HTML文書間のデータ共有を考慮した放送型情報提供方式

石川裕治[†] 田辺雅則[†,☆]
箱守聰[†] 井上潮[†]

本論文ではデータファイルの共有を考慮したHTML文書の放送方式について述べる．HTML文書はリンク先として，同じデータファイルを共有していることが多い．n個のHTML文書が同じデータファイルfを共有しているとすると，HTML文書を個別に伝送した場合，fをn回放送することになり効率が悪い．本論文では，**a)** 共有データファイルの伝送回数を最適化し，**b)** データの放送順序をHTML文書のアクセス確率とデータサイズを考慮して決定することで，端末のHTML文書の取得時間を短縮する方式を提案する．そして提案方式が$O\big(n\log(n)\big)$で準最適な放送順序を求められることを明らかにし，それは最適な場合とほぼ変わらないことを机上評価により示す．また，実際に公開されているWWWのHTML文書の集合を用いて提案方式の評価を行い，共有関係を考慮しない場合に比べ，取得時間が最高で30%以上短縮されることを示す．

Data Broadcast Method for HTML Documents with Shared Files

Yuji Ishikawa,[†] Masanori Tanabe,[†,☆] Satoshi Hakomori[†]
and Ushio Inoue[†]

This paper studies a data broadcast method for HTML documents with shared files. HMTL documents often share the same files as their hyperlink destination. We assume that n HTML documents share a set of file f. If such HTML documents with shared files are transfered individually, f is broadcast n times and the efficiency of transmission becomes lower. In this paper, we propose a method, which **a)** broadcasts shared files fewer semi-optimal times and **b)** decides transmission order of the files with their access probablity and data size, for reducing waiting time of clients. We show that the method finds a semi-optimal file sequence in $O\big(n\log(n)\big)$ and give an experimental result of which waiting time of the semi-optimal case is almost the same as the optimal case. Then we evaluate the method with HTML documents open to the public on the Internet. Those results indicate that there are cases where the proposed method improves waiting time by over 30% compared to methods which do not consider shared files.

1. はじめに

近年，携帯電話・PHSといった各個人を対象とした通信サービスからデジタル化による公衆を対象とした放送サービスに至るまで無線通信技術のめざましい発展により，無線ネットワークを通じて様々な情報にアクセスできるようになりつつある．このような状況を受けて，今後は無線ネットワークを通じたWWW（World Wide Web）などのマルチメディア情報の提供サービスが普及していくと思われる．

具体的なサービスの例としては，博物館，遊園地，商店街を訪れた人々の目的に合った情報の提供や，交通，天気，災害など公衆向けの情報の配信が考えられる．無線通信のインフラとしては対象領域の広さに応じて，無線LAN，構内PHS，セルラー，衛星などを使い分けることができる．

多数の利用者にマルチメディア情報を提供するには，無線ネットワークの利用効率を高める必要がある．たとえば，現在の無線通信のインフラの中でも広帯域の部類に入る無線LANであっても伝送速度は数Mbps，同時接続数は数十程度である．そのため，多数の利用者がサーバに接続して情報を取得するには十分ではない．この問題を解決するため，従来のオンデマンド型の通信方式に加え放送型の通信方式を用いることで，多数の利用者に効率的に情報を提供する方式が提案されている[1),2)]．放送型通信では1回の送信で多数の端

† 株式会社NTTデータ情報科学研究所
Laboratory for Information Technology, NTT Data Corporation
☆ 現在，株式会社NTTデータ金融システム事業本部
Presently with Financial Systems Sector, NTT Data Corporation

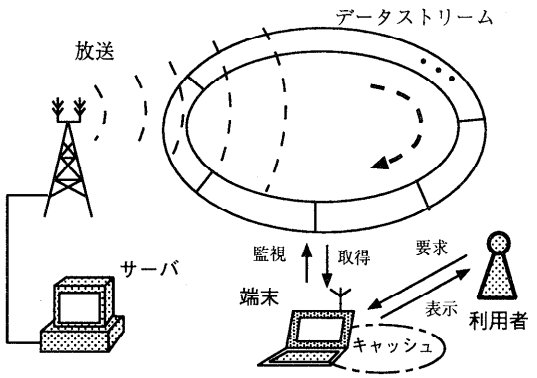

図1 放送型情報提供システムの構成
Fig. 1 Structure of our information-providing system.

末が同時にデータを取得できる．このため，多数の端末から共通して要求されるデータを放送型通信で提供することにより，ネットワークの負荷を減らすことができる．

放送型の情報提供システムの構成を図1に示す．システムはデータを提供するサーバと，データを取得して利用者に表示する端末から構成されている．サーバはデータストリームと呼ぶデータ列を構成し，周期的に繰り返し放送する．一方，端末はデータストリームを監視して必要なデータを取得する．取得したファイルは利用者に対して表示し，必要に応じて端末内部のキャッシュに保存される．

本論文では，提供データの形式としてはHTMLを前提とする．その理由は，**(1)** 複数のメディアの情報を統合して提供できる，**(2)** 現状，大量のコンテンツがHTML文書として作成・蓄積されている，ためである．

放送型情報提供システムではデータストリーム構成とキャッシュ戦略の2つが大きな研究課題となる．キャッシュ戦略としてはFIFOやLRUなどの一般的な戦略よりも，キャッシュミスを起こしたときの放送からのデータ取得時間を考慮した戦略の方が優れていることが知られている[3]．このデータの取得時間は端末がどの順序でデータを受信するか，すなわちデータストリームの構成によって決まる．一般に1つのWWWサイトにあるHTML文書はアイコン，ロゴ，背景などに同じデータを共有していることが多く，データストリームの構成時にはデータ共有を考慮することが重要である．

放送型情報提供システムのアプリケーションとしては，株価や為替レートなど，更新頻度の高いデータを配信する場合が考えられる．データが更新された場合，データの値だけの変更ならばデータストリームを再構成する必要はない．しかし，データ項目の追加，削除やデータ長の変更が生じるとデータストリームを再構成する必要がある．したがって，本論文では以下の2点を議論の対象とする．

(1) データストリームの構成にかかる処理時間
(2) 構成されたデータストリームを周期的に放送した場合のデータ取得時間

本論文では，放送によってHTML文書を効率的に伝送することを目的として，HTML文書間のデータファイル共有を考慮したデータストリーム構成方式を提案する．まず2章で関連研究を紹介する．次に3章で，放送によりHTML文書を伝送する方式について述べる．4章でデータファイルの共有を考慮したデータストリーム構成方式について述べた後，5章で机上評価および結果の考察を行う．最後に6章で全体のまとめを述べる．

2. 関連研究

放送型のデータ通信方式の研究分野では以下のような研究課題があり，インターネット放送，デジタル衛星放送の発展とともに活発に研究されている．

(1) データストリーム構成．データが放送されるまでの待ち時間を最小にする放送データの配列決定方式．
(2) キャッシュ戦略．特にデータをあらかじめ取得しておくプリフェッチ技術．
(3) マルチチャネル対応．帯域分割の最適化手法．
(4) 伝送誤りの考慮．再送要求が難しい状況におけるパケット落ちへの対処方法．

本研究は(1)を対象とする．文献1)~5)は同じ問題を扱っており，サーバが多数のデータを周期的に放送し，端末がフィルタリングをすることで疑似的なオンデマンドを実現することを目標に，データのアクセス確率に基づいて各データの放送頻度を決定し，取得時間の期待値を最小化する方式を検討している．

具体的には，文献1)，2)，4)では放送とオンデマンドの組合せについて，文献3)では放送とキャッシュの組合せについて，それぞれ検討している．また文献5)では待ち行列理論を応用して最適な放送頻度を理論的に示している．これらの研究では放送データをHTML文書に特化せず，放送頻度の最適化によって取得時間を短縮しており本研究とはアプローチが異なる．

一方，モバイルコンピューティング環境におけるHTML文書の伝送方式にはWebExpress[6]がある．この方式では，各端末から送信されるキャッシュデー

タの内容情報をもとにして，サーバは重複するデータを送らないようにし，通信帯域を節約している．しかし，サーバが個々の端末に応じた処理を行うため，放送型通信には不向きである．

放送型のデータ通信方式としては DataWave, Bitcast, PerfecPC! などが実用段階に入っている[7]．これらの方式では受信機として PC を想定しており，大容量のキャッシュ領域を想定している．しかし 1 章で述べたように，株価や為替レートといった更新頻度の高いデータの配信は，放送型データ通信技術の有力なアプリケーションである．こういったアプリケーションに対してはキャッシュを積極的に利用することができない．

また，より大きなマーケットとして家電分野を考えた場合，たとえばデジタル衛星放送ではサービス普及に向け家電各社が，チューナーの価格を数万円台に設定して価格競争をしている．各種の記憶媒体は安価になったとはいえ，大容量の記憶媒体を使って放送データを保存するような方式は採用しにくい．したがって，端末の記憶容量を抑えた方式を考える必要がある．

これに対し開発された方式として DVX[8] がある．DVX は MPEG2 規格の静止画間にリンクを張る機能を持っているため HTML 文書の提供と同等のサービスが可能である．DVX では多くのページで使われるグラフィックスオブジェクトをあらかじめ受信機側に組み込むことで効率化を図っているが，共有されるデータが動的に変わる場合には対応が難しい．

3. 放送による HTML 文書の伝送

3.1 HTML 文書の送信と取得

一般に HTML 文書はハイパーリンクによって関連付けられた複数のファイルから構成される．たとえば一般的な WWW のブラウザでは，1 つの HTML ファイルと複数の画像ファイルで構成された HTML 文書が 1 つの画面に表示される．

本論文では，1 つの HTML 文書は 1 つの HTML ファイル e と，その中で `IMG` タグの `SRC` オプションによってリンクされる複数のデータファイルによって構成されるとする．一方，`A` タグの `HREF` オプションによってリンクされるデータファイル g は，e に対する取得要求とは別に要求しないと一般に取得されないので，HTML 文書の構成要素には含めない．この仮定のもとに，1 つの HTML 文書の伝送を，その文書を構成するファイル集合の伝送であるとする．たとえば，ある HTML 文書が HTML ファイル e_1 と画像ファイル f_1, f_2 から構成されるとすると，この HTML 文書の伝送とは $\{e_1, f_1, f_2\}$ の 3 つのファイルを伝送することである．

データストリーム中には各 HTML 文書を重複なく配置し，かつ，1 つの HTML 文書を構成するファイルも重複なく配置する．つまり端末は 1 周期待てば，どの HTML 文書に対しても必ず 1 度はそれを構成するファイルを放送から取得できる．各ファイルはヘッダを付加したパケットの形で提供される．

著者らは各 HTML 文書を構成する複数のファイルを 1 つの大きなパケット（パッケージ）にまとめて伝送する方式を文献 9) で提案した．放送データの受信装置ではパケットのヘッダを読み込むと，そのパケットのデータを引続き読み込むべきかどうかを確認する処理を行う．HTML 文書をパッケージにまとめることによりデータストリーム中のパケット総数が減るため，この確認処理の回数を減らすことができ，受信装置の負荷が軽くなる．

3.2 パッケージ採用の是非

n 個のファイル $\{f_1, \ldots, f_n\}$ から成る 1 つの HTML 文書を送信する場合，データストリーム中に n 個のファイルをそれぞれ独立なパケットとして連続して配置すると，待ち時間は最小になる[9]．しかし $\{f_i\}$ をまとめてパッケージにすると待ち時間の期待値は増加する．つまり，パッケージにすることでヘッダの検出回数を減らす代わりに，待ち時間を犠牲にしている．そこで，パッケージング方式による待ち時間の期待値の増加分 ΔW について議論する．

単位時間あたりサイズ 1 のデータが送信されるとすると，データストリームのサイズは周期と等しくなり，これを T とおく．f_i $(i = 1, \ldots, n)$ のサイズを u_i, $u_s = \sum_{i=1}^{n} u_i$ とすると，ΔW は式 (1) に表される[9]．

$$\Delta W = \frac{1}{2T}\left\{u_s^2 - \sum_{i=1}^{n} u_i^2\right\} \qquad (1)$$

ΔW はパケットサイズ u_i の割合によって変わってくるが，すべてのサイズ u_i が等しいときに最大となるので，式 (2) の値が上限である．

$$\Delta W \leq \frac{1}{2T}\left\{u_s^2 - \sum_{i=1}^{n}\left(\frac{u_s}{n}\right)^2\right\} = \frac{u_s^2}{2T}\left(1 - \frac{1}{n}\right) \rightarrow \frac{u_s^2}{2T}\ [n \rightarrow \infty] \qquad (2)$$

ΔW の上限が周期の 1%未満となる条件を求めると式 (3) のようになる．

$$(\Delta W <) \frac{u_s^2}{2T} < \frac{T}{100} \iff u_s < \frac{\sqrt{2}}{10}T \qquad (3)$$

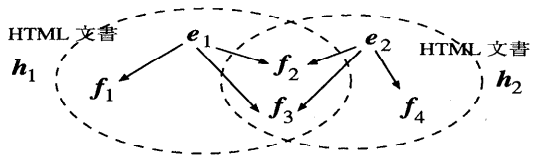

図 2 HTML 文書のデータ共有
Fig. 2 HTML documents with shared files.

つまり，あるハイパーテキスト h に必要なパケットのサイズの和 u_s がデータストリームのサイズ T の14%以下であれば，ΔW は周期 T の1%未満である．

一般に，データストリームは多数のハイパーテキストのデータを含むので u_s に比べて T は非常に大きいと考えられ，パッケージを用いることによる待ち時間の増加は無視できる．よって，以下では複数のファイルをパッケージにまとめて伝送することとする．

放送から HTML 文書を取得する手続きは以下の3ステップから成る：**(1)** ある HTML 文書 h の取得要求を端末が受け付ける，**(2)** 放送データを監視し，ヘッダを見て h のパッケージを取得する，**(3)** パッケージからファイルを取り出して h を利用者に表示する．

(1) と (3) はキャッシュからのデータ取得でも同様の処理が行われる．一方，(2) において，放送型通信ではデータに逐次的にしかアクセスできないため，データを送信する順番が処理時間を決定する．そのため，HTML 文書のデータ共有を考慮してデータストリームを構成することで，(2) の処理時間を短縮することができる．(2) におけるヘッダ検出までの待ち時間とパッケージ読込み時間の和を**取得時間**と呼び，この取得時間を短縮するための方式について議論する．

3.3 取得時間短縮のための課題

2つの HTML 文書が同じデータファイルを用いてそれぞれ2つのページを構成している場合を考える．この例を**図 2** に示す．図2の場合，h_1 と h_2 の構成ファイルをそれぞれ1つのパッケージにまとめて伝送すると $\{f_2, f_3\}$ を2回送ってしまうことになる．つまり，データ共有を考慮せずに HTML 文書を個別に伝送すると，データを共有している HTML 文書の数だけ共有データを送信してしまい伝送効率が悪くなる．

この問題を解決するには，HTML 文書を共有データとその他のデータ（専有データと呼ぶ）に分け，共有データをまとめて伝送回数を減らせばよい．たとえば，h_1 なら $\{e_1, f_1\}$ と $\{f_2, f_3\}$，h_2 なら $\{e_2, f_4\}$ と $\{f_2, f_3\}$ に分ける．そして $\{e_1, f_1\}$，$\{e_2, f_4\}$ および $\{f_2, f_3\}$ から3つのパッケージ H_1，H_2，H_s を作り，データストリーム中に配置する．これにより共有データ H_s の伝送は1周期に1度で済む．その結果，周期が短縮され，データが放送されるまでの待ち時間が短くなる．

しかし，端末は1つの HTML 文書を取得するのに2つのパッケージを取得しなければならないため，共有データと専有データがデータストリーム上で離れて配置されていると取得時間が長くなる．つまり，周期を短縮するために共有データをまとめるが，共有データと専有データは近くに配置したい，というトレードオフを解決する必要がある．具体的に，取得時間が最小になるデータストリームの構成を決定するには以下の3つの課題を解決する必要がある．

課題 1 共有データの数を，それを共有している HTML 文書の数の半分より少なくすると，すべての専有データを共有データの隣に置くことはできない．そのため，どの専有データを優先して共有データの近くに置くかを決める必要がある．

課題 2 共有関係を持たない HTML 文書が存在することで，共有データと専有データの距離を広げている．そのため，共有関係を持たない HTML 文書のデータに関しても適切な配置を決める必要がある．

課題 3 共有データは多ければ周期が長くなり，少なければ専有データを近くに配置しにくくなる．そのため，共有データの配置数を最適化する必要がある．

4章では，各課題に対する我々の解決策を示し，データ共有を考慮したデータストリームの構成方式を提案する．

4. 共有データを考慮した伝送方式

本論文では方式の最適性を取得時間によって評価する．最初に取得時間を定式化する．そして，それを基に各課題を解決するための方式を示す．

4.1 準備：取得時間の定式化

取得時間を定式化するにあたって以下の仮定をおく．

- 一般性を失うことなく議論を簡単にするため，単位時間あたりサイズ1のデータが送信されるものとする．つまりデータストリームのサイズが T であるとすると，周期も T である．
- 共有ファイルが複数存在する場合，一般には各ファイルを共有する HTML 文書の集合が一致しない．そのため，共有ファイルを選択し，各ファイルを共有する HTML 文書の集合の共通部分 S をとり，S に含まれる HTML 文書の共有関係のみを考慮するものとする．この場合においても，十分有効な結果が得られることを 5.2 節で示す．なお，共有ファイルの適切な選択方法に関しては 5.2 節において，方法の一例および実データによる評価結果を示し，本論

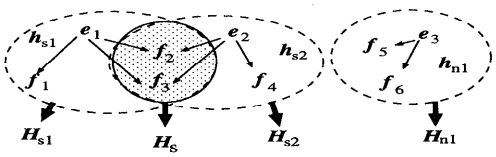

図 3 パッケージの記号
Fig. 3 Symbols of packages.

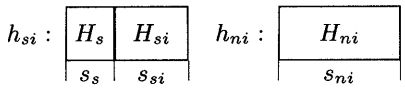

図 4 サイズの記号
Fig. 4 Symbols of sizes.

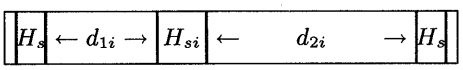

図 5 共有データに対する距離
Fig. 5 Symbols of distances to shared data.

文ではそれ以上議論しない.

つまり放送する HTML 文書は，あるデータファイルの集合 $\{f_i\}$ を共有する n_s 個の HTML 文書 $S = \{h_{si} \mid i = 1,\ldots,n_s\}$ と，共有関係を持たない n_n 個の HTML 文書 $\{h_{ni} \mid i = 1,\ldots,n_n\}$ に分けられる.

- 利用者の要求はランダムに発生する．そのため，データストリームのどの点からも一様な確率でアクセスが開始されるとする.

h_{si} と h_{ni} のアクセス確率をそれぞれ p_{si}, p_{ni} とする．また $\{f_i\}$ のパッケージを H_s，各 h_{si} の専有データのパッケージを H_{si}，各 h_{ni} のパッケージを H_{ni} とする（**図 3**）．各パッケージのサイズを**図 4** のようにおく.

データストリーム中に各 H_{si} と H_{ni} は 1 つだけ，H_s は m $(1 \leq m \leq n_s)$ 個配置することを考える．この場合の周期 T_m は式 (4) に表される.

$$T_m = \sum_{i=1}^{n_s} s_{si} + \sum_{j=1}^{n_n} s_{nj} + m s_s \qquad (4)$$

h_{si} を取得するには H_s と H_{si} の 2 つのパッケージを取得する必要がある．取得時間に影響するのは H_{si} が放送される前に最後に放送された H_s と，H_{si} が放送された後に最初に放送される H_s だけであり，他のパッケージの配置は h_{si} の取得時間とは関係がない．具体的には，H_{si} から最も近い H_s までの距離を d_{1i}，(その H_s とは反対側の) 2 番目に近い H_s までの距離を d_{2i} とする（**図 5**）と，h_{si} の取得時間の期待値 E_{si} は式 (5) のとおりとなる．なお，式 (5) の導出過程は付録 A.1 に示す.

$$E_{si} = \frac{T_m}{2} + s_{si} + \frac{(d_{1i} + s_s)(d_{2i} + s_s)}{T_m} \qquad (5)$$

一方，h_{ni} を取得する場合，H_{ni} の取得時間の期待値 E_{ni} は，H_{ni} の配置に関係なく $T_m/2 + s_{ni}$ である☆.

したがって，H_s を m 個配置するときの取得時間の期待値 $\mathcal{E}_m$ は式 (6) に表される.

$$\mathcal{E}_m = \sum_{i=1}^{n_s} p_{si} E_{si} + \sum_{j=1}^{n_n} p_{nj} E_{nj} \qquad (6)$$

つまり最適なデータストリームを構成する問題は，アクセス確率 p_{si}, p_{nj} とデータサイズ s_s, s_{si}, s_{nj} が与えられたときに，式 (6) の値を最小化する m と $\{H_s, H_{si}, H_{ni}\}$ の配置（すなわち d_{1i}, d_{2i} の値）を見つける，という問題にほかならない．$1 \sim n_s$ の各 m に対して，$\{H_s, H_{si}, H_{ni}\}$ のすべての配置について $\mathcal{E}_m$ の値を求めれば，最適な配置を見つけることができる．しかし，配置の組合せは

$$\sum_{m=1}^{n_s} \frac{(n_s + n_n + m - 1)!}{(m-1)!\,(n_s - 1)!}$$

通りであり，膨大であるため現実的な手段とはいえない．そこで，本論文では最適解の必要条件を用いて，実用的な時間で準最適解を求める方式を提案する.

まず 4.2 節において m が与えられている場合に準最適なパッケージ配置を $O\big(n_s \log(n_s)\big)$ で決定する方式として**順序配置方式**を提案する．その後，4.3 節で最適な m を決定する方式を示す．最適な m の決定は単純には $O\big((n_s)^2 \log(n_s)\big)$ かかるが，$O\big(n_s \log(n_s)\big)$ で最適な m を推定する方法を示す．なお 4.2, 4.3 節の方式の精度については次の 5.1 節で評価を行う.

以下では 3.3 節の課題を解決しながら，最終的に**図 6** の構成を持つデータストリームを得るまでを説明する.

4.2 順序配置方式

4.2.1 課題 1：パッケージ H_{si} の列の構成

どの H_{si} を優先して H_s の近くに置くかについて，次の 2 つの優先順序が考えられる.

(1) 頻繁にアクセスされる h_{si} のパッケージ H_{si} を優先して近くに配置し，取得時間 E_{si} を短縮する．結果に期待値 $\sum_{i=1}^{n_s} p_{si} E_{si}$ は小さくなる.

(2) サイズ s_{si} の小さな H_{si} を優先して，その後ろに並ぶパッケージがあまり H_s から離れない

☆ 平均で半周期待ってデータを s_{ni} 時間読み込む，が式の意味である.

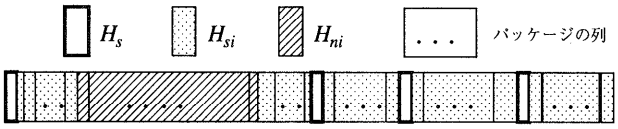

図 6 データストリームの構成（$m = 4$ の場合）
Fig. 6 Structure of Datastream (in case of $m = 4$).

ようにする．

式 (6) を解析し優先順序を検討した結果[10]，最適な配置に対する以下の必要条件を見つけた☆．

$\mathcal{E}_m$ を最小にするには，p_{si}/s_{si} の値の大きい H_{si} ほど H_s の近くに配置する必要がある．

そこで，以下の 2 つの方針に基づき準最適な d_{1i} を決める方式を提案する．

方針 1： すべての H_{si} を H_s の前後に連続して配置する（各 H_{si} と H_s をできるだけ接近させる）．

方針 2： p_{si}/s_{si} の値に基づいて H_{si} を整列する（上記の必要条件を満たす）．

順序配置方式（前半部）

手順 1： H_{si} を p_{si}/s_{si} の値の大きい順に並べ，キュー $\mathcal{Q}$ を作成する．

手順 2： パッケージを格納するための空の配列 $\mathcal{H}_{si}$ $(i = 1, \ldots, 2m)$ を用意する．なお，各 $\mathcal{H}_{si}$ の長さを $\delta_i = \sum_{H_{sj} \in \mathcal{H}_{si}} s_{sj}$ とする．

手順 3： $\mathcal{Q}$ から先頭の要素 H_{si} を取り出し，長さ δ_j の最も小さい $\mathcal{H}_{sj}$ に H_{si} を加える．このときの δ_j の値を d_{1i} とする．

手順 4： $\delta_j = \delta_j + s_{si}$ とする．$\mathcal{Q}$ が空でなければ手順 3 に戻る．

この方式には以下の特徴がある．

特徴 1： 手順 3 において長さ δ_j が最短の列に要素を加えていくので，各 δ_j はほぼ均等に増加していく．よって最終的に，$2m$ 個の各列 $\mathcal{H}_{sj}$ の長さはほぼ等しい長さ $l = \left(\sum_{i=1}^{n_s} s_{si}\right)/2m$ となる．

特徴 2： 手順 3 の処理時間のオーダはヒープ構造を利用することにより $O\big(\log(m)\big)$ となる．また，手順 3 は n_s 回実行されるため手順 3 と 4 で $O\big(n_s \log(m)\big)$ かかる．

$\mathcal{H}_{si}[j]$ を $\mathcal{H}_{si}$ の j 番目のパッケージとすると，方針 1 より，各 $\mathcal{H}_{si}$ はデータストリーム中に次の (a), (b) どちらかの状態で配置される．

(a) $H_s\, \mathcal{H}_{si}[1]\ \ldots\ \mathcal{H}_{si}[k]$ （$H_s\, \mathcal{H}_{si}$ で表す）

(b) $\mathcal{H}_{si}[k]\ \ldots\ \mathcal{H}_{si}[1]\, H_s$ （$\bar{\mathcal{H}}_{si}\, H_s$ で表す）

つまり，H_s と H_{si} を全体として図 7 のように配置

$$H_s\, \mathcal{H}_{si(1)}\, \mathcal{H}_{si(2)}\, H_s\, \mathcal{H}_{si(3)}\, \bar{\mathcal{H}}_{si(4)} \cdots H_s\, \mathcal{H}_{si(2m-1)}\, \bar{\mathcal{H}}_{si(2m)}$$

図 7 H_{si} と H_s の配置構成
Fig. 7 Arrangement of H_{si} and H_s.

する．ただし，手順 4 までの時点では，まだ，各パッケージ列 $\mathcal{H}_{si}$ を $i(1)$ から $i(2m)$ のどこに配置するかは特に定めない．その理由としては特徴 1 に述べたように，どのパッケージ列もほぼ同じ長さ l になるので，$i(1)$ から $i(2m)$ のどこに配置しても各 d_{2i} の値に大きな変化はなく，$\mathcal{E}_m$ の値に影響はないと考えるからである．

むしろ d_{2i} に影響を与えるのは $\mathcal{H}_{si}$ ではなく H_{ni} の配置であり，それを次で考える．

4.2.2 課題 2：H_{ni} の配置決定

方針 1 から各 H_{ni} は $\mathcal{H}_{sj}$ や $\bar{\mathcal{H}}_{sj'}$ に挿入せずに，$\mathcal{H}_{sj}$ と $\bar{\mathcal{H}}_{sj'}$ の間，たとえば図 7 の $\mathcal{H}_{si(1)}$ と $\bar{\mathcal{H}}_{si(2)}$ の間に配置する．

そのとき H_{ni} を配置することによって $\{d_{2k} \mid H_{sk} \in \mathcal{H}_{si(1)}, \mathcal{H}_{si(2)}\}$ の各 d_{2k} を s_{ni} だけ増加させている．各 $\mathcal{H}_{si}$ に対し，C_i を式 (7) のように定める．

$$C_i = \sum_{H_{sj} \in \mathcal{H}_{si}} p_{sj}(d_{1j} + s_s) \qquad (7)$$

各 d_{2k} が s_{ni} 増加すると取得時間の期待値 $\mathcal{E}_m$ は $s_{ni}(C_{i(1)} + C_{i(2)})/T_m$ だけ増加する．

以上のことは他の H_{nj} の配置とは無関係に成り立つ．よって C_i の値が 1 番目と 2 番目に小さい $\mathcal{H}_{si}$ を $\mathcal{H}_{si(1)}$ と $\bar{\mathcal{H}}_{si(2)}$ の位置に配置し☆☆，その間に H_{ni} をすべて配置する．こうすることにより H_{ni} の配置による H_{si} の取得時間の増加を最小限に抑えることができる．

順序配置方式（後半部）

手順 5： すべての $\mathcal{H}_{si}$ に対し C_i の値を計算し，C_i の値が 1 番目および 2 番目に小さい $\mathcal{H}_{sj}$，$\mathcal{H}_{sk}$ を見つける．

手順 6： 図 7 のように H_s と $\mathcal{H}_{si}$ を配置する．そ

☆ 証明については各 h_{si} の取得時間を式 (5) として定式化したことにより，文献 11) 400 ページ，定理 S の証明に帰着できる．

☆☆ 1 番目と 2 番目というのは手続きの一例であり，実際には m 個の組，$\{\mathcal{H}_{si(2j-1)}, \bar{\mathcal{H}}_{si(2j)}\ (j = 1, \ldots, m)\}$ のうちのどれでもよい．

の際 $i(1)=j$, $i(2)=k$ とする.

手順 7： $\mathcal{H}_{si(1)}$ と $\tilde{\mathcal{H}}_{si(2)}$ の間に H_{ni} をすべて配置する.

以上に示した順序配置方式はそれぞれの m に対して，準最適なパッケージ配置を見つけることができ，それは 図 6 の構成を持つ．順序配置方式のオーダは手順 1 から $O\big(n_s\log(n_s)\big)$ である.

4.3 課題 3：$\mathcal{E}_m$ を最小にする m の推定方法

この節では $\mathcal{E}_m$ を最小にする m の値 m_{opt} を求める方法について述べる.

単純に考えると，1 から n_s までのそれぞれの m に対して，順序配置方式を適用して配置を決定し，$\mathcal{E}_m$ をすべて求めてしまう方法がある．この場合，順序配置方式の手順 1〜2 は最初に 1 度だけ行えばよく，手順 5〜7 については $O(n_s)$ で済む．よって，特徴 2 より，1 から n_s までのすべての m に対して，順序配置方式を実行したときの処理時間のオーダは式 (8) のようになる.

$$O\left(\sum_{m=1}^{n_s} n_s\log(m)\right) = O\big(n_s\log(n_s!)\big) = O\big((n_s)^2\log(n_s)\big) \quad (8)$$

ここではより低いオーダで各 $\mathcal{E}_m\,(m=1,\ldots,n_s)$ の推定値 $\tilde{\mathcal{E}}_m$ を求める方法を示す.

式 (6) において，配置しないと決まらないパラメータは d_{1i} と d_{2i} であり，この 2 つを配置せずに推定する.

今，$\{H_{si}\}$ が p_{si}/s_{si} の値により昇順で添え字付けされているとすると，$\sum_{j=1}^{i-1} s_{sj}\,(=\Delta_i$ とする) は手順 3 で H_{si} より先に並べられるパッケージのサイズの総和であり，$\Delta_i/2m$ は H_{si} を配置する時点における $2m$ 個の $\mathcal{H}_{si}$ の平均長である．特徴 1 からこの平均長 $\Delta_i/2m$ が d_{1i} の推定値 $\tilde{d}_{1i}$ となる．実際には手順 3 において最短の列に H_{si} を並べるので，d_{1i} の真値はもう少し小さい.

次に d_{2i} の推定法を示す．順序配置方式では C_i の値により，ある特定の H_s の間隔だけが大きくなる．しかし，各 $\mathcal{H}_{si}$ が不明なので H_s の間隔 w は均等であると仮定する．このとき，$S=\sum_{i=1}^{n_s} s_{si}+\sum_{j=1}^{n_n} s_{nj}$ とすると $w=S/m$ となり，d_{2i} の推定値 $\tilde{d}_{2i}$ は $w-s_{si}-\tilde{d}_{1i}$ となる.

以上のようにして d_{1i}, d_{2i} を推定する．$\tilde{d}_{1i}$, $\tilde{d}_{2i}$ および $T_m=S+ms_s$ を式 (6) に代入すると，式 (6) の未知変数は m だけになる．この式を $\tilde{\mathcal{E}}_m$ とする．$\tilde{\mathcal{E}}_m$ の m の係数はあらかじめ計算しておけるので，m が変わるごとに n_s 項の和を再計算するといった必要がなくなる.

以上をまとめて $O\big(n_s\log(n_s)\big)$ で m_{opt} の推定値 $\tilde{m}_{opt}$ を求める方法を示す．なお，手順 1 だけが $O\big(n_s\log(n_s)\big)$ で，手順 2〜5 までの処理は $O(n_s)$ である.

手順 1： $\{H_{si}\}$ を p_{si}/s_{si} の値により昇順で並べる.

手順 2： $\{\Delta_i \mid i=1,\ldots,n_s\}$ を求め，$\{\tilde{d}_{1i},\tilde{d}_{2i} \mid i=1,\ldots,n_s\}$ を算出する.

手順 3： $\tilde{\mathcal{E}}_m$ の m の係数を求める.

手順 4： $\{\tilde{\mathcal{E}}_m \mid i=1,\ldots,n_s\}$ を求める.

手順 5： $\tilde{\mathcal{E}}_m$ を最小にする m を探し，$\tilde{m}_{opt}$ とする.

つまり，$\tilde{m}_{opt}$ を求め，$m=\tilde{m}_{opt}$ の条件のもとで順序配列方式を用いることにより，取得時間を最小化するのに準最適なパッケージ配置を $O\big(n_s\log(n_s)\big)$ のオーダで求めることができる.

5. 評　　価

4 章で提案した方式の有効性を確認するため，統計モデルおよび実測データを用いて取得時間の期待値を求め，比較を行った．なお，アルゴリズムの処理時間については実装方法が大きく影響するので，定量的な評価は行わない.

5.1 統計モデルに基づく評価

まず，統計モデルに基づいて各パッケージのアクセス確率およびデータサイズを生成して，方式の評価を行った．ここでは次の統計モデルを仮定した.

p_{si}, p_{nj}： 文献 3) と同様に Zipf の法則[11] に従うとする．つまり p_{si} が k 番目に大きい値であるとき，p_{si} は $1/k$ に比例する．p_{nj} も同様である.

s_s, s_{si}, s_{nj}： s_{si} の平均値および s_s の値を 25 Kbyte，s_{nj} の平均値を 50 Kbyte とする．つまり h_{si} と h_{ni} のどちらも，それを構成するファイルのパケットサイズの総和の平均値は同じく 50 Kbyte であるとし，これを s_{ave} とおく.

s_{si} と s_{nj} は指数分布に従うものとし，p_{si} と s_{si}，および，p_{nj} と s_{nj} に相関はないとする.

伝送速度は 1 Mbyte/sec (8 Mbps)，放送によって提供されるハイパーテキストの総数は 200 個とした．つまり，共有データをまとめなければデータストリームのサイズは (200 個 ×50 Kbyte =) 10 Mbyte，周期は 10 秒である.

5.1.1 $\mathcal{E}_m$ の推定値 $\tilde{\mathcal{E}}_m$ の検証

方式の比較を行う前に，m_{opt} の推定に用いられている $\tilde{\mathcal{E}}_m$ の値がどの程度正確な推定値であるかを検証する．そのために，実際に順序配置アルゴリズムにより配置を決定して取得時間の期待値 $\mathcal{E}_m$ を算出し，

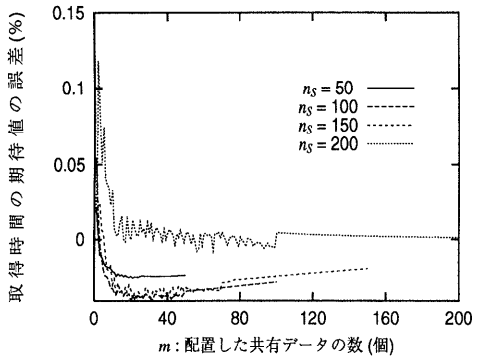

図 8 取得時間の期待値 $\mathcal{E}_m$ 推定値の誤差
Fig. 8 Error of estimated waiting time.

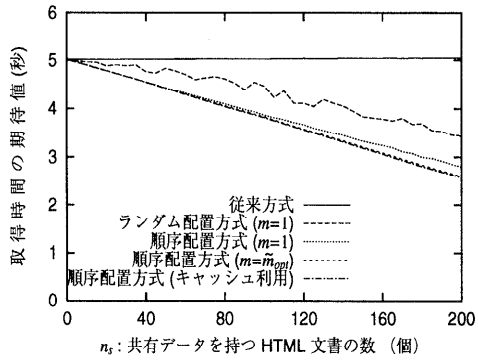

図 9 各方式の取得時間の比較
Fig. 9 Waiting times of package arrangment methods.

$\tilde{\mathcal{E}}_m$ と比較した．この結果を**図 8** に示す．図 8 の縦軸は $(\tilde{\mathcal{E}}_m/\mathcal{E}_m - 1) * 100$ の値である．

図 8 ではデータを共有しているハイパーテキストの割合が高く，また，共有データの配置が少ないほど誤差が増大する傾向にある．しかし，誤差は最大でも 0.12% 以下であり，m_{opt} の推定に問題はないと考えられる．よって以後，m の最適値として $\tilde{m}_{opt}$ を用いる．

5.1.2 提案方式と他の方式の比較

提案方式を評価するにあたり，順序配置方式の $m = 1$ の場合と $m = \tilde{m}_{opt}$ の場合，および以下の 3 つの場合による期待値を算出し評価した．

従来方式： データの共有を考慮しない方式．データが共有されているいないにかかわらず，各 HTML 文書のパッケージは文書を構成するすべてのファイルを含むようにする．

この場合，周期が T_{n_s} (= 10 秒) になり，h_{si} の取得時間の期待値は式 (5) の第 3 項が s_s になった式で表されるため，従来方式の取得時間の期待値 $\mathcal{E}_0$ は式 (6) より以下のようになる．

$$\mathcal{E}_0 = \frac{T_{n_s}}{2} + \sum_{i=1}^{n_s} p_{si}(s_{si} + s_s) + \sum_{j=1}^{n_n} p_{nj} s_{nj} \quad (9)$$

ランダム配置方式： データストリーム中に $H_{si}\,(i = 1, \ldots, n_s)$ と m 個の H_s を，間隔を均等にして配置する．H_{si} の配置順序は無作為に決定する．この方式は，共有データは m 個にまとめるが 4.2 節の方針 1，2 は適用しない，という場合である．取得時間の計算式に変更はない．

順序配置方式（キャッシュ利用）： 順序配置方式の $m = \tilde{m}_{opt}$ の場合と同様にデータストリームを構成するとともに，H_s をキャッシュしたと仮定し，その取得時間を 0 とする場合である．

この場合，h_{si} の取得時間の期待値は式 (5) の第 3 項が 0 になった式で表されるため，取得時間の期待値 $\mathcal{E}_c$ はパッケージを最適に配置した場合の取得時間の期待値 $\mathcal{E}_{opt}$ よりもさらに小さくなり，取得時間の下限を与える．$\mathcal{E}_{opt}$ は実用的な時間では求まらないので，$\mathcal{E}_c$ を代わりに比較対象とする．なお，$\mathcal{E}_c$ は式 (6) より以下のようになる．

$$\mathcal{E}_c = \frac{T_{\tilde{m}_{opt}}}{2} + \sum_{i=1}^{n_s} p_{si} s_{si} + \sum_{j=1}^{n_n} p_{nj} s_{nj} \quad (10)$$

図 9 に，以上にあげた各方式の，共有データを持つハイパーテキストの数 n_s とハイパーテキストの取得時間の期待値の関係を示す．

従来方式はデータの共有を考慮しない伝送方式なので n_s の値にかかわらず取得時間はほぼ $T/2$ で一定である．それに比べ，m を最適化した場合の順序配置方式の取得時間が最短であり，n_s の増加とともに取得時間が大幅に短縮されている．

本論文において今まで示してきた各種の工夫が，それぞれどの程度，取得時間を短縮しているかは各方式の差を見ることで分かる．つまり，ランダム配置方式の時間短縮効果は共有データを 1 つにまとめたことによるものであり，この効果が最も大きい．さらにランダム配置方式と $m = 1$ の順序配置方式の差は，単に共有データを 1 つにまとめるだけでなく，方針 1 と 2 を適用することによって得られた効果である．そして最適な m を用いることで取得時間はさらに改善される．

図 9 から，キャッシュを利用する場合と $m = \tilde{m}_{opt}$ の場合はほぼ一致しており，$\mathcal{E}_{\tilde{m}_{opt}} \simeq \mathcal{E}_c$ であることが分かる．また $\mathcal{E}_{\tilde{m}_{opt}} \geq \mathcal{E}_{opt} \geq \mathcal{E}_c$ であるから，$\mathcal{E}_{\tilde{m}_{opt}} \simeq \mathcal{E}_{opt}$ であり，$m = \tilde{m}_{opt}$ の場合の順序配置方式は，ほぼ最適なパッケージ配置を実現していると考えられる．

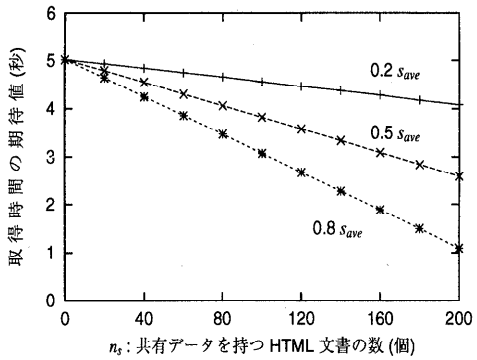

図 10 共有データのサイズと取得時間の関係
Fig. 10 Relation between size of shared data and waiting time.

5.1.3 共有データのサイズ s_s を変化させた場合

$\mathcal{E}_{\tilde{m}_{opt}} \simeq \mathcal{E}_c$ から時間短縮効果 $\Delta\mathcal{E}$ は式 (11) のように表される.

$$\begin{aligned}\Delta\mathcal{E} &= \mathcal{E}_0 - \mathcal{E}_{\tilde{m}_{opt}} \\ &\simeq \mathcal{E}_0 - \mathcal{E}_c \\ &= \frac{s_s}{2}(n_s - \tilde{m}_{opt}) + s_s \sum_{i=1}^{n_s} p_{si} \qquad (11)\end{aligned}$$

式 (11) の結果と $n_s \gg \tilde{m}_{opt}$ および $\sum_{i=1}^{n_s} p_{si} \leq 1$ から, 提案方式の時間短縮効果に大きな影響を与えるパラメータは n_s と s_s であり, その効果は n_s と s_s に比例することが分かる. n_s に比例することは図 9 においてすでに示したとおりである.

s_s については, s_s の値を $0.2 \cdot s_{ave}$, $0.5 \cdot s_{ave}$, $0.8 \cdot s_{ave}$ と変えたときの, 順序配置方式 ($m = \tilde{m}_{opt}$) の取得時間の期待値を**図 10** に示す. この結果から s_s の値にほぼ比例して, 時間短縮の効果も大きくなることが示されている.

5.2 実データによる評価

この節では実際にインターネット上で公開されている HTML 文書の集合を使って提案方式の評価を行う.

5.2.1 解析対象サイトと共有データの決定

HTML 文書の取得に関しては, 以下のように 2 つのサービスを想定して 2 種類のカテゴリーの WWW サイトを選択した.

- 博物館, 展示会などにおいて多数の来場者に展示品の解説などを携帯端末で提供
  → 展示系のサイト (カテゴリー 1)
- ニュースや交通情報などを公衆に配信
  → 配信系のサイト (カテゴリー 2)

実際の HTML 文書では共有されるデータファイルは複数存在し, 各データファイルの共有関係も異なっている. そのため, 共有データのパッケージ H_s に含めるデータファイルを適切に選び出す必要がある. 5.1 節で述べたように, n_s と s_s が大きいほど, 提案方式の時間短縮効果は大きい. そこで本節の実験ではいわゆる強欲 (greedy) な戦略に基づいた以下の手続きにより共有データのパッケージ H_s を構成した.

> $n_s \times s_s$ の値を最も大きくするデータファイルを順次選択していき, どのデータファイルを選択しても $n_s \times s_s$ の値を大きくすることができなくなったとき, それまで選択したデータファイルをパッケージにして H_s とする.

表 1 カテゴリー 1 のサイトのデータサイズ
Table 1 Data sizes of WWW sites in category 1.

サイト	T_{all}	n_{all}	n_s/n_{all} (%)	s_{ave}	s_s/s_{ave} (%)
1	19547	470	20.0	41.6	102.4
2	3856	51	98.0	75.6	37.5
3	3140	34	100.0	92.4	27.8
4	25238	165	63.6	153.0	31.3

5.2.2 展示系のサイト (カテゴリー 1) のデータ

カテゴリー 1 のサイトに関しては以下の 2 つの理由により総当たり的な調査は行わなかった.

- インターネット上には展示品紹介等のコンテンツを持つサイトが少ない
- 展示系のサービスに絞っても, さらに様々なサービスが存在する

この項では仮想展示会, 博物館, 美術館, 動物園の 4 つのサービスから比較的多数のデータファイルが共有されているサイトを選択し, 解析結果を示す.

(1) Virtual Open House (http://vexpo.mm.rd.nttdata.co.jp/V_EXPO/indexlist.html)
(2) 横浜市歴史博物館(http://www.via.or.jp/~imnet/yokohama/index.html)
(3) 東京都現代美術館(http://www.via.or.jp/~imnet/mot/index.html)
(4) 上野動物園 (http://www.999.com/uenozoo/ index2.html)

サイト 1 に関しては HTTP のサーバプログラムのアクセスログから各ページのアクセス頻度を算出し, アクセス確率として用いた. 利用したアクセスログは 1 年分 (44812 ヒット) のデータである. サイト 2〜4 に関してはアクセスログを入手できなかったので 5.1 節と同じ仮定を用いた.

各サイトごとの主なデータを**表 1** に示す☆. T_{all} は全ページのサイズの総和を, n_{all} は全ページ数をそ

☆ サイト 4 は規模が大きいため上記 URL から 2 回以内で移動可能なページのみを対象とした.

表 2 カテゴリー 1 のサイトの評価結果
Table 2 Evaluation results of WWW sites in category 1.

サイト	$\mathcal{E}_0$ (秒)	$\mathcal{E}_{\tilde{m}_{opt}}$ (秒)	改善率 (%)	$\tilde{m}_{opt}$
1	9.82	7.87	19.9	2
2	2.00	1.32	33.8	3
3	1.66	1.26	24.1	3
4	12.8	10.3	19.1	3

れぞれ示す．各データサイズの単位は KB である．たとえばサイト 2 では 28379 byte のデータが，全体の 98.0% (50/51) のページに使われている．共有データの内訳はロゴ，ボタン，ヘッダ，フッタの画像ファイルである．これらは 1 ページの平均サイズ (75605 byte) の 37.5%を占めている．

これらのサイトに対して取得時間の期待値を算出した結果を**表 2** に示す．$\mathcal{E}_0$ は，式 (9) で表される従来方式による取得時間の期待値であり，改善率とは $(1-\mathcal{E}_{\tilde{m}_{opt}}/\mathcal{E}_0)*100$ の値である．データの共有を考慮することにより，20%程度の時間短縮が図られていることが分かる．特に，データの共有率，共有データのサイズの両方が比較的大きかったサイト 2 においては 30%を超える改善が図られている．

5.2.3 配信系のサイト（カテゴリー 2）のデータ

近年，新聞社のサイトは WWW による記事の公開を充実させているため，カテゴリー 2 のサービスを行う WWW のサイトの例として適当であると判断した．そこで 1998 年 5 月 11 日の時点において以下の条件を満たす 18 個のサイトを解析の対象とした．

(1) Yahoo!^{TM} Japan☆において 'メディアとニュース：新聞：新聞社' の分類項目に含まれる．
(2) ニュースを掲載し，週 1 回以上更新している．
(3) データを正しくダウンロードできる☆☆．

表 1 および表 2 において示した項目と同じ項目に関して，18 個のサイトの平均値を**表 3** に示す．n_s/n_{all}，s_s/s_{ave} および改善率については，個々のサイトの値を**図 11** に示し，他の値に関しては付録 A.2 の表 4 に示した．図 11 を見ると，n_s/n_{all} または s_s/s_{ave} のどちらか片方の値が小さいために，改善率が低い値にとどまっていることが分かる．

5.2.4 考　　察

5.1 節で示したように，本論文で提案する方式の有効性は，データ H_s を共有する HTML 文書の割合 (n_s/n_{all}) および H_s のサイズの割合 (s_s/s_{ave}) にか

☆ Yahoo! Japan (http://www.yahoo.co.jp/) は Yahoo Japan Corporation が運営する WWW サイトである．
☆☆ 我々のシステムの不具合でダウンロードに失敗したサイトもある．

表 3 カテゴリー 2 のサイトのデータサイズおよび評価結果
Table 3 Data sizes and evaluation results of WWW sites in category 2.

T_{all}	n_{all}	n_s/n_{all} (%)	s_{ave}	s_s/s_{ave} (%)
11986.9	599.9	39.9	31.0	40.1
$\mathcal{E}_0$ (秒)	$\mathcal{E}_{\tilde{m}_{opt}}$ (秒)	改善率 (%)	$\tilde{m}_{opt}$	
11.5	10.6	7.98	2.61	

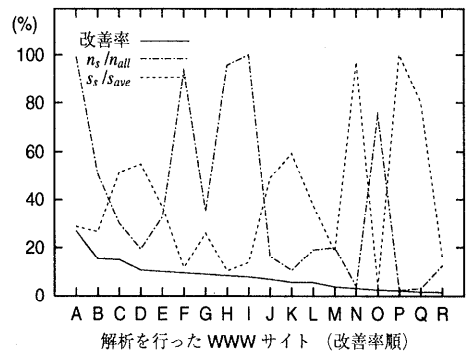

図 11 カテゴリー 2 の各サイトにおける取得時間の改善率
Fig. 11 Progression of waiting time of each WWW site in category 2.

かっている．5.2 節の実験では比較的簡単な手続きで H_s を構成したにもかかわらず，n_s/n_{all} と s_s/s_{ave} の両方の値が大きな H_s を持つサイトを複数確認することができた．

n_s/n_{all} と s_s/s_{ave} の両方の値が高く，改善率の高いサイトは，サイト全体としてページのデザインが統一性を持っているという傾向が見られた．つまり，デザインを統一しデータ共有を積極的に行うようにすれば，提案方式により，限られた帯域でより多くの HTML 文書を提供することができるといえる．

テレビ CM の製作者が限られた時間にできるだけ多くの情報を詰め込もうとするように，データ放送においても提供者はつねに限られた帯域で多くの情報を提供する必要に迫られる．そのような状況では，やはりコンテンツ製作の段階から帯域の節約を意識する必要がある．現状では帯域を節約するために，色数を減らして画像データのサイズを抑えるといった工夫が行われているが，本論文において提案した方式により，HTML 文書の提供者にはもう 1 つ別のアプローチが開かれたといえる．

6. ま と め

本論文では放送による HTML 文書の伝送方式について述べた．データの共有がある場合に，利用者の待ち時間を短縮するため，データの共有関係を考慮した

放送データの配置決定方式として順序配置方式を提案した．さらに，この方式を基に共有データの配置数を効率良く最適化する方式を示した．机上評価の結果，データが共有されている場合，従来方式よりも，順序配置方式のほうがパッケージ取得のための取得時間の期待値が小さくなることを示した．その効果はデータを共有するHTML文書の数および共有データのサイズにほぼ比例して大きくなることが明らかになった．さらに，実際にインターネット上で公開されているHTML文書に対して提案方式を適用し，その有効性を示した．

今後は，本論文で示した放送の伝送効率を基にしてキャッシュやオンデマンドとの最適な組合せを明らかにし，放送・オンデマンド統合型システムとしてレスポンスの向上を目指していく必要がある．また，提案方式の有効性はデータの共有度にかかっているが，共有度を上げるための課題としては以下の2つがあげられる．

- データの共有関係からHTML文書の集合を適切に分割する方式の確立．本論文では，あるデータを共有するHTML文書の集合とそれ以外のHTML文書の集合という分け方しかしていないが，実際にはデータAを共有する集合，データBを共有する集合 ... といったように多数の集合を扱うことが考えられる．
- 伝送システムの部分だけでなく，コンテンツ作成段階からデータ共有による放送帯域節約のための枠組みを作っていく必要がある．

本論文ではHTML文書の間に張られたリンク構造を考慮していない．具体的にはHTML文書 h_1 から h_2 に対しハイパーリンクが張られ，かつ，利用者の多くがそのリンクをたどるのであれば，h_1 のパッケージと h_2 のパッケージを，データストリーム上において近くに配置したり，まとめて1つのパッケージにしたりすることが考えられる．このような方式の拡張にはHTML文書のアクセス遷移確率を考慮する必要があり，今後の課題である．

参考文献

1) Imielinski, T. and Viswanathan, S.: Adaptive Wireless Information Systems, Technical Report, DCS-TR-312, Rutgers University (1994).
2) 箱守 聡，田辺雅則，石川裕治，井上 潮：放送型通信/オンデマンド型通信を統合した情報提供システム，DiCoMoワークショップ予稿集，pp.55–60 (1997).
3) Swarup Acharya, M.F. and Zdonik, S.: Disseminationg Updates on Broadcast Disks, *Proc. 22nd VLDB Conf.*, Mumbai (Bombay), India, pp.354–365 (1996).
4) 田辺雅則，石川裕治，箱守 聡，井上 潮：放送/オンデマンド統合型情報提供システムMobiCasterにおけるデータ提供方法の決定方式，DiCoMoワークショップ予稿集，pp.499–506，情報処理学会 (1998).
5) Hameed, S. and Vaidya, N.H.: Log-time Algorithms for Scheduling Single and Multiple Channel Data Broadcast, *Proc. MOBICOM '97*, Budapest Hungary, pp.90–99, ACM (1997).
6) Housel, B.C. and Lindquist, D.B.: WebExpress: A System for Optimizing Web Browsing in a Wireless Environment, *Proc. MOBICOM '96*, Rye NY USA, pp.108–116, ACM (1996).
7) 高田 隆（編）：次世代ディジタルテレビの全貌，日経BP社，東京都 (1997).
8) 角谷和俊，楠見雄規，岡村和男：デジタル放送インタラクティブ・データ配信のためのカルーセル型送出方式DVXとその応用，*Proc. ADBS '97*, Tokyo, Japan, pp.23–30 (1997).
9) 石川裕治，田辺雅則，箱守 聡，井上 潮：放送型情報提供システムにおけるハイパーテキストの伝送方式，第54回情報処理学会全国大会論文集（3），pp.593–594 (1997).
10) 石川裕治，田辺雅則，箱守 聡，井上 潮：ハイパーテキスト間のデータ共有を考慮した放送型情報提供方式，データベースシステム研究報告，97-DBS-113, pp.251–256，情報処理学会 (1997).
11) Knuth, D.E.: *Sorting and Searching*, The Art of Computer Programming, 1st edition, Vol.3, Addison Wesley (1973).

付 録

A.1 式(5)の導出過程

図12のように座標軸を定めて E_{si} の計算式を導く．h_{si} を取得するには H_s と H_{si} の2つのパッケージを取得する必要がある．H_s と H_{si} の両方を取得するには次の2つの場合が考えられる．

場合1： 座標 x $(0 < x \leq s_s + d_i)$ 地点において端末がアクセスを開始したとすると，H_{si} の先頭が

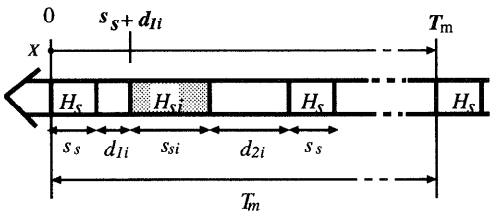

図12 データストリームに対する座標軸の設定
Fig. 12 Axis defined on data-stream.

表 4 カテゴリー 2 のサイトのデータサイズおよび評価結果
Table 4 Data sizes and evaluation results of WWW sites in category 2.

サイト	T_{all}	n_{all}	s_{ave}	$\mathcal{E}_{\tilde{m}_{opt}}$	$\tilde{m}_{opt}$
A	9581.5	119	80.5	3.56	4
B	15406.2	991	15.5	6.55	1
C	11913.0	321	37.1	5.10	3
D	18696.8	798	23.4	8.39	2
E	2310.4	43	53.7	1.08	2
F	746.7	46	16.2	0.349	4
G	10422.5	535	19.5	4.77	4
H	901.2	46	19.6	0.431	5
I	164.9	7	23.6	90.6	3
J	859.8	68	12.6	0.412	1
K	4603.9	173	26.6	2.20	1
L	492.2	42	11.7	0.243	1
M	98730.8	5447	18.1	47.6	5
N	3554.4	155	22.9	1.74	1
O	4388.1	65	67.5	2.20	6
P	23463.9	1744	13.5	11.5	1
Q	7336.9	100	73.4	3.66	1
R	2190.1	98	22.3	1.10	2

現れるまで $s_s + d_{1i} - x$ だけ待たねばならない．その後，$s_{si} + d_{2i} + s_s$ 時間かけて H_{si}, H_s の順にデータを取得して終了する．

つまり取得時間は $E = 2s_s + d_{1i} + d_{2i} + s_{si} - x$.

場合 2： 座標 x $(s_s + d_{1i} < x \leq T_m)$ 地点においてアクセスを開始したとすると，H_{si} の先頭が現れるまで $T_m - x + s_s + d_{1i}$ だけ待たねばならない．この間に少なくとも 1 度 H_s を取得できる．その後，s_{si} 時間かけて H_{si} を取得して終了する．

つまり取得時間は $E = T_m + s_s + d_{1i} + s_{si} - x$.

利用者の要求はランダムに発生する．そのため，データストリームのどの地点からも一様な確率でアクセスが開始される．つまり，微小区間 dx からアクセスが開始される確率は dx/T_m．よって E_{si} は $E \times dx/T_m$ を x について 0 から T_m まで積分することで求められる（式 (12)）．

$$
\begin{aligned}
E_i &= \frac{1}{T_m}\int_0^{s_s+d_{1i}} (2s_s + d_{1i} + d_{2i} + s_{si} - x)dx \\
&\quad + \frac{1}{T_m}\int_{s_s+d_{1i}}^{T_m} (T_m + s_s + d_{1i} + s_{si} - x)dx \\
&= \text{式 (5)} \qquad (12)
\end{aligned}
$$

A.2 カテゴリー 2 として選択したサイトの詳細

5.2.3 項において解析対象とした WWW サイト A ～R のデータサイズおよび評価結果を**表 4** に示す．改善率，n_s/n_{all}，s_s/s_{ave} の値に関しては図 11 に示している．

(平成 10 年 8 月 31 日受付)
(平成 11 年 4 月 1 日採録)

石川 裕治（正会員）

昭和 46 年生．平成 8 年東京工業大学大学院理工学研究科システム科学専攻修士課程修了．同年 NTT データ通信（株）入社．現在（株）NTT データ情報科学研究所勤務．モバイルコンピューティングの研究開発に従事．

田辺 雅則（正会員）

昭和 42 年生．平成 4 年京都工芸繊維大学大学院修了．同年 NTT データ通信（株）入社．現在（株）NTT データ金融システム事業本部勤務．分散処理，エージェント技術，モバイルコンピューティングの研究開発に従事．

箱守 聰（正会員）

昭和 38 年生．昭和 61 年名古屋大学工学部電気工学科卒業．昭和 63 年同大学大学院工学研究科修士課程修了．同年 NTT データ通信（株）入社．現在（株）NTT データ情報科学研究所勤務．分散処理，モバイルコンピューティングに興味を持つ．ACM，IEEE 各会員．

井上 潮（正会員）

昭和 28 年生．昭和 50 年名古屋大学工学部電気工学科卒業．同年日本電信電話公社入社．平成 7 年 NTT データ通信（株）勤務．現在（株）NTT データ情報科学研究所主幹技師．工学博士．オンライン情報検索システム，データベース管理システム，データベースマシン，マルチメディアデータベース，モバイルコンピューティングの研究開発に従事．電子情報通信学会，IEEE，ACM 各会員．