デマンド型ニュース配送方式の実装と評価

菅野 浩徳、曽根 秀昭、根元 義章
東北大学大学院情報科学研究科

概要

インターネットにおけるコンテンツ提供サービスには、予め複数のサーバにコンテンツを転送蓄積し、そこからユーザに供する形態のものがある。ネットニュースもその一つであり、利用者からの需要を考慮せず記事の配送を行うために、購読率が低く、無駄な配送が生じる。この問題を解決するため、我々はデマンド型ニュース配送方式を提案している。本稿では、当システムの実装方式について検討を行うとともに、実装実験の結果から提案手法の有効性を示す。

Implementation and Evaluation of an On-Demand NetNews Distribution System

Hironori KANNO, Hideaki SONE, Yoshiaki NEMOTO
Graduate School of Information Sciences, TOHOKU University.

Abstruct

In the contents offering service in the Internet, there is a way that contents are transferred to more than one server and accumulated on them in advance. After accumulation, those containts are offered to a user from there. NetNews System is one of them. In this system, a rate of reading articles is low and network traffic contains unnecessary data. These unnecessary data may be caused by not reflecting on actual user's demand. To overcome this problem, we already proposed an On-Demand NetNews distribution system to solve the above problem. In this paper, we discuss an implementation method of our system and show its efficiency by the result of experiment.

1 はじめに

インターネットで利用されているコンテンツサービスには、FTP、WWW、VOD サーバ等におけるミラーリングや、メーリングリスト、ネットニュースなど、同じ内容のデータを複数のサーバに予め転送蓄積しておき、そこから近接のユーザに供する形態のものがある。

これは、コンテンツの流量が少なく、通信帯域やサーバの能力が不十分だった時代には最適な構成であったと言える。しかしながら、本質的に利用頻度の少ないデータも転送し蓄積することが必要とされていること、またインターネットの発展と利用形態の多様化などによって、転送すべきデータ量が増加し、通信トラヒックの増加、配送に多くの時間を要するなどの問題を抱えている。

通信量や配送時間の問題については、ギガビットを超える広帯域なネットワークが実用段階に入ってきており、大量のデータを短時間に転送できるようになった事で解決できるとしても、利用頻度の少ないコンテンツを含む配送の在り方については、ネットワークシステムにおける効率性から見て好ましくなく改善を要する。

我々は、このような旧来のコンテンツ配信サービスの中から、まずネットニュースについて取り上げ、新たな効率的配送方式について検討を行う。

2 ニュース配送における問題

近年のインターネットの利用者数の拡大に伴って、ネットニュースの流量も増加しており、一日当たりの配送量が受信約 60GB、配信約 500GB を越える事もある [1]。このような配送量を維持するために、広帯域な通信回線や高いサーバ能力、大容量の DISK が必要となる。

しかしながら、配送された記事の利用効率 (購読率) は約 5%程度と低く、残りの 95%は、配送されながらも一度も読まれる事なく削除されており、無駄な配送蓄積が生じている [2]。

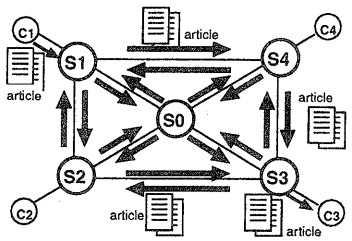

図 1: ニュース配送モデル

これは、図 1 に示すように、各ニュースサーバ {Sn} 間で、全ての記事を交換し合い蓄積することに起因している。サーバ間の配送は、クライアントからのアクセスとは非同期に行われ、配送の際にその記事に対する需要を知る術がない。この問題を改善するため、これまで、自サーバにおけるニュースグループの利用統計から配送を受けるグループを決定し、上流サーバに通知する事で記事の流量を抑える方式 [3] や、末端のニュースサーバをキャッシュサーバとし、一度取得した記事の再利用を行う事で効率化を図る方式の研究 [4] が行われている。末端のニュースサーバをキャッシュサーバとして利用するものとして NNTPCache、DNEWS などといった実装がある。

ここで、他サーバへの中継配送機能を持つサーバをバックボーンノードサーバ、中継配送機能を持たないサーバをリーフノードサーバと呼ぶことにすると、これまでの提案方式はいずれもバックボーンノードサーバとリーフノードサーバ間の配送効率を改善することを目的としており、より配送量が多く、サーバ負荷の高いバックボーンノードサーバ間の配送については考慮されていない。

これらの現行のニュース配送における問題点を解決するには、需要に応じた記事の配送が有効であると考えられる。このためにはユーザからのリクエストに応じて目的とする記事データを配送する、デマンド型が最適であろうと考えた。

このデマンド型配送方式を採用する事で、これまでのようにすべての記事を各サーバが蓄積する必要性は薄れ、各ニュースサーバにオリジナルの記事だけを分散して蓄積すれば、サーバ負荷や回線負荷なども軽減される事につながる。同時に中継配送の必要性も無くなり、従来のバックボーンノードサーバにおける問題点も解決できる。

本方式では、記事の要求から配送完了までの配送時間が懸念されるが、広帯域ネットワークの利用によれば、記事データはより高速に配送され、この問題は解決される。

このような検討を踏まえ、我々は、広域かつ広帯域なネットワーク上に一次配信サーバ（一次サーバ）、二次配信サーバ（二次サーバ）、ディレクトリサーバから構成されるデマンド型ニュース配送方式を提案している [2]。本稿では、このデマンド型ニュース配送方式における動作や通信プロトコル、メッセージ形式、キャッシュ機構などについての検討を行い、さらに実装を用いた実験によってその有効性を示す。

3 デマンド型ニュース配送方式

3.1 構造

提案しているデマンド型ニュース配送方式のモデルを図 2 に示す。

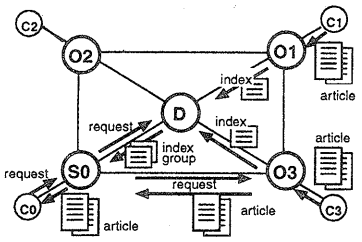

図 2: 提案配送方式モデル

図 2 中の {O1,O2,O3} は一次サーバを、{S0} は二次サーバを、D はディレクトリサーバを、{C0,C1,C2,C3} は、ニュースリーダが動作するクライアントを示す。各サーバは、広域かつ広帯域なネットワーク上に配置される。図 2 のネットワーク接続は論理的なものであり、物理的な構成を示すものではない。

蓄積される記事には、On 内で一意な認識番号を与え、それをファイル名としてニューススプールに格納する。例えば、`/news/spool/15` のようなファイル名である。各ニュースサーバは、FQDN(Fully Qualified Domain Name) によって一意性が保証されるので、そのサーバ中に存在する記事ファイルの一意性が保証されていれば、ネットワーク上での一意性も保証される。つまり URL(Uniform Resource Locators) によって記事の蓄積場所を示す事が可能となる。本モデルでは、`news://o1.foo.com/15` のように URL 表記する。

D では、グループファイルや記事インデックスファイルを一元管理する。グループファイルとは、図 3 に示すようなデータベースであり、存在するニュース

グループの名前や、記事番号などを管理する。これはINNなどにおけるactiveファイルと同等である。

```
newsgroup       high        low         flag

fj.announce     0000000020 0000000001 m
fj.net-people   0000000010 0000000001 y
fj.net.fax      0000000023 0000000002 y
fj.net.ip       0000000111 0000000001 y
```

図 3: グループファイルの例

記事インデックスファイルとは、図4に示すように、投稿記事のヘッダ[5]から利用者が記事の選択を行うために必要となる情報を切り出し、記事の投稿順に与えるシーケンシャルな番号を付加したデータからなるデータベースであり、ニュースグループ毎に存在する。

```
X-seq: 0000000001
Newsgroups: fj.os.bsd.freebsd
Message-ID: <03sp.siw@foo.com>
Date: 17 Feb 2000 21:54:38 +0900
From: bar <bar@foo.com>
Subject: Information
Lines: 41
X-URL: news://news.foo.com/82
```

図 4: 記事インデックスの例

3.2 記事の配送手続き

クライアント Cn からの投稿記事は自サイトの一次サーバ On に蓄積される。同時に On はディレクトリサーバ D に対して投稿記事の記事インデックスを送る。記事を購読する時は、Cn から二次サーバ Sn に対してグループファイルや記事インデックスの配送要求を行い、Sn は D にその要求を中継し、D からのレスポンスやデータを Cn に返す。同様に Cn は Sn に記事データの配送要求を行い、Sn は On にその要求を中継し、On からのレスポンスやデータを Cn に返す。例えば図2において、C1からの投稿記事は O1 に蓄積される。同時に O1 は投稿記事の記事インデックスを D に送る。C0 は S0 にグループファイルや選択したニュースグループの記事インデックスの配送を要求し、S0 は D からグループファイルや記事インデックスを受け取り C0 に返す。C0 は S0 に記事の配送を要求し、S0 は O3 から記事を受け取り C0 に返す。

クライアントからのニュース記事の投稿及び購読は、図5に示すような一連の手続きにより行われる。

このとき、図5中の (1) ではグループファイルが、(3) では記事インデックスファイルが、(4) では記事ファイルが必要となる。記事インデックスファイルは、(2) の処理によって生成される。

本モデルでは、一次サーバ、ディレクトリサーバへのアクセスについて、必ず二次サーバを経由する

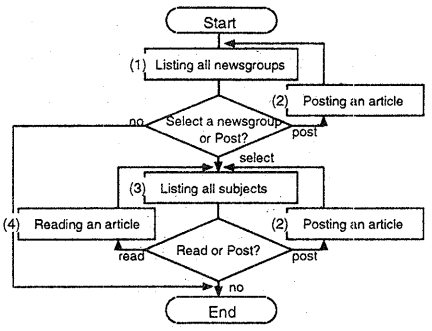

図 5: 記事の投稿及び購読フロー

という特徴がある。そこで二次サーバにキャッシュ機構を持たせることで、再利用性の高いオブジェクトについて重複配送を抑止し配送効率を向上させることができる。

3.3 システムの動作とプロトコル

クライアントと一次、二次サーバ間の通信プロトコルは、RFC977(Network News Transfer Protocol)[6]に準拠したコマンドとメッセージを使用する。このためこれまでのニュースリーダがそのまま利用可能である。また、一次、二次、ディレクトリサーバ間の通信には、RFC977準拠コマンドのほか、以下のコマンドを追加している。

- XPOST
  一次サーバからディレクトリサーバに記事インデックスを送るために用いる。
- XARTICLE
  二次サーバが一次サーバから記事データを取得するために用いる。
- XSTAT
  ディレクトリサーバ上のグループファイルや記事インデックスファイル等の更新時刻を取得するために用いる。

当システムの動作について、記事の投稿の場合と購読の場合を、それぞれ図6と図7に示す。

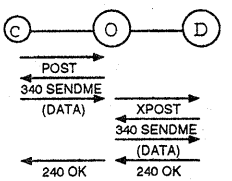

図 6: 記事の投稿

記事の投稿の場合、クライアント C は一次サーバ O に POST コマンドによって投稿する。O は受け

取った記事から記事インデックスを作成し、XPOST コマンドによってディレクトリサーバ D に投稿する。D は O に転送完了レスポンス (240) を返し、O はそれを C に返すことで処理が完了する。

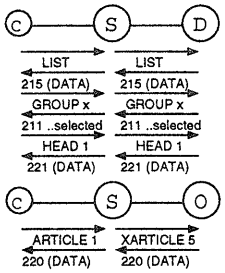

図 7: 記事の購読

記事を購読する場合、クライアント C からの LIST/GROUP/HEAD[1] といったコマンドは、S によってディレクトリサーバ D に中継される。ディレクトリサーバでは、LIST コマンドについてはグループリスト配送レスポンス (215) とリストを、GROUP コマンドについてニュースグループ確認レスポンス (211) を HEAD コマンドについては、記事インデックス配送レスポンス (221) とデータを返す。

記事を要求する ARTICLE コマンドの場合に、S は記事インデックスから URL を照会し、XARTICLE コマンドによって記事を保持する O に記事を要求する。O は S にデータ配送レスポンス (220) と記事データを返し、S はそれを C に返すことで処理が完了する。

3.4 キャッシュ動作

キャッシュの動作については、以下の 2 方式を採用した。

- 更新時刻比較方式 (図 8)
  二次サーバ内にオブジェクトをキャッシュする際に、当該オブジェクトの更新時刻も保持しておき、次に要求があった時点で、ディレクトリサーバ D に存在するオブジェクトの更新時刻を問い合わせ (XSTAT)、比較し、同一であればキャッシュ内のデータを使用する方式である。これは、グループファイルや記事インデックスファイルなど時間の変化に伴って内容が変化し、同一性を保持することが困難なオブジェクトに適用される。
- ファイル確認方式 (図 9)
  二次サーバ内にオブジェクトをキャッシュする際に、一意性を確保できるようなファイル名で格納し、次に要求 (ARTICLE) があった時点で、それが存在すればキャッシュ内のデータを使用する方式である。ファイル名の生成には、記事の URL を入力として、一意なファイル名を生成する一方向関数 $url2fname(url)$ を用いる。例えば入力 URL が、`news://o1.foo.com/35` とすると、出力は、`/news/cache/o1.foo.com.35` を得る。これは、記事ファイルのように同一性を保持することが可能なオブジェクトに適用される。

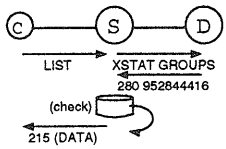

図 8: 更新時刻比較方式

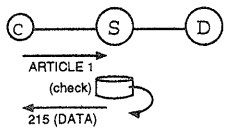

図 9: ファイル確認方式

4 実装による実験

4.1 実装

提案するシステムの実装を行った。開発したシステムは C 言語により記述し、一般的な UNIX プラットフォーム上で動作させた。

トランスポート層のプロトコルは TCP を使用し、一次サーバ、二次サーバ機能を行うプログラムとして"n3d"、ディレクトリサーバ機能を行うプログラムとして"n3dsd" を、それぞれデーモンとして常駐させる。"n3d" では起動時にキャッシュ機能を有効化、無効化するパラメータを設定できるので、実験条件に応じて設定を変えることでキャッシュによるトラヒックの変化についても確認できる。

4.2 実験

今回の実装を用いた配送システムの評価は、従来の配送システムとのトラヒック量の比較によって行う。

実験には図 10 のようにディレクトリサーバ 1 台 (O) と、一次サーバが 2 台 (O1,O2)、二次サーバ 1 台 (S) として、実験的なネットワークを構築した。各サーバの OS は、Solaris2.6J である。

O1,O2 の各サーバに、既存のニュースサーバから配送された記事を注入する。この際、乱数から 2 値

[1] XOVER も可能。

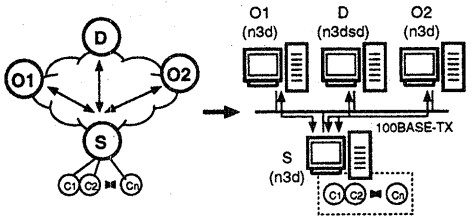

図 10: 想定モデルと実験装置

を発生する関数により、O1,O2に記事が分散されるよう留意した。Sには本来の二次サーバ機能の他、クライアントプロセスを動作させる機能を持たせ、運用中のニュースサーバのアクセスログを入力としてアクセスする機能を持たせた。実験ネットワーク上を流れるトラヒックの測定にはtcpdumpを用いた。

実際の運用データを解析するのではなく、既存のニュースサーバからそのログを基に実験システムに記事を注入する方法で実験を行った理由は、ネットワークやニュースサーバの利用状態が刻々と変化する運用システムでは、配送量などのトラヒックを同じ条件で比較するのは困難であると判断したためである。

本実験において使用した配送データ及びアクセスログは、ある商用ISP(Internet Services Provider)において実際に稼働しているニュースサーバのものを使用した。使用したデータ項目は、ニュースグループ名、アクセスコマンド、転送量、記事ファイル名、メッセージIDだけであり、データの匿秘性に注意を払い、アクセス元アドレスなどアクセス者が特定できる項目は一切使用していない。

4.2.1 実験1

まず、ある6日間の配送記事を実験システムに注入した。実験は1日毎のデータをO1,O2に注入し、Sからアクセスを行い計測するという手順を繰り返した。この際、キャッシュ機能の効果を明らかにするために、キャッシュ機能を無効とした場合と有効とした場合とに分けて行った。キャッシュは、その効果が最大になる場合を想定し、測定中にはファイルの更新などが行われないものとした。

実験1では、先に述べた運用中のニュースサーバへの配送データとアクセスデータをほぼそのまま利用する。これにより、実際に運用中のシステムに適用した場合の評価を行なう。

実験1に用いたデータの内容を表1に示す。平均購読率とは、配送記事数Trに対して、実際に読まれた記事数Arの割合(Ar/Tr)を対象日数で平均したものである。このとき同一記事について複数回アクセスがあった場合でも一回として計算し、複数回分計上したものを括弧内に示した。

表 1: 実験1データの内容

name	value
対象日数	6
ニュースグループ数	21,503
総記事数	611,737
総配送量	30,149 MB
平均購読率	4.2 % (8.2 %)

表 2: 実験2データの内容

name	value
対象日数	6
ニュースグループ数	430 (主に fj.*)
総記事数	9,780
総配送量	32 MB
平均購読率	49.9 % (67.7 %)

4.2.2 実験2

次に、ある6日間の配送記事について、比較的購読率の高い特定のニュースグループに絞って、その記事を実験システムに注入し実験した。これにより、現在の配送方式において良好な購読率を保っているサーバに適用した場合の評価を行なう。

実験手順やキャッシュの条件などは実験1と同様である。実験2に用いたデータの内容を表2に示す。

4.3 実験結果と考察

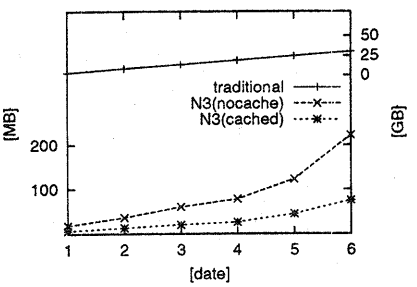

図 11: 累積配送量 (実験1)

実験1による累積配送量のグラフを図11に示す。横軸は日数、縦軸は累積配送量である。6日目の時点で従来方式が30GB程度、提案方式(キャッシュ無効)の場合で220MBと、提案システムの方がはるかに配送量を抑えられることがわかる。またキャッシュ機能を有効とする事で、その配送量をさらに65%程度効率化できる事が確認できる。

実験2による累積配送量のグラフを図12に示す。横軸は日数、縦軸は累積配送量である。キャッシュ

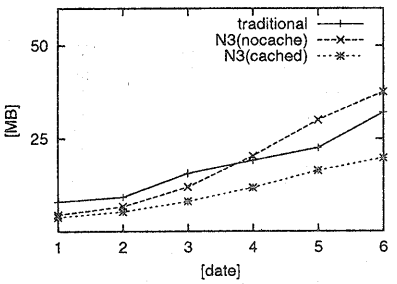

図 12: 累積配送量 (実験 2)

機能を無効とした場合に 4 日目以降から従来のシステムよりも配送量が増えているが、キャッシュ機能を有効とした場合には、従来の配送量よりも低く押える事ができている事が分かる。実際には、オリジナルのオブジェクトの更新や、キャッシュ内オブジェクトの削除による再配送によってキャッシュ効率が低下するので、その時の配送量はキャッシュ機能を無効にした場合のグラフと有効にした場合のグラフとの間に位置すると考えられる。従ってキャッシュ機能を有効とした場合でも、従来の配送量を越えてしまう可能性がある。オブジェクトの更新による再配送については、更新部分だけを差分データとして配送し配送量を抑える、オブジェクトの削除による再配送については、キャッシュ保持期間を長くし、オブジェクトへのアクセス回数をできる限り抑える、といった対策が考えられる。

各サーバへの通信量を見るために、実験 1 において観測されたディレクトリサーバと一次サーバへのパケット量を図 13 に示す。横軸は日数、縦軸が累積パケット量である。一次サーバへの累積パケット量は 2 台分を合算してある。

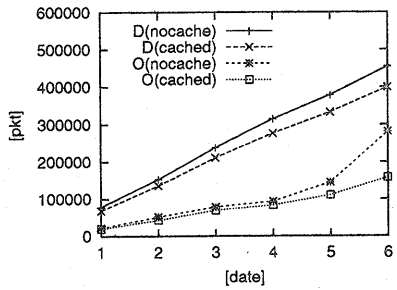

図 13: 累積パケット量 (実験 1)

これから、一次サーバへのアクセス量が一次サーバへのアクセス量と比較して大きい事が分かる。一次サーバは分散して配置されるため、一台当たりにかかるトラヒックはさらに分散される。

ディレクトリサーバの応答性を確保するためには、ディレクトリサーバへのアクセス負荷を低く抑える必要があるが、それにはディレクトリサーバの分散化、データ配送の効率化などを視野に入れる必要があると考える。

5 おわりに

本稿では、提案するデマンド型ニュース配送方式について、通信プロトコルやメッセージ、キャッシュ機能などの検討を行い、実装による実験を通じてその評価を行った。

実験によれば、バックボーンノードサーバのような環境にあるサーバおいて、特にその効果がある事が確認できた。一方、キャッシュが十分に機能しない場合には、従来の配送方式よりも配送トラヒックが増加する可能性がある事を示した。当システムの実際の運用を考えた場合、ディレクトリサーバへのアクセスの集中による負荷が懸念されるため、ディレクトリサーバの分散化が必要であると考えている。また、マルチキャスト機能により、ディレクトリサーバ上のデータを配送することで、ディレクトリサーバへのアクセス負荷を抑えられると共に、共有するバックボーン上のトラフィックも効率化できると考えている。今後これらの検討を進めると共に、記事の配送時間についても定量的な評価を行ってゆく予定である。

昨今、電子メールにおいても、添付ファイルの多用によるトラヒック増や、宣伝や勧誘などの、利用者側で意図しないメールの配送が増加している。このようなメール配送に関するトラヒック対策についても当モデルの応用が可能と考えており、検討を進める予定である。

参考文献

[1] Daily Usenet eport, http://newsfeed.mesh.ad.jp/

[2] 菅野浩徳、曽根秀昭、根元義章：“ネットニュースにおけるデマンド型配送方式の提案” 電子情報通信学会 2000 年総合大会講演論文集、B-7-111 PP.204, (2000)

[3] 渡辺健次、小串英俊、近藤弘樹：“読まれているニュースグループのみ自動的に購読するニュースシステムの構築” 情報処理学会分散システム運用技術研究会報告、No.2 96-DSM-2 PP.19-24, (1996)

[4] 舟阪淳一、最所圭三、福田晃：“NetNews のためのキャッシングアルゴリズム” 電子情報通信学会論文誌 B、Vol.J82-B No.5 PP.818-826, (1999)

[5] M.Horton,R.Adams : Standard for Interchange of USENET Messages, *RFC1036*, (1987)

[6] Brian Kantor,Phil Lapsley : Network News Transfer Protocol, *RFC977*, (1986)