データベースシステム 109－43
（1996. 7. 25）

HTMLデータの総合管理システム

小林 伸行　　北川 文夫
MSR　岡山理科大学理学部

インターネット上での情報提供は、プラットホーム独立の情報ブラウザの普及や、データ記述がHTMLという簡易な指定だけでできることから、WWWを使って行われることが多くなっている。

WWWのデータ単位を構成するページは、HTMLで書かれたテキストであり、1つのファイルを構成する。ページは他のページへのアンカーを持つことができるので、ハイパーテキストを構成する。したがって、HTMLファイルを保存するための構造は論理的なリンク構造と同じにすることは困難である。つまり、HTMLファイル群を管理することを困難である。特に、既存のデータの削除や新しいデータの追加をしたときに、リンク構造をある程度保持することさえも、非常に難しい。

本稿ではでは、まず、既存のいくつかのWWWサイトのリンク構造パターンの分析を示し、次に、データ管理のために、リンク構造をディレクトリの物理構造にマッピングする方法とそれを用いた管理システムの概要を示す。

A design for HTML-file Management System

Nobuyuki Kobayashi　　Fumio Kitagawa
MSR　Okayama University of Science

It is increasing to offer some useful information for many people using WWW, because Web-browsers are platform independent and Web-data are easy to constructed using by HTML. An unit of WWW-data which is a 'page' is a text file written by HTML. A page is able to have an anchor for the another page, then a page having some anchor is called a node of Hyper-text. It is difficult to make a physical structure for saving hyper-text with the same structure of logical links. That is the same meaning of difficulty of managing HTML files. Especially, it is too difficult to keep the relation of link when deleting some existing pages or adding some new pages. In this paper, we show some patterns of some page structures for some WWW-site. Then, we propose a structure of managing HTML files, and its system.

1 はじめに

近年の急速なインターネットの普及に伴い，手軽に様々なホームページにアクセスできるようになった．また，日本でも多くのWWWサイトがホームページを開設しており，様々な趣向をめぐらせたものも数多く存在する．しかし，各WWWサイトを回るとページの管理が不完全で，特にページの追加や削除により，リンクエラーが起こるという状況に直面する．これはページを作成するためのHTMLエディタは多くあり，ページ自体の作成は容易にできるのだが，各サイトのリンク等の構造を管理できるエディタは，まだそんなに多くは存在しない．そこで，サイト内でリンクエラーが起きないようなホームページの作成を簡単に行うためにには，各WWWサイトおけるそのWWWサイト内のリンク構造を管理するシステムの開発が必要不可欠である．

ところで，一般にページのリンクの方法を考えると，多くは一方向のリンクがクモの巣状に広がっていると考えられる．つまり，各リンクで双方向にデータのやり取りを行っているのではなく，一方向で無秩序に広がっている．しかし実際に，あるWWWサイト内のリンクを考えると，管理者が管理しやすいように意外に単純な構造で作成していることが多い．そこで本報告では，まず現在運営されているサイトのリンク構造をいくつかのパターンに分類し，その特徴やより良い管理方法を考え，これに基づいたシステムを提案する．

2 WWWサイト内のリンク構造

まず，ここでは既存のWWWサイトにおけるリンク構造の分類を行う．例えば，日本語と英語，テキストとグラフィック等の二重化と，リンク構造を論理的な木構造として考えることができ，リンクと木構造の階層の関係について，木構造の階層が同一階層または下層に対してのリンク関係を内容構造とし，上層に対してのリンクを戻り方とする．これら３つのパターンは次のように分類される．

$$\begin{array}{ccccc} \text{二重化} & \times & \text{内容構造} & \times & \text{戻り方} \\ \left(\begin{array}{c}\text{完全二重化}\\ \text{部分二重化}\\ \text{非二重化}\end{array}\right) & \times & \left(\begin{array}{c}\text{子レベルのリンク}\\ \text{孫レベルへのリンク}\\ \text{同レベルの前後へのリンク}\end{array}\right) & \times & \left(\begin{array}{c}\text{トップページへ}\\ \text{直系のページへ}\\ \text{直前のページへ}\\ \text{トップページのリンク先へ}\\ \text{なし}\end{array}\right) \end{array}$$

2.1 二重化の分類について

二重化とは，例えば日本語と英語，グラフィックとテキスト等でリンク構造を二重に作成する為によく使われる構造である．一般には次の３種のパターンで分類される．

- 完全二重化

  対応したページ間すべてに双方向のリンクがある．

- 部分二重化

  対応したページが存在し，トップレベルのページにのみ双方向のリンクがある．

- 非二重化

  対応したページが存在しない．

2.2 内容構造の分類について

WWWサイト内のリンクを論理的な木構造としてとらえた場合のリンクと木構造との関係が，同一階層または下層に対してのリンクの分類で，以下の３つのパターンがあり，一つに定まらず複合する場合がある．

- 子レベルへのリンク最も基本的なリンク，サブディレクトリにある．
- 孫レベルへのリンク

  子レベルのHTMLのリンク先の中から，主だったものへのショートカットが用意されている．
- 同レベルの前後へのリンク

  親レベルのリンク先の中から，自ページの前あるいは後ろとなるページへリンクする．

2.3 戻り方の分類について

WWWサイト内のリンクを論理的な木構造としてとらえた場合のリンクと木構造との関係が，上層に対してのリンクの分類で，次の４種類がある．この戻り方はWWWサイト内での移動性の向上のため，よく使用されているものでありその性能上複合することが多い．

- トップページへ

  WWWサイト内のトップページへ戻るためのリンクが用意されている．
- 直前のページへ

  トップページから現在のページまでの経路の直前のページへのリンクが用意されている．
- 直系のページへ

  トップページから現在のページまでの経路のすべてのページへのリンクが用意されている．
- トップページのリンク先へ

  トップページの主なリンク先へのリンクが用意されている．トップページへと複合する．
- なし

  ブラウザの機能の使用を前提にしている．

2.4 実際のページの例

ここでは既存のWWWサイトで，これまで述べてきた分類の例をを紹介する．
完全二重化 × 子レベルのリンク × トップページへ :(図1) NTT等
部分二重化 × 孫レベルのリンク × 直系のページへ :(図2) Yahoo等

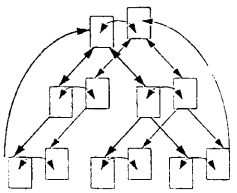


図1

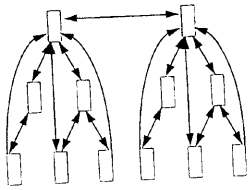


図2

3 HTMLファイル管理システム

3.1 HTMLファイル管理システムの構造

このように現在運営されているWWWサイトは，基本的なリンク構造として木構造を使用している．ところが，ファイルの物理構造とは必ずしも一致していないため保守管理が困難になっているところが多いと思われる．保守管理を容易にするためにはWWWサイトのリンク構造をディレクトリの物理構造に対応させるのが自然であると考えられる．そこで我々は論理的なページ間のリンク構造から，木構造への対応づけを考え，その木構造をメインルートと呼ぶことにする．メインルートをディレクトリ構造に反映させファイル管理することで，ファイル管理が簡単になると考えられる．メインルート以外のリンク関係については付加的なリンク情報として管理することにする．メインルートに登録された各ページは基本的に１つのディレクトリに１つのファイルにすることにする．木構造を意識して作成したHTMLファイル群ではない場合の，メインルートはおよそ次のように決める．

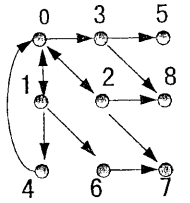
- 例 木構造の決定手順
  ここではリンク情報から，木構造を決定する手順を説明する．メインルートの各節にるページはノードを代表し，そのファイル名をノード名とする．ノードとはいくつかの関連あるページの集合である．

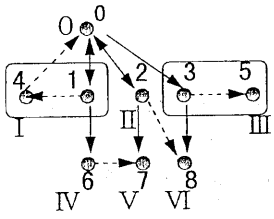
(0) HomePageから探索する．

(1) リンクのあるページにすべて適当な通し番号をふる．

(2) ページaからのみ参照され，子を持たないページはaと同一ノードとする．

(3) ホームページからのリンクの深さが同一で直前のページが複数の場合，ページ番号の若い方をメインルートにする．





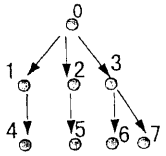
メインルートを決定後，各代表ページに対応したノード番号を振る．

3.2 HTMLファイル管理システム

このようにして木構造を決めたHTMLファイル群は次の管理情報により管理する．

(1) 構造データ

メインルートの論理構造と各ノードの物理的位置情報を管理するものであって次のような木構造を持つ．



(2) ディレクトリ情報

HTMLの各ノードのディレクトリ情報は次のように管理する.

ノード番号	ディレクトリ名	代表ファイル名
0	/home/www/	index.html
1	/home/www/wahtsnews/	index.html
2	/home/www/pc/	index.html
⋮	⋮	⋮

図3 ディレクトリ情報テーブル

(3) ファイル位置情報

サブノードも含む各HTMLファイル管理に必要な情報は次のように決定する.

ページ番号	ファイル名	ノード番号
0	index.html	0
1	index.html	1
2	index.html	2
3	wn0701.html	1
4	wn0702.html	1
⋮	⋮	⋮

図4 ファイル位置情報テーブル

(4) リンク情報

リンク情報は次のような表で管理する.

ページ番号	ページ番号
0	1
1	0
0	2
⋮	⋮

図5 リンク情報テーブル

ここで，第1列のページから第2列のページへのリンクがあることを示す.
外部サーバへのリンクは，ここに記述するが，当該サーバのファイル管理には無関係なので，ここでは扱わないことにする.

(5) データ操作

例えば，ディレクトリ名称を変更する場合は次のような手順になる.

1) 図3のディレクトリ名を変更が発生する.
2) 図4の当該ディレクトリのページすべてに対し，物理的移動がある.
3) 図5の当該ページ番号へのリンクに対し，リンクもとのHTMLファイル中のアンカーの変更が発生する.

3.3 システムの機能

以上のような管理ファイルによって，HTMLファイル群を管理するシステムを提案する．ここでは，その機能について説明する．システム機能には診断系，表示系，更新系の3つがあり，それぞれ次のようになる.

(1) 診断系

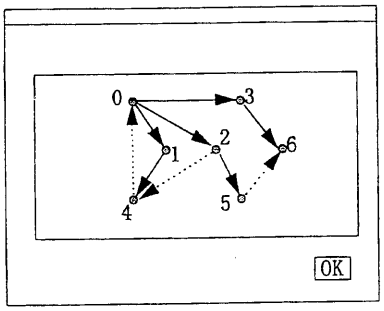
既存のHTMLファイル群から，リンク情報をもとにサブディレクトリ化への支援を行いながら上述の管理ファイルを生成する．

ユーザインタフェースは次のようなものを考えている．

1) リンク情報が示され，システムがつけたメインルートが色を分けて表示される．

2) ユーザは変更が必要なリンクをクリックし，構造を変更することができる．

3) OKを押すと，管理情報を生成し，以後システム上での操作が可能になる．



(2) 表示系

各管理情報を表示する．ツリー表示やリンク情報をツリーに付加した形などをダイアグラムで表示する．

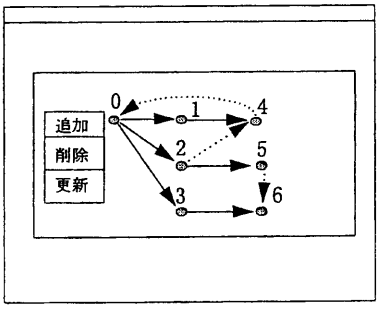
(3) 更新系

ここでの操作には，リンク情報の変更が生じるが上述のデータ操作例のようにリンク情報の整合性を保つことができる．以下のような操作がある．

- 新しいページの追加
- 既存のページの削除
- リンクの付け替え

ユーザインタフェースは次のようなものを考えている．

既に作成されたファイルをツリー上にあるノードに加えたい時ノードをクリックすると，ファイル名を聞いてくるので，作成したいファイル名を入力する．ノード中のどこにアンカーを付けるかを示された後，この新しいページの中のアンカーを付ける画面に変わり，いくつかの場所を指定することができる．



3.4 実現方法

HTMLファイルのサーチはテキストファイルに対しての編集に有利なPerlで記述し，視覚化をJavaを使って行うことを予定している．

4 終わりに

今回はHTMLファイル管理が大変だという動機からシステム設計を行った．このシステムの実現の課題は論理的なリンク構造をうまくツリー構造にすることができるかという点である．うまくツリー構造にできれば，今後は図やCGIに対応することを予定している．