものである．信頼性

や効率の面で改善すべき点が多いことから，本稿では，P2P レコード交換システムの物理レイヤの構成方式を拡張し，それらの問題の解決に重点をおいて議論する．

本稿の構成は以下のようになる．2. で基本的なトレーサブルな P2P レコード交換システムの枠組を述べ，3. ではすでに提案した P2P レコード交換システムの問題点を検討し，4. で提案したシステムの物理レイヤにおける物理構成方式の改善を説明する．5. で関連研究を述べる．最後に 6. で結論と今後の課題をまとめる．

2. トレーサブルな P2P レコード交換システム

2.1 トレーサブルな P2P レコード交換システムの構成

本節では，論文 [10] において提案した P2P ネットワークにおける**レコード交換システム**の構成を説明する．

ここで，**レコード**とは，属性と値からなるタプル構造のデータを意味しており，そのスキーマ（属性集合と各属性の役割）は P2P ネットワーク上で共有されているとする．各ピアには，同じ構造を持ったレコード集合が存在するが，その内容は必ずしも同じではない．各ピアは自律的に個々のレコード集合を管理しており，他のピアのレコード集合と内容の同期などは特に行わないものとする．基本的には，各ピアに 1 人のユーザが対応し，そのユーザが興味あるレコードの集合を保持すると想定する．各ピアは，P2P ネットワーク上の他のピアに対して条件を指定した問合せを行い，問合せ結果のレコードを自身が管理するレコード集合に選択・追加することが可能であるとする．

しかし，P2P ネットワークを介して取得されたデータの信頼性を確保するために，トレーサビリティが重要である．レコードの出所の把握，重複したレコードの検出，レコードの行き先の取得，レコード更新処理の追跡などのトレース機能を実現するために，我々は 3 層から構成されるトレーサブルなレコード交換システムを提案した．

次は小説の例を用いてこの 3 層の P2P レコード交換システムの構成を示す．

（1） **ユーザレイヤ**：ユーザに対するインタフェースとしての問合せ機能や追跡処理の機能を提供する．例えば，図 1 に示すように，ある時刻において，ピア A，B，C はそれぞれこのような小説の集合を保存しているとする．

ピア A

title	author
t1	a2
t7	a6

ピア B

title	author
t1	a2
t5	a5

ピア C

title	author
t1	a1
t3	a3

図 1　各ピアにおけるレコード集合 Novel

ユーザは必要に応じて P2P ネットワーク上に問合せを発行し，関心があるレコードの集合を取得する．ユーザは取得したレコード集合の内容を吟味し，選択したものをシステムに登録する．そのため，実際には，ピア A の一番目のレコードはピア B からもらったものであるなどといった状況が発生する．

（2） **論理レイヤ**：このレイヤは，P2P ネットワーク上に分散したデータおよび履歴情報を仮想的なビューにより表現する役割を持つ．Data ビュー，Change ビュー，Exchange ビューという仮想的な 3 つのビューが提供され，これらを用いて，トレース処理の問合せを簡単に記述できる．

図 2 の Data[Novel] リレーションは，先に示した図 1 に含まれる各ピアのレコード集合 Novel の全レコードを統合したビューを表す．なお，各リレーションに現在存在している値だけでなく，過去に存在していたが，修正・削除の結果，ユーザの立場からは存在していないレコードについても情報が含まれる．Data[Novel] リレーションの左側の 2 つの属性はユーザレイヤのレコードの属性を表している．残りの属性は管理用の属性である．peer 属性にはそのレコードを所有するピアの情報が記述され，id 属性には各ピアがレコードに付与する論理 ID が収められる．

title	author	peer	id
t1	a2	A	#A011
t7	a7	A	#A040
t7	a6	A	#A041
t1	a1	B	#B030
t1	a2	B	#B032
t5	a5	B	#B035
t1	a1	C	#C005
t3	a3	C	#C055

図 2　ビュー Data[Novel]

図 3 に示す Change[Novel] ビューは，新規作成・修正・削除に関する履歴を保持する大域的なビューである．peer 属性にはそのレコードを所有するピアの情報が記述され，from_id 属性は修正前のレコード ID を，to_id 属性は修正後のレコード ID を表し，time 属性は修正時のタイムスタンプを表す．from_id 属性が空値（–）である場合はローカルな新規作成レコードの挿入を表し，to_id 属性が空値である場合はレコードの削除を表す．このように，履歴情報と論理的な ID を用いて，トレーサビリティのために必要となる情報を表現する．

peer	from_id	to_id	time
A	–	#A040	4/10/08
A	#A040	–	8/10/08
A	#A040	#A041	8/10/08
B	–	#B035	8/26/08
B	#B030	#B032	4/20/08
C	–	#C005	2/2/08

図 3　ビュー Change[Novel]

なお，他のピアからコピーしてきたデータを最初にデータ集合に挿入するときには履歴データには記録せず，その後の修正・削除履歴のみを記録する．コピーされたデータの追加に関する履歴は，後述の Exchange ビューを利用して取得できるからである．

図 4 に示すリレーション Exchange[Novel] はピア間のレコードの交換に関する情報を保持する大域的なビューであ

る．from_peer, to_peer はレコードのコピー元，コピー先をそれぞれ表し，from_id, to_id は各レコードのそれぞれのピアにおける論理 ID を表す．time 属性はタイムスタンプ情報であり，コピー先のピアにレコードがコピーされた時刻を表す．

from_peer	to_peer	from_id	to_id	time
C	B	#C005	#B032	4/20/08
B	A	#B032	#A011	3/2/09
A	E	#A011	#E017	7/5/08

図 4　ビュー Exchange[Novel]

（3） **物理レイヤ**：物理レイヤでは，データおよび履歴情報が各ピアに分散していることを明示的に意識する．各ピアは，自身に関連する情報のみを個別に管理する．このレイヤでは，論理レイヤの仮想的なビューに対する問合せが，自律的で分散したピアの協調に基づいて実行される．各ピアは，自分自身のデータと履歴情報を Data，Change，From，および To の 4 つのリレーションを用いて表現する．

図 5, 6 は，図 2, 3 に示した論理レイヤのリレーション Data[Novel] および Change[Novel] に対応する．ピア A における物理レイヤのリレーションを示す．具体的には，ピア A に関するタプルのみを抜き出したものが内容となる．なお，図 5 と図 6 では，peer 属性の値が A であることは明らかであるため，この属性は削除している．

title	author	id
t1	a2	#A011
t7	a7	#A040
t7	a6	#A041

図 5　ピア A の Data[Novel]

former_id	current_id	time
–	#A040	4/10/08
#A040	–	8/10/08
#A040	#A041	8/10/08

図 6　ピア A の Change[Novel]

論理レイヤのリレーション Exchange[Novel] に関しては，その内容をピア間で分散して管理する．具体的には，レコードの複製元にさかのぼって追跡する場合に利用される From[Novel] リレーションとレコードの複製先に向かって追跡する場合に利用される To[Novel] リレーションに分けられる．図 7 と図 8 はピア A に関する例であり，それぞれピア A が受け取ったレコードに関する情報と自身のレコードをどのピアに渡したかという情報を保持している．

id	from_peer	from_id	time
#A011	B	#B032	3/2/09

図 7　ピア A における From[Novel]

id	to_peer	to_id	time
#A011	E	#E017	7/5/08

図 8　ピア A の To[Novel]

他のピアにも同じリレーションを保存されるが，内容は異なっている．各ピアは，自分自身に関連したデータのみを管理すればよいため，メンテナンスコストが低いという利点がある．物理レイヤについて問合せを直接表現することが容易ではないことから，論理レイヤのビューを利用する．次節で，問合せ記述の例を示す．

2.2　問合せの記述

情報を追跡するには再帰的な処理が求められることから，datalog [2] を用いた記述を試みる．datalog は，近年ではネットワーク上での問合せ処理などで，新たな応用が見られる [11]．

英数大文字により変数名を表し，下線（_）により無名の変数を表す．最後の行に示すルールが問合せを表す．

問合せ 1：ピア A が保持する，タイトル t1，著者 a1 というレコードに対し，その元々のデータを最初に作成したピアを知りたいとする．そのような問合せは以下のように記述できる．

```
BReach(P, I1) :- Data[Novel]('t1','a1', 'A', I2),
                 Exchange[Novel](P, 'A', I1, I2, _)
BReach(P1, I1) :- BReach(P2, I2),
                  Exchange[Novel](P1, P2, I1, I2, _)
Origin(P) :- BReach(P, I),
             NOT Exchange[Novel](_, P, _, I)
Query(P) :- Origin(P)
```

問合せ 2：ピア C が保持する著者 a1 により書かれた t1 という小説のレコードをピア A がコピーしたかを調べよ．

```
Reach(P, I1) :- Data[Novel]('t1', 'a1', 'C', I2),
                Exchange[Novel]('C', P, I2, I1, _)
Reach(P, I1) :- Reach(P1, I2),
                Exchange[Novel](P1, P, I2, I1, _)
Check(I) :- Reach('A', I)
Query(I) :- Check(I)
```

問合せ 3：ピア C が保持する著者 a1 により書かれた t1 という本のデータをコピーしたピアをすべて列挙せよ．このような前向きの追跡を行うには，あるピアにおいて提供された情報をどのピアがコピーしたかという情報を把握する必要がある． 以下のように記述できる．

```
Reach(P, I1) :- Data[Novel]('t1', 'a1', 'C', I2),
                Exchange[Novel]( 'C', P, I2, I1, _)
Reach(P, I1) :- Reach(P1, I2),
                Exchange[Novel](P1, P, I2, I1, _)
Query(P) :- Reach(P, I1)
```

前節で述べたように，仮想的な 3 つのビューのみを利用して，さまざまな問合せを記述できる．4. ではこの 3 つの問合せの効率化処理を示す．

このレコード交換システムでは，各ピアは，P2P レコード交換ネットワークの一貫性を満たす上で必要な責任を十分果たすことを想定している．しかし，実際にはいくつかの問題点がある．詳しくは次節で述べる．

3.　これまでの枠組みにおける問題点

以前提案した P2P レコード交換システムのフレームワークでは，前節に述べたように，P2P ネットワーク上でのすべての

ピアが協調してレコード交換システム機構を実現することを想定していた．しかし，実際の P2P レコード交換システムにおいては，いくつかの問題が生じる．

- ピアの参加・退出の問題：通常，P2P ネットワークでは，不特定のピアが必要に応じてネットワークに参加・退出を行い，緩やかに情報を共有する．レコード交換ネットワークに新たにピアが参加する場合，そのピアは既存のピアに対して接続するなのど，通常とられるアプローチに従うものとする．一方，本アーキテクチャでは，ピアの退出の問題がより重要となる．本研究では，各ピアは無責任な匿名のユーザではなく，情報流通において応分の責任を果たすようなある程度の信頼がおける組織を想定している．しかし，ネットワークの分断や，ピア自体の障害などにより，突然ネットワークから退出してしまった場合，そのピアやそのピアを経由して得られたレコードに関するトレース処理が困難となる．
- 問合せ処理の効率化の問題：これまでの提案では，各ピアは自分自身に関するデータおよび履歴情報だけを持つものとし，最小の情報を管理するものとしていた．必要な情報は，必要になった時点でネットワーク上のピアから収集するアプローチをとる．このアプローチはメンテナンスのコストが低いが，問合せを処理するために，関連するすべてのピアに要求を伝播する必要があるため，問合せ処理のコストが一般に大きい．

他の問題（例：レコード検索）もあるが，本稿では，P2P ネットワークの障害対策と問合せ処理の効率化という 2 つの問題を重点を置き，物理レイヤにおける構成方式の改善について議論する．

4. 物理レイヤの構成方式の改善

4.1 複製を用いた信頼性の確保

ユーザレイヤと論理レイヤの構成方式には変更を行わず，物理レイヤに新たに情報を追加することにより，障害に対応可能となるようなシステムのフレームワークを提案する．

従来，障害によるデータの破壊や損失を防止する手法としてデータベースの**複製**（replication）を行ってデータを保護する方法がある．例えば [6] では，複製元であるプライマリデータベースが行ったデータ処理のログ等を，複製先のバックアップデータベースに転送し，同じデータ処理を行わせることで，データベースごとのデータ複製を行っている．データの損失を最小限にするために，データ処理毎にプライマリデータベースとバックアップデータベースで同期を取るための複製処理が必要となる [6]．

P2P ネットワーク上の各ピアに同期的な複製機能を導入することで，障害時にデータの損失なく，トレース処理を継続することが可能となる．以下ではそのアイデアについて述べる．

4.1.1 ペアリング方式

第一に，動的な P2P ネットワークのピアの障害に対する対応策として，データ複製を用いた**ペアリング方式**を提案する．この方式では，ピアどうしがペアを構成して，お互いにデータを複製して保存する方針をとる．図 9 にこの方式の概念図を示す．この図において，例えばピア A とピア G はペアとなっている．ピア A の複製データをピア G に保存し，ピア G にはピア G に関するデータだけではなく，ピア A の Data リレーション，Change リレーション，To リレーションと From リレーションも保存する．同様に，ピア G の複製データをピア A に保存し，相互の複製によりデータの信頼性を高めている．

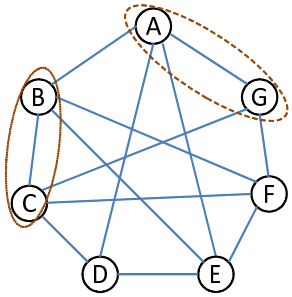

図 9　相互ペアリング方式

この方式において，あるピアにおいて障害が発生した際にも対応可能にするには，どのピアがどのピアとペアを構成しているかを P2P ネットワーク上で共有しておく必要がある．そのための手法としては，DHT などを活用し，論理的なピア ID（たとえば，図の例の場合は A）と，それに対するプライマリピア（A）のアドレスとセカンダリ（G）のアドレスを共有しておく．これにより，ピア A が障害によりアクセスできなくなった場合，セカンダリピアである G を用いて，トレースに関する問合せを実行することができる．

この方式は，ピアが情報交換のネットワークから永久に退出するときにも対応可能である．たとえば，上の例においてピア A が退出する際には，ピア G がピア A の履歴情報に対するセカンダリピアからプライマリピアへと役割を置き変えるだけでよい．信頼性の保証のためには，ピア A の履歴情報に対する新たなセカンダリピアを設定し，履歴情報を複製する必要がある．

先の例では $k = 1$ 個のピアとペアをなす状況を示したが，安全性をさらに高めるために，複製数を増やすことも考えられる．たとえば，図 10 では，ピア A は $k = 2$ 個のピア E および G とペアを組んでいる．k の値が大きければ安全性は増すが，複製管理のコストが増大するため，コストに配慮した k の値の設定が必要となる．

この方式の問題として，特定のピアに負荷が集中してしまう可能性がある．この負荷問題を避けるための方策としては，ペ

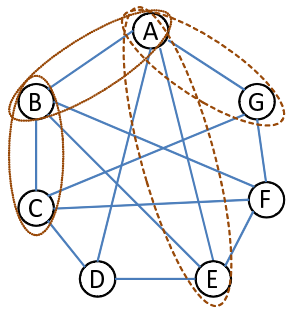

図 10　相互ペアリング方式（$k = 2$）

アとなるピアを選択するための負荷分散を考慮したプロトコルを設け，各ピアはペアを選択する際にそのプロトコルに従うことが考えられる．あるピアがペアとなるピアを探す際，まず，P2P ネットワーク上に問合せを発信し，サービス可能なピアはその問合せに応答する．その際，各ピアは，自身の処理能力や，すでにペアを組んでいるピアの数などを報告する．問合せを発行したピアは，それらの情報に加え，ピア間の通信速度などを考慮し，適切なピアを想定するものとする．

このような考え方をさらに発展させると，時間が経つにつれ負荷が集中してきたピアについては，それが関与している複製処理の一部を他のピアに動的に移行させることも考えられる．

上で説明したペアリング方式は，ピアどうしが双方向に複製を行うというものであったが，さらにこれを緩和して，一方向の複製を行う方式も考えられる．図 11 に，この一方向のペアリング方式を示す．ピアが P2P ネットワークに参加するときに，他のピアに対して，複製データを管理してくれるように要求するメッセージを発行する．その後，返事してきたピアから余裕保存スペースがあるピアを選んで，自分の複製データを移す．

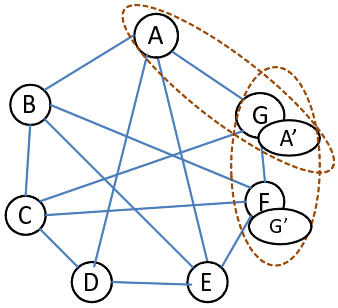

図 11　一方向のペアリング方式

例えば，ピア A に対するセカンダリのピアがピア G であるなら，ピア A の複製データはピア G に保存する．しかし，この方式では，ピア G の複製データは必ずしもピア A に保存するわけではない．

4.1.2　スーパーピア方式

もう一つの選択肢として，**スーパーピア**（super peer）方式も考えられる．スーパーピア方式は，近くに位置するピアどうしが連携し，代表のピアを選択するというものである．スーパーピアは，処理能力が高く，システムの安定性が保証しているピアから選ばれる．このアプローチは，P2P 情報流通ネットワークを物理的にグループ化することに対応する．スーパーピアには，そのグループのすべてのピアのデータと履歴情報が管理される．図 12 にこのアイデアを示す．各グループのスーパーピアはピア A，ピア K，とピア F である．例えば，スーパーピア A は，ピア A 自身のデータと履歴に加え，同じグループに属するピア B，ピア C とピア D のデータと履歴情報も持つ．

スーパーピア方式は，スーパーピアの負荷が大きくなるという問題と，スーパーピア自体の障害に対応できないという欠点がある．しかし，トレース問合せを効率的に処理するために有効に働く可能性がある．他のグループからきたトレース処理の問合せは，グループ内の情報をすべて保持するスーパーピアで

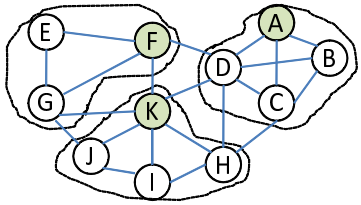

図 12　スーパーピア方式

一括して実行することにより，問合せ処理の効率化が期待できる．たとえば，ピア D がピア E に対し履歴情報をトレースする問合せを発行するとき，直接にピア E に問合せを発行するのではなく，ピア E のスーパーピアである F に送付する．ピア F はピア E の履歴情報をすべて持っているため，ピア E に対する問合せを処理できる．また，ピア F はピア F 自身の履歴情報に加えピア G の情報も保持していることから，問合せの結果ピア F, G の情報が必要になった場合には，自身が管理する情報を用いて回答できる．

上記のアイデアについて，今後さらに検討し具体化を進める予定である．

4.2　問合せの効率化のための構成の改善

以下では，問合せの効率化のために他の改善手法を導入する．

（1）　Data リレーションへの属性の追加

まずは Data リレーションに 3 つ属性を追加する．図 13 は，図 5 に示した物理レイヤのリレーション Data を修正したリレーションを示す．originator 属性はレコードを最初に作成したピアを表す．origin_id という属性はレコードの生成時点における作成ピアにおける ID を表す．すなわち，これら 2 つの属性値を組み合わせると，そのレコードが元々はどのレコードを起点としているかがすぐに判断できる．#hop は，レコードを最初に作成したピアから現在のピアまで，何ステップでレコードが到達したかというパス数を意味している．そのピアでそのレコードが最初に生成された場合には，#hop 属性の値は 0 である．

title	author	id	originator	origin_id	#hop
t1	a2	#A011	C	#C005	2
t7	a7	#A040	A	#A040	0
t7	a6	#A041	A	#A040	0

図 13　属性を追加したピア A の Data[Novel]

この Data リレーションの属性追加に応じて，論理レイヤの Data ビューにも属性を追加する．図 2 の論理レイヤの Data ビューは，図 14 のように修正される．

これら 3 つの属性は，格納コストを多少増やすことになるが，問合せ能力の増大や，処理の効率化に活用できる．#hop 属性は，パス長に関する問合せを可能とする．属性 originator と origin_id は，元々の作成者や元のレコード ID を求めるなどの問合せの処理の効率化に役立つ．以下では，属性を追加する前と属性を追加した後について，2. で述べた問合せの例をもとに比較を行う．

問合せ 1：2.2 で例を示したが，ピア A が保持する，タイトル

title	author	peer	id	originator	origin_id	#hop
t1	a2	A	#A011	C	#C005	2
t7	a7	A	#A040	A	#A040	0
t7	a6	A	#A041	A	#A040	0
t1	a1	B	#B030	C	#C005	1
t1	a2	B	#B032	C	#C005	1
t5	a5	B	#B035	B	#B035	0
t1	a1	C	#C005	C	#C005	0
t3	a3	C	#C055	M	#M001	5

図 14　ビュー Data[Novel]

t1，著者 a1 というレコードに対し，その元々のデータを最初に作成したピアを得るための問合せである．そのような問合せは，以下のように容易に記述できる．

```
Origin(P) :- Data[Novel]('t1','a1', 'A', _, P, _, _)
Query(P) :- Origin(P)
```

このような問合せは再帰を含んでおらず，他のピアの情報にもアクセスする必要がないため，問合せをフォワードする必要はなく，そのピアのみの処理で即座に答えることができる．

問合せ 2：同様に，2.2 で示した問合せ 2（ピア C が保持する著者 a1 により書かれた t1 という小説のレコードをピア A がコピーしたかを調べよ）について考える．この問合せは次のように修正できる．

```
Get(I1) :- Data[Novel]('t1', 'a1', 'C', I1, _, _, _)
Check(I) :- Get(I1),
            Data[Novel](T, A, 'A', I, 'C', I1,_)
Query(I) :- Check(I)
```

この論理レイヤの問合せは次のように処理できる．

（a） まず，問合せを物理レイヤへの問合せへと変換する．

```
Get(I1) :- Data[Novel]@'C'('t1', 'a1', I1, _, _, _)
Check(I) :- Get(I1),
            Data[Novel]@'A'(T, A, I, 'C', I1,_)
Query(P) :- Check(I)
```

この変換のための規則については，[9] で示しているが，論理的なビューではなく，物理的なリレーションを直接参照する点が異なる．ここでたとえば，`Data[Novel]@C` という記法は，ピア C における Data[Novel] リレーションを表している．

（b） 次いで，ルールを実際に評価する．問合せがピア A において発行されたとして考える．上に示した物理レイヤの問合せの 1 番目のルールはピア C の情報にアクセスするため，ピア C に発行される．その結果を得た後，2 行目以降の処理を行い，ピア A は問合せを処理する．

これまでのアプローチでは，ネットワーク上の情報流通経路をたどる再帰的な問合せが必要であったのに対し，この方式により，2 つのピアのみが関与する形で効率的に処理できる．

問合せ 3：先の問合せ 1, 2 のように，追加された属性を用いて効率的に記述可能な問合せが存在する．しかし，当然ながら，追加した属性を有効に活用できない問合せが存在する．ここで，2.2 の問合せ 3（ピア C が保持する著者 a1 により書かれた t1 という本のデータをコピーしたピアをすべて列挙せよ）について考える．この問合せは，追加された属性を用いて以下のように記述できる．

```
Get(I1) :- Data[Novel]('t1', 'a1', 'C', I1, _, _, _)
Reach(P) :- Get(I1),
            Data[Novel](T, A, P, I, 'C', I1,_)
Query(P) :- Reach(P)
```

物理レイヤへの問合せ変換結果は，ほぼ問合せ 2 の場合と同様になる．大きな違いは，第 2 のルールにおいて `Data[Novel]@P` における P が定数のピア名ではく，変数であるということである．

```
Get(I1) :- Data[Novel]@'C'('t1', 'a1', I1, _, _, _)
Reach(P) :- Get(I1),
            Data[Novel]@P(T, A, I, 'C', I1,_),
Query(P) :- Reach(P)
```

1 番目のルールは問合せ 2 と同様に即座に実行できる．しかし，第 2 のルールについては，それを実行すべきピアが不明であることから，処理を継続することはできない．すなわち，この問合せ 3 については，2.2 に示した再帰的な問合せの記述が必要となる．

以上のとおり，追加された属性は，与えられた問合せの効率的な処理において必ずしも有効ではないが，トレーサビリティに関する問合せの多くは，出所を確認する問合せであると考えられる．追加された originator および origin_id 属性により，出所のピアを容易に特定できることで，その種の問合せについては効率的に答えることが可能となる．頻繁に利用される情報を比較的低コストで管理できることから，これらの属性の追加は有効に働くものと考えられる．

（2） **パスキャッシュ**

問合せ処理の効率化に役立つ他の手法として，頻繁に問合せされる情報を手元に保持しておく**キャッシング**（caching）の技術が存在する．本 P2P レコード交換のフレームワークにおいては，トレース問合せの対象となるのは履歴情報であることから，他のピアの履歴情報を手元に置いても，更新などの問題は発生しない．もちろん記憶コストが発生するため，トレードオフを考慮した有効な手法を開発することが求められる．

キャッシングの一つのアプローチとしては，問合せの結果入手した他のピアからの履歴情報をそのまま手元に残すことが考えられる．しかし，このアプローチは場当たり的であり，問合せ処理の効率化に対する見通しが立てにくい．そのため，以下では秩序だったアプローチとして，**パスキャッシュ**（path cache）のアプローチを提案する．

元々の我々フレームワークでは，あるレコードがどのピアから来たか，また，どのピアにあるレコードを提供したかという直接的な情報しか記録してない．この方式は単純であり，格納コストの面で無駄がないという利点があるが，トレース処理のときにレコードが交換された過程に沿ってパスをたどる必要がある．

パスキャッシュでは，さらに前のピアの情報も記録する．図 15 にその概念図を示す．アイデアは，コピー元とさらにそのコピー元の情報を全部記録する．図 7 に示している From リレーションは，図 16 のようになる．たとえば，ピア A のレコード

#A011 がピア B の#B032 からコピーされたという情報を記録するだけではなく，ピア B の#B032 がピア C の#C005 からコピーされたという情報も記録する．同様に，ピア E の From リレーションにはピア A とピア B の情報を記録する．

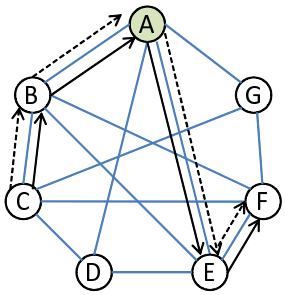

図 15　パスキャッシュ

peer	id	from_peer	from_id	time
A	#A011	B	#B032	5/2/08
B	#B032	C	#C005	4/20/08

図 16　パスキャッシュのピア A における From[Novel]

このようなパスキャッシュは，レコードの流通経路を遡及的にたどる際に有効となる．たとえば，ピア A から問合せを開始し，レコード #A011 が得られた経路に沿って遡及的に追跡を行い，その経路上に出現するすべてのピアの名前を得るような問合せを考える．従来のアプローチの場合は，ピア A における From[Novel] より，ピア B からそのレコードが得られたという情報を入手し，ピア B に問合せをフォワードして問合せを進めることになる．一方，パスキャッシュを用いると，ピア A の From[Novel] により，そのレコードはピア B から直接的に得られ，さらにそれはピア C からコピーされたことが分かる．そのため，問合せはピア B を飛び越して，ピア C に直接フォワードすることができる．このアプローチにより，この問合せに関しては，問合せ処理に関与するピアの数を半分に減らすことができ，問合せ処理の効率化を図ることができる．

このような拡張について，格納コストは余分に必要となるが，その他の管理コストは特に発生しない．パス情報をキャッシュする処理は，レコード交換時に実行することができる．

同様な考え方を，To リレーションの側に適用することができる．図 8 に示している To リレーションは，図 17 のようになる．ピア A のレコード#A041 はピア E に提供して，ピア E はピア F に提供したという情報を保存する．ピア C の To リレーションにピア B とピア A の情報を記録する．ピア B の To リレーションはピア A とピア E の情報を保存する．このような方式により，From リレーションの場合と同様，ある種の問合せの効率化を図ることができる．

peer	id	to_peer	to_id	time
A	#A011	E	#E017	7/5/08
E	#E017	F	#F021	1/10/09

図 17　パスキャッシュのピア A の To[Novel]

しかし，To リレーションの場合には，パスキャッシングでは管理上のコストが余分に発生する．図 17 のような情報を管理するためには，ピア E からピア F にレコードがコピーされる際に，コピーが発生したという情報をピア A に伝える必要がある．すなわち，ピア E からピア F へのレコードのコピーのトランザクションには，ピア E, F のみならずピア A も関与する必要がある．これは付加的なオーバヘッドとなる．

本節では，物理レイヤにおけるシステムの構成の改善と問合せ処理の効率化に役立つ物理構成の改善方式について議論した．今後，これらの具体化を進めていきたいと考えている．

5. 関 連 研 究

近年，情報の信頼性を確保するデータの出所を追跡可能とする *lineage tracing* あるいは *data provenance* が大いに注目され，大きな課題になっている [13], [14]．本研究では，情報交換が頻繁になっている P2P ネットワークにおける科学的なデータがどのピアからどのような経路で入手されたか，また，どのピアに複製，修正されたかを追跡することに注目している．

P2P データベースに関しては，動的な環境でのデータ管理，異種のスキーマのマッピングや，問合せ言語の開発，索引・複製技術など，さまざまな研究がなされている [1]．それらの目標は，異種混合・自律的な P2P ネットワーク環境中のデータ統合問題，異種スキーマ，信頼性管理などの問題である．特に，P2P ネットワークにおける協調的なデータ共有を目指す Orchestra プロジェクト [5] は本研究と関連が深い．これに対し本研究では，単純なレコード交換に注目し，より完全なトレースできるフレームワークを構築した．datalog を用いる問合せを P2P ネットワークに実行できる．P2P ネットワーク上の情報交換を背後で支え，トレーサビリティを実現するための基盤技術としてデータベース技術を用いる点に特徴がある．

本研究で提案した複製による信頼性の確保のための物理構成方式は，P2P ネットワーク上で情報を共有して保持する LOCKSS [4] の複製方針と深い関連するが，本研究では，ピアの同期的複製を行い，より統制された実現を行う必要がある点が異なっている．

6. まとめと今後の課題

本研究では，P2P ネットワークにおけるトレーサビリティを実現するためのレコード交換システム機構の物理レイヤにおける構成方式の改善について，アイデアを示した．特に，

（1） ピアの間で情報を複製管理することによる，ネットワークやピアの障害への対応

（2） 物理レイヤ・論理レイヤのリレーションへの属性の追加による，頻繁に用いられる問合せの効率化

（3） 情報流通の経路のキャッシングにより，効率的な問合せ処理の実現

について，例を用いて説明した．今後はこれらのアイデアの詳細化を行い，実験等による評価を進めたい．

謝　　辞

本研究の一部は，日本学術振興会科学研究費基盤研究（19300027）の助成による．

文　　献

[1] K. Aberer and P. Cudre-Mauroux. Semantic overlay networks. In *VLDB*, 2005. (tutorial notes).

[2] S. Abiteboul, R. Hull, and V. Vianu. *Foundations of Databases*. Addison-Wesley, 1995.

[3] P. Buneman, J. Cheney, W.-C. Tan, and S. Vansummeren. Curated databases. In *Proc. ACM PODS*, 2008.

[4] T. Giuli, P. Maniatis, M. Baker, D. S. H. Rosenthal, and M. Roussopoulos. Attrition defenses for a peer-to-peer digital preservation system. In *Proc. USENIX Annual Technical Conference*, pp. 163–178, 2005.

[5] Z. Ives, N. Khandelwal, A. Kapur, and M. Cakir. Orchestra: Rapid, collaborative sharing of dynamic data. In *Proc. CIDR*, pp. 107–118, 2005.

[6] J.Gray, P. Helland, P. O'Neil, and D. Shasha. The dangers of replication and a solution. In *Proc. ACM SIGMOD*, pp. 173–182, 1996.

[7] F. Li, T. Iida, and Y. Ishikawa. Traceable P2P record exchange: A database-oriented approach. *Frontiers of Computer Science in China*, 2(3):257–267, 2008.

[8] 李 峰栄, 飯田 卓也, 石川 佳治. トレーサブルな P2P レコード交換システムにおける問合せ処理. 情報処理学会研究報告, 2008(56):105–112, 2008.

[9] 李 峰栄, 飯田 卓也, 石川 佳治. トレーサブルな P2P レコード交換システムにおける問合せ処理の効率化について. 情報処理学会研究報告, 2008(88):97–102, 2008.

[10] F. Li and Y. Ishikawa. Traceable P2P record exchange based on database technologies. In *Proc. Asia Pacific Web Conference (APWeb 2008)*, pp. 475–486, 2008.

[11] B. T. Loo, T. Condie, M. Garofalakis, D. E. Gay, J. M. Hellerstein, P. Maniatis, R. Ramakrishman, T. Roscoe, and I. Stoica. Declarative networking: Language, execution and optimization. In *Proc. ACM SIGMOD*, pp. 97–108, 2006.

[12] W. S. Ng, B. C. Ooi, K.-L. Tan, and A. Zhou. PeerDB: A P2P-based system for distributed data sharing. In *Proc. ICDE*, 2003.

[13] W.-C. Tan. Research problems in data provenance. *IEEE Data Eng. Bull.*, 27(4):45–52, 2004.

[14] J. Widom. Trio: A system for integrated management of data, accuracy, and lineage. In *Proc. CIDR*, pp. 262–276, 2005.