データベースシステム 125－33
（2001．7．18）

半構造データの差異発見・比較のためのスキーマ生成機構

小島岳史[†] 清光英成[††] 田中克己[†††]

本論文では，Web 上に存在する半構造データ間の差異発見・比較を行うためのスキーマ生成機構と半構造データ比較エンジンを提案する．本論文では同じ，あるいは類似の性質を持つデータを共通の側面から捉えることで得られる情報を同属情報と呼び，同属情報の構造及び内容の類似性から共通のスキーマを生成する．生成した共通のスキーマに基づいて同属情報間の差異発見・比較を行う．upper schema を用いた共通する属性の値の比較と，lower schema を用いた構造的な比較を行う．発見した差異は，視覚化されてユーザに提示される．これにより，インターネットによる情報収集にかかる負担を軽減するという結果が期待できる．

Schema Generation for Difference Discovery and Comparison of Semistructured Data

TAKESHI KOJIMA,[†] HIDENARI KIYOMITSU[††] and KATSUMI TANAKA[†††]

In this paper, we suggest a comparison engine and schema generator for semistructured data. Our comparison engine discovers differences between semistructured data. The schema generator makes common schema of OEM graphs from semistructured data, semistructure data can be comparable. We assume that data with the same or similar property belong to the same category, we call these data "same field data".

Difference between same-field data can be discovered from their OEM graphs by using its common schema. We use upper schema for discovering the difference on attribute values, lower schema for discovering that on structure. Moreover, visualization of difference are represented by the colored OEM graphs.

1. はじめに

近年，情報検索技術の発展により，ユーザの求める情報を Web 上から効率よく探すことが出来るようになった．しかし，検索エンジンによって示された各 Web ページが利用者の要求を満たすかどうかは，ユーザ自身が手動で一つ一つ判別しなければならない．なぜなら，Web 上に存在する多くのデータは構造がまちまちな半構造データであり，データ間の差異発見と比較を自動的で行う機構はない．各 Web ページの情報を手動で比較したり，違いを見つけることは，ユーザにとって大きな負担である．

そこで，本研究は半構造データの差異発見と比較を行うための比較エンジンと．半構造データを比較するために複数の半構造データの共通スキーマを導出する機構を提案する．半構造データの特徴はデータの属性値とデータの構造で表現できると考えられる．例えば，複数のプリンタの製品仕様というデータの比較を考えると，複数の製品が有する機能について性能の差を調べることが属性値の比較で，特定の製品のみが持つ機能を見つけることは構造の比較である．

本研究では Suciu らの upper schema と lower schema[1] を導入している．upper schema は同属情報全てが持つ属性のみで構成されたスキーマであるため，属性値の比較が出来る．また，lower schema は全ての同属情報が持つ全ての属性を持つスキーマであるため．属性値が空値かどうかを調べることで，その属性構造の比較が出来る．これら二つのスキーマは OEM をボトムアップにたどることで生成する．

さらに，データの差異を視覚的に表現するために個々の同属情報の OEM と lower schema の OEM に色をつけて表示する．属性値や構造の差の大きさは，OEM のオブジェクトにつけられた色の濃さで表現する．

2. 関連研究

2.1 二次情報サイト

二次情報サイトとは，サイトの管理者が複数のページに記載されているデータを収集・分析し，一つの形

† 神戸大学大学院自然科学研究科
Graduate School of Science and Technology, Kobe University
†† 神戸大学国際文化学部
Faculty of Cross-Cultural Studies, Kobe University
††† 京都大学大学院情報学研究科
School of Informatics,Kyoto University

式にデータをまとめなおしてユーザに提示するサイトである．このようなサイトの例として，スマートバイヤーズガイド [リブラ][2)]，DealTime < ディールタイム >[3)]，eArena[4)]，価格.com[5)] がある．これらのサイトはオンラインショッピングサイトや商店の販売情報のサイトに記載されている商品の価格を収集し，比較してユーザに提示している．

この形式の利点としては，データの形式が統一されているので，比較を自由に行えることである．ただし，データの収集・分析と統一データの作成には，サイト管理者が非常に大きな負担を負う．

二次情報サイトはユーザの負担をサイト管理者が負担しているものといえる．本研究は，ページ閲覧者，サイト管理者にかかわらず，人が Web からの情報収集をするのに伴う負担を軽減するのが目的である．そのため，データの比較が容易になるように自動的にデータを分析する．

2.2 メタ検索サイト

メタ検索サイトは，複数の検索機能を持つサイトへ，同時にキーワードの投入を行う．そして，検索の出力は一まとめにしてユーザに提示される．複数のサイトからの出力をまとめる方法についてはいくつかの研究がある．

冨田らは，オンラインショップサイトの検索結果のページから商品情報を抽出するために，商品検索システム RBIMD[6)] を提案した．これは，同一ショップの商品検索結果の HTML 文書中に出現する商品記述の属性値の並び順や HTML タグの用い方は一定である，という性質を利用している．この性質は「商品情報記述パターンの一定性」と呼ばれている．RBIMD は，前もってサイトごとの商品情報記述パターンの一定性を分析しておくことで，HTML 文書から必要なデータの属性を取り出す．そして，取り出した情報を一覧表にして出力する．同様の手法によって商品情報を収集する研究として，Doorenbos らの shopbot[7)] がある．これは，商品情報を収集するだけでなく，複数のオンラインショッピングサイトの値段比べをするシステムを提案している．このシステムは実際に商品価格比較サイト Excite[8)] で使われている．

メタ検索サイトは前もってサイトごとの検索結果の HTML 文書を分析しておく必要があるが，人が情報を収集するのに伴う負担は大幅に軽減されていると言える．しかし，提示されるデータは詳細なものではなく，より詳しい情報を得るためには，リンクをたどって複数のページを閲覧する必要がある．また，データの比較はすべてのデータに共通する属性のみである．本研究では，共通の属性以外の属性をデータごとの特異性と考え，差異発見をする．

3. 半構造データの収集と比較

本研究は，類似な性質を持つデータが複数あるときに，ユーザが個々のデータの特徴を理解することを支援する事を目的とする．また，ユーザがデータの特徴を容易に理解出来るように，あるデータとその他のデータとの間の差異を提示する．半構造データ間の差異は，属性値の比較と構造の比較から得ている．以下に，類似な性質を持つデータの収集方法と比較方法を述べる．

3.1 同属情報の自動的な収集

Web 上または Web ページ上に存在する半構造データを自動的に収集するために，同属情報という考え方を導入する．本論文中において同属情報とは，同じ，あるいは類似の属性集合によって，対象の特徴を表現しているデータのことを言う．例えば，何種類かのプリンタという対象について，それぞれの性能を示す仕様のデータは同属情報である．

本研究では，Web 上の同属情報である半構造データには，次の二つの性質があると想定する．

- 同属情報である半構造データは，データ元が異なっていても，データの記述は同じ，あるいは類義・同義の単語・表現が使われる．
- 同属情報である半構造データは，データ元が異なっていても，その同属情報に特有の共通な構造がある．

以上の二つの性質から，HTML データ，Web ページに記載されているデータ，XML データ全体または一部などの同属情報である半構造データを判別し，取得することが出来ると考える．

例えば，プリンタの仕様という同属情報は，表によって Web ページ上に記載されている．そこで，<table>，<tr>，<td> タグなどで囲まれた部分を取得する．また，すでに他のデータ元からプリンタの仕様のデータが得られているのならば，そのデータ内に頻出する単語やその類義語・同義語も利用できる．

3.2 同属情報の差異発見と比較

同属情報は半構造データであるために，データごとに構造が異なっており，単純には属性同士を比較することは出来ない．また，仮に同じ名前の属性があったとしても，本当に同じことを表す値がどうかもわからない．そこで，本研究では，属性値だけでなくデータ構造を比較することで特異なデータを発見する．

値の比較は，比較対象の同属情報に共通する属性の値について，数値データの大小や文字列の一致，不一致などを調べることである．例えば，複数のプリンタという対象についての同属情報とは，製品仕様のデータである．製品仕様のデータの属性のうち，印字速度という属性はすべての製品仕様のデータに存在する．

従って，印字速度という共通の属性の値で比較することでプリンタ同士の差異を発見できる．

構造の比較は，データの構造の違いを特徴として捉え，構造を比較して差異を発見することである．例えば，プリンタの製品仕様という同属情報を考える．プリンタの製品仕様のデータをツリー構造のデータにした場合，あるプリンタに独自の機能は特異な部分木として現れる．この特異な部分木を持つという特徴を差異として捉えることが出来る．

4. 半構造データのスキーマ生成機構

値の比較を行うためには，収集したデータすべての属性が同じ意味を持ち，同じ名前やパスによって指定できなければいけない．また，構造の比較を行うためには，表現が異なっても同じ意味を持つ構造はそのことがわかるようになっているほうが都合が良い．

そこで，本研究では収集したデータに共通するスキーマを生成する手法を提案する．共通スキーマを用いることで，同じ意味を持つ属性でもデータごとに表現方法が異なっているものを同じパスで指定できるようにする．

以下に，共通のスキーマを生成するための手法を述べる．

4.1 OEM への変形

収集してきた同属情報は半構造データであるため，共通の表現方法で記述されているかどうかはわからない．そこで，それぞれのデータを情報交換モデルのひとつである OEM(Object Exchange Model)[9] に変形する．

OEM は識別子と値を持つオブジェクトで構成されているデータモデルである．OEM のオブジェクトが持つ値には，atomic な値と，complex な値がある．atomic な値は，整数，実数，文字列，画像，プログラムなどそれ以上に分割できない最小の値である．atomic な値を本論文では属性値と呼ぶことにする．complex な値とは，0 個以上の子オブジェクトのことである．親オブジェクトと子オブジェクトには，ラベルつきの枝でリンクが張られる．また，OEM オブジェクトは複数の親オブジェクトや循環するリンクを持つことを許している．atomic な値，complex な値を持つオブジェクトをそれぞれ atomic オブジェクト，complex オブジェクトと呼ぶ．

同属情報を OEM に変形するときに，OEM の atomic オブジェクトの値が文章になった場合は，形態素解析などの自然言語処理を行って，十分に小さく，かつ属性の値としてデータの特徴を示すことが出来るだけのプリミティブな値に分割しておく．また，属性値が数値の場合，単位を持つ値ならば，その単位を属性値のデータ型として定義しておく．

表 1　仕様書 1
Table 1　specifications1

項目	項目の情報
フロッピー	3.5 型（1.44MB ／ 720KB）× 1
内蔵HD	80GB Ultra ATA（66）
CD−RW	読み出し 32 倍速，書き込み 8 倍速

表 2　仕様書 2
Table 2　specilications2

項目		項目の情報
外部記憶	FDD	3.5 インチ（1.44MB ／ 720KB）× 1
	HDD	60GB
	CD−RW	読み出し 32 倍速，書き込み 12 倍速
HDDコントローラ		Ultra ATA（66）

表 1，表 2 はパーソナルコンピュータの製品仕様の一部で，外部記憶装置部分のデータを示している．これらのデータはメーカーのホームページから得られた．パーソナルコンピュータの製品仕様は，パーソナルコンピュータの特徴を示す属性の集合で，メーカーごとに記述方法が異なっている同属情報である．これを上で述べたように OEM にしたものが，図 1 である．この図では識別子は省略してある．

4.2 OEM の問題点

取得した同属情報を OEM に変形し，比較しようとすると，次の二点の問題がある．

- データごとのラベルに使われている単語が異なっており，それには同義語や類義語，あるいはデータ独自の言葉が使われている．そのため，値の比較を行う共通の属性を単純にパスのラベルからは発見できない．例えば，図 1 では，記憶装置の名前が “内蔵 HD” と “HDD” と異なっていることや，フロッピーディスクの大きさの単位が “型”，“インチ” と異なっている．
- データごとの表現や値の分類方法の違いにより，機械的に OEM に変形した場合は同じ内容でもパスが異なっていたり，あるいは逆にパスが同じでも別のオブジェクトを示している可能性がある．そのため，OEM の構造的な差異を調べることが困難である．例えば，図 1 では，各種記憶装置の特徴を示す部分木の前に” 外部記憶” というエッジがあることや，HDD コントローラについての属性値が明らかに別の場所にある．

以上の問題点より，半構造データを単純に OEM に変形しただけでは，あるデータの属性のひとつの値について比較を行う前に，他のすべてのデータの属性の値を調査して，比較可能な値であるかどうか調べなけ

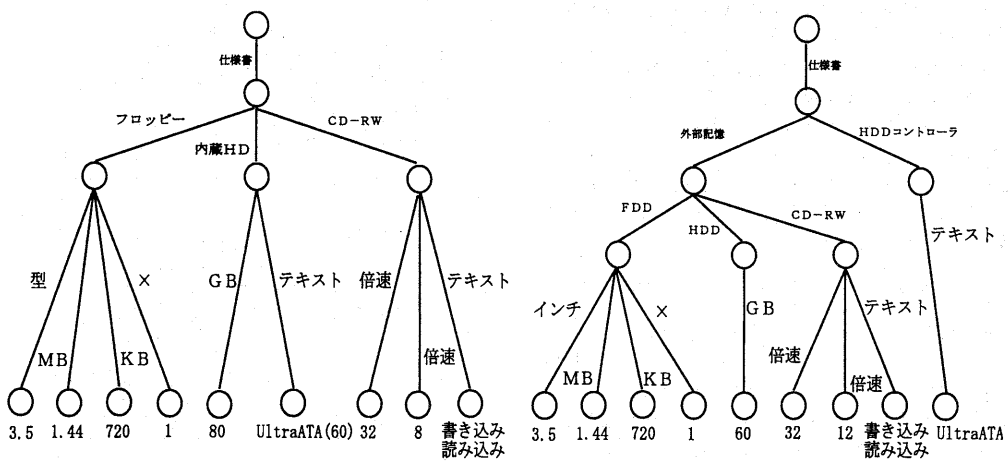


図 1　表 1, 表 2 から得られる OEM
Fig. 1　OEM expression for spexifications 1 and 2

ればならない．そこで，収集したすべての同属情報の OEM について，共通のスキーマを生成する．そして，共通のスキーマに基づいてデータの値を表に格納する．表に格納されている値は属性が明確になっているので，比較が可能である．

DataGuide

OEM を要約し，パスのラベルに依存しないで同じ属性の値を発見できるようにする手法として，Goldman らの strong DataGuide[10] がある．複数のデータからなる OEM を要約することは，データ間の属性値の比較を可能にする．strong DataGuide のスキーマに従ってそれぞれのデータの同じ属性値はハッシュ表に格納され，属性が明確になるからである．

strong DataGuide はターゲットセットに基づいて OEM を要約する．ターゲットセットとは，ある OEM オブジェクトからあるひとつのパスによって到達可能な全てのオブジェクトの識別子の集合である．strong DataGuide のオブジェクトは，ターゲットセットで示される全てのオブジェクトに対応している．

strong DataGuide は以下の手順で生成される．

(1) OEM のオブジェクトからグラフ探索をして，あるパス l のターゲットセット T(l) を得る．

(2) T(l) が今までに発見されたことのないターゲットセットならば，対応する新しい DataGuide のオブジェクトを生成し，DataGuide に l というパスでリンクを張る．

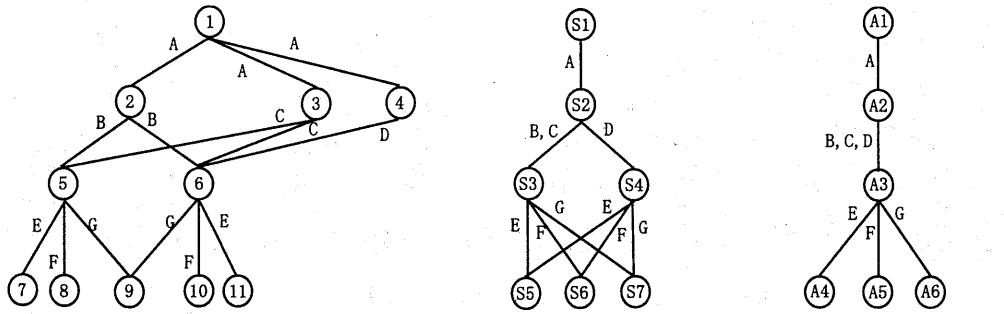
(3) T(l) が既存のターゲットセットならば，対応する既存の DataGuide オブジェクトに l というパスでリンクを張る．

図 **2**(b) は (a) の strong DataGuide である．オブジェクト S2 は，(a) のパス A でリンクされているターゲットセット 2, 3, 4 に対応する DataGUide オブジェクトである．また，パス B とパス C は同じターゲットセット 5,6 を持つので，同じパスとみなされて，(b) ではターゲットセット 5,6 に対応する DataGuide オブジェクト S3 にひとつのパスでリンクが張られている．

strong DataGuide 中のターゲットセットが一致するパスは元の OEM 中のターゲットセットを共有するように定義してあるため，strong DataGuide は OEM のパスのラベルに依存しないで同じ属性の値を発見することが出来る．従って，データごとにパスに使われている単語が異なっているという OEM の問題点が解決できる．

近似的 DataGuide

strong DataGuide は，ターゲットセットがすべて一致しなければ，異なったパスが同一のものであるとみなせない．図??(a) のオブジェクト 4 からのパス D は，ターゲットセットが 6 なので，B,C と D は別のパスとして扱われる．しかし，例えば，オブジェクト 5 以下のオブジェクトが 5 インチのフロッピーディスク，オブジェクト 6 以下のオブジェクトが 3.5 インチのフロッピーを表現しているような場合，これら二つの部分木は本質的には同じなので，データのスキーマとしては，B,C,D が同じパスであることが望ましい．このような問題を解決するための手法として，近似的 DataGuide(Approximate DataGuide)[11] が



(a) OEM　(b) storong DataGuide　(c) Approximate DataGuide

図 2　ある OEM とその strong DataGuide と近似的 DataGuide
Fig. 2　OEM, strong DataGuide and Approximate DataGuide

ある.

近似的 DataGuide を用いることによって，図 2(a) の OEM のパス B, C, D が類似なとき，これらのパスを一つにまとめて，図 2(c) のように要約することが出来る.

パスの類似は，例えば次のように定義できる. strong DataGuide 生成時，ある二つのパス X, Y のターゲットセット T(X) と T(Y) を比較するときに，

$$|T(X) \cap T(Y)| / max(|T(X)|, |T(Y)|)$$

が一定の閾値を超えているときに，これらのターゲットセットを与えるパスは類似である. ただし，これはもっとも単純な定義であり，扱う同属情報の種類によって，構造の類似の定義を考慮する必要がある.

4.3　ボトムアップ型 DataGuide

strong DataGuide や近似的 DataGuide はトップダウンなアプローチで OEM を要約しているが，本研究は，atomic オブジェクトが親オブジェクトの特徴を示すと考えて，ボトムアップなアプローチで OEM を要約する.

任意の atomic オブジェクトと一つの complex オブジェクトからなる部分木を，属性木と呼ぶことにする. 図 3(a) では属性木を破線で囲んで示している. 同一な属性木をひとつのオブジェクトとしてまとめていくことで，OEM を要約する. 同一な属性木の識別は，属性木が持つ atomic オブジェクトの数と属性値のデータの型によって行う. 以下にその手順を述べる.

(1)　OEM を深さ優先探索することにより，属性木を発見する.

(2)　データ型が同じ atomic オブジェクトの数が一致する属性木を探し出す.

(3)　一致する属性木の complex オブジェクトを DataGuide オブジェクトに置き換える. DataGuide オブジェクトの識別子と，置き換えられたオブジェクトの識別子をハッシュ表に格納する. (図 3(b))

(4)　OEM を深さ優先探索して，DataGuide オブジェクトにリンクしているオブジェクトを探し出す.

(5)　同じ DataGuide オブジェクトにリンクしている complex オブジェクトを DataGuide オブジェクトに置き換える. DataGuide オブジェクトの識別子と，置き換えられたオブジェクトの識別子をハッシュ表に格納する. (図 3(c))

(6)　DataGuide オブジェクトにリンクしているオブジェクトがルートでなければ，(4) にもどる.

トップダウンなアプローチでは，図 3(a) のようなルートに近いパスや構造が異なっている OEM を要約することが出来ない. しかし，ボトムアップなアプローチならば，OEM のオブジェクト間の part-of 関係を利用してこのような OEM でも要約することが出来る.

本論文で提案する手法では，upper schema と lower schema を生成する. upper schema とは，全てのインスタンスに共通する属性のみで構成されるスキーマのことである. ただし，upper schema の属性は，空値を許さない. また，lower schema とは，全てのインスタンスが持つ全ての属性で構成されるスキーマのことである. lower schema をインスタンスに適用したとき，インスタンスにない属性は空値で表現される.

upper schema は，全インスタンスに共通な属性のみを持つので，属性値の比較が可能である. そこで，upper schema は値の差異を調べるのに使う. また，lower schema は属性値が空値であるかどうかでインスタンスがその属性を持つかどうかがわかるので，構造の差異を調べるのに使う. 本研究では，差異発見ができるこれら二つのスキーマを合わせて，共通のス

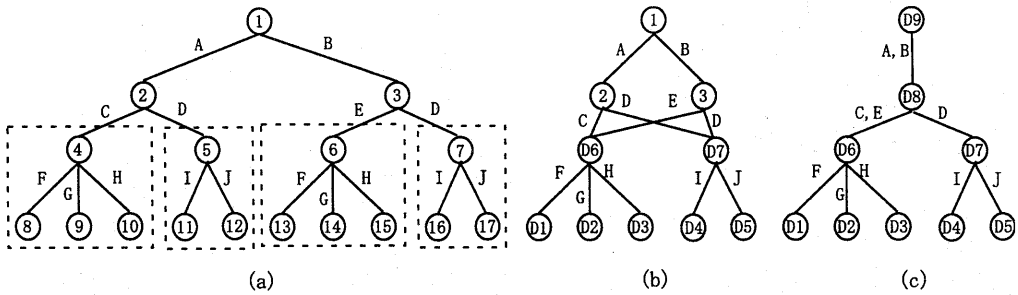


図 3 ボトムアップ型 DataGuide
Fig. 3 bottom up DataGuide

キーマと呼ぶ.

全ての属性木に DataGuide オブジェクトを対応させて OEM を要約すると，lower schema が生成できる．また，lower schema から upper schema を生成することが出来る．そのためには，まず lower schema の属性から，属性値の数がインスタンスの個数と一致するものを全て選ぶ．そして，選んだ属性のみでスキーマを構成すると，upper schema になる．つまり，lower schema を生成するアルゴリズムがあれば，どちらのスキーマも生成することが出来る．以下に，lower schema 生成のアルゴリズムを示す．

ここで

- *root* を OEM のルートオブジェクト
- *child*(o) をオブジェクト o の一つの子オブジェクト
- *children*(o) をオブジェクト o の子オブジェクトの集合
- $Dg(O)$ を complex オブジェクトの集合 O に対応する DataGuide オブジェクトを返す関数
- v を任意の complex オブジェクト
- $V = \{v_1, v_2, \ldots, v_k\}$ を *children*(v) が atomic である v の集合

とする．アルゴリズムの入力は OEM で，出力は入力の OEM を要約した DataGuide のルートである．

(1) $V \neq \phi$ ならば，V について以下を行う．
$V_i = \{v_i \mid children(v_i) = children(v_j), (\forall\, v_i, v_j \in V)\}$
$Dg(V_i) = n_i,\ N = N \cup n_i,\ V = V - V_i$
として (1) を行う．

(2) $child(u_k) = n(n \in N)$ であるような u_k の集合 U について，
(a) 全ての u_k が *root* ならば，
$Dg(root)$ として終了する．
(b) 少なくとも一つの *root* でない u_k が存在するならば，
$V \cup \text{children}(u_k)(u_k \neq root)$
として (1) を行う．

5. 半構造データの比較エンジン

共通のスキーマを用いて半構造データの比較をし，データ間の差異を発見する．以下に，データ間の差異を OEM のオブジェクトを色付けすることによって表現する手法を述べる．

この手法では，データの差異の表現のために，データのインスタンスまたは共通のスキーマの OEM を表示する．表示された OEM のオブジェクトには色がつけられており，色の違いと濃淡によって，値の差異と構造の差異が表現されている．また，ツリーが大きくなる場合は，一定の深さより深いところにある部分木は表示されない．これらの部分木は，ユーザは任意に展開してデータの詳細を得ることが出来るようになっている．

5.1 インスタンスの表示

データのインスタンスが表示されるのは，以下の二つの場合である．ひとつは，検索機能によって条件に適合するデータを発見した場合に個々のデータをユーザに提示する場合である．もうひとつは，ユーザが任意に選んだデータを表示する場合である．データのインスタンスを表示するときには，以下の基準でオブジェクトに色付けがされる．

- complex オブジェクトで，upper schema に含まれるオブジェクトは全て同じ色がつけられる．
- complex オブジェクトで upper schema に含まれないオブジェクトは upper schema とは別の色が付けられる．特異な枝が含まれている部分木が展開されていない場合は，部分木がリンクしているオブジェクトに色づけされる．
- atomic オブジェクトで，数値データなどの大小が比較できる値を持っている場合，平均値と値の差に応じて色の濃さが変わる．色の濃さは彩度を変化させることで表現する．彩度は値の偏差値に応じて変化させる．例えば，偏差値から 50 を引

いた値の二倍を 50 に足した値を彩度とする．インスタンスの表示例を図 4 に示す．

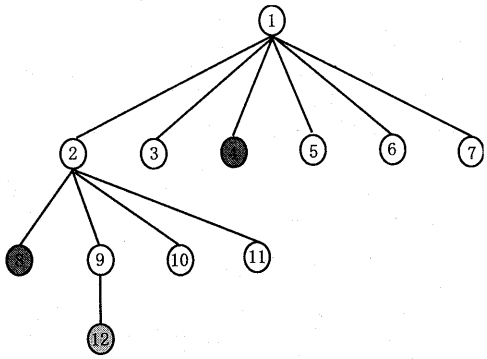


図 4　インスタンス表示例
Fig. 4　example of instance display 1

この図では，オブジェクト 1，2，3，5，6，7，9，10，11 が全てのインスタンスに共通のオブジェクトである．また，オブジェクト 4，8 を展開したときに，特異な部分木が現れる．オブジェクト 12 は atomic オブジェクトで，色が濃くなっているので，平均値よりも高い数値を持っていることを示している．

5.2　スキーマの表示

データのスキーマを表示するのは，三通りの場合がある．ひとつはデータ全体の傾向を知るために，全てのインスタンスから生成された lower schema を表示する場合である．二つ目は検索機能によって条件を満たす複数のデータを発見し，それらのデータから生成された lower schema を表示する場合である．三つ目は，ユーザが任意に選んだデータ間で共通のスキーマを生成して lower schema を表示する場合である．スキーマを表示するときの色付けの基準は以下のものである．

- complex オブジェクトと atomic オブジェクトには別の色がつけられる．
- lower schema 中の complex オブジェクトは，インスタンス全体の中で対応するオブジェクトを持っているインスタンスの割合によって色の濃さが変わる．割合が高い，類似な構造部分は色を淡くする．割合の低い，特異性の高い部分は色を濃くする．
- lower schema 中の atomic オブジェクトは，データの値の分散に応じて色の濃さが変わる．分散が大きければ色を濃くし，小さければ色を淡くする．

あるスキーマの表示例を図 5 に示す．

オブジェクト 1，3，5，6，7 は無着色なので，全てのインスタンスがこれらのオブジェクトを持つことがわかる．また，オブジェクト 2, 9，10，11 は淡い色がついているので，これらのオブジェクトを持つインスタンスが多数存在することを示している．オブジェクト 4，8 は色がかなり濃くなっているので，これらのオブジェクトを持つインスタンスが非常に少ないことがわかる．atomic なオブジェクト 12 は，色が濃くなっているので，属性値の分散が大きいことがわかる．

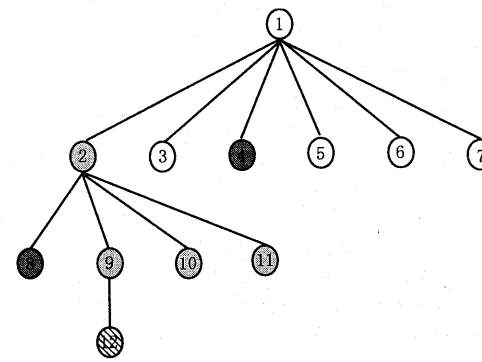


図 5　スキーマ表示例
Fig. 5　example of schema display

6.　製品情報への応用例

前節までに提案した手法の具体的な応用例として，パーソナルコンピュータの製品仕様を収集し，各商品の仕様データの比較，差異発見により，ユーザが購入する商品を選択する支援を行うシステムを考える．

パーソナルコンピュータなどの製品仕様は，表または箇条書きによって記述されているため，データ元が異なってもある程度のデータ構造の類似性がある半構造データ，つまり同属情報である．製品情報を比較するような Web サイトは複数存在するが，これらは既存のデータベースシステムに手動でデータを入力してサービスを実現している．従って，ユーザにとっての負担は軽減できていてもデータベース作成者の負担は大きい．このシステムは，データベースの作成にかかる負担を軽減する．

製品仕様の収集

各メーカーのサイトの商品仕様が記載されているページから，商品仕様のデータ部分を判別し，取得する．以下にその手順を述べる．

- html データをページ中の <table> タグから対応する </table> タグ又は <ul> タグから対応する までの文字列に分解する．
- 分離した文字列の中に使われているタグ以外の文字列とパーソナルコンピュータの製品仕様に頻繁に使われている単語でマッチングを行い，一致している数か割合が一定の閾値を

超えている表または箇条書きを製品仕様とみなす.

製品仕様データの OEM 化

取得した製品仕様のデータを，OEM で表す．このとき，データ中に含まれる文章を形態素解析する．そして，数字と助数詞の組を探し出し，数字を OEM の atomic オブジェクトの属性値とする．助数詞は単位とみなして値のデータ型として定義し，同時に atomic オブジェクトへの枝のラベルにも使う．また，単純に文章だけの値も atomic な文字列型の値を持つオブジェクトとなり，”テキスト”というラベルの枝でリンクが張られる．

共通のスキーマの生成

収集した半構造データから得られた OEM から，共通のスキーマを生成する．パーソナルコンピュータのメモリ，ハードディスク，フロッピーディスクなどの属性は，Byte という単位で属性値が与えられる．本論文で提案する手法は，属性値の定義域も考慮するので，それぞれ数十〜数百メガ，数十ギガ，数百〜一メガの範囲で値をとるこれらの属性を別々に判別して共通のスキーマを生成することが出来る．

製品仕様の比較と差異発見

製品仕様を値と構造について比較し，データ間の差異を発見する．

構造についての比較とは，この場合，比較するデータがそれぞれどんな機能を持っているか，もっていないかを調べることになる．値についての比較とは，この場合，比較するデータすべてに共通する機能の性能の比較ということになる．

本システムはデータの比較により製品ごとの仕様のデータの特徴をユーザに提示することで購入する商品の選択を支援するが，更にユーザ履歴などからユーザーの興味を把握する[12] ことで，より有用なシステムとなるだろう．

7. おわりに

本論文では，現行の Web システムでは，ユーザが Web から情報を得るためには，ユーザが自身が手動で複数のページを閲覧して条件に適合するデータを比較しなければならないという問題を解決するために，半構造データを自動的に収集する方法と比較する方法を提案した．更に，半構造データを比較するための共通のスキーマの生成方法として，OEM をボトムアップに調べる手法を提案した．又，データ比較によって発見された差異を視覚化してユーザに提案する手法を提案した．

そして，提案した手法の具体的な応用例として，パーソナルコンピュータの製品仕様を比較することで，ユーザが購入する商品を検討する支援を行うシステムを提案した．また，本手法は他にも，人の履歴や観光地の情報を比較するのにも使うことが出来る．

謝辞　本研究の一部は，文部省科学研究費基盤 (C)「分散型ハイパーメディアからの構造発見とアクセス管理」（課題番号 12680416）及び，日本学術振興会未来開拓学術研究推進事業における研究プロジェクト「マルチメディア／コンテンツの工事処理の研究」（プロジェクト番号 JSPS-RFTF97P00501）によっております．ここに記して謝意を表すものとします．

参考文献

1) P.Buneman, S.Davidson, M. Fernandez, D. Suciu: “Adding Structure to Unstructured Data”, Sixth International Conference Proceedings, pp. 336-350, Delphi, Greece, January, 1997
2) http://www.libra.ne.jp/
3) http://www.dealtime.co.jp/
4) http://impress.earena.co.jp/
5) http://www.kakaku.com/
6) 富田一郎，手塚祐一，山本修一郎，長岡満夫: HTML 文書からの商品情報抽出方式の提案，情報処理学会第 56 回全国大会講演論文集 (3)，pp.79-80，1998 年 3 月
7) Robert D. Doorenbos, Oren Etzioni, Daniel S. Weld: “A Scalable Comparison-Shopping Agent for the World-Wide Web”, Proceeding of the First International Conference on Autonomous Agents,1997
8) excite.com: http://www.jango.excite.com/
9) S. Abiteboul, D. Quass, J. McHuge, J. Wiener: “The Lorel Query Language for Semistructured Data”, Jornal of Digital Libraries,1(1), November, 1996
10) Roy Goldman, Jennifer Widom: “DataGuides: Enabling Query Formulation and Optimization in Semistructured Databases”, Proceedings lf the Twenty-Third International Conference on Very Large DataBases, pp. 436-445, Athens, Greece, August, 1997
11) Roy Goldman, Jennifer Widom: “Approximate DataGuides”, Technical report, Stanford University, 1998
12) 高野正樹，鈴木優，波多野賢治，吉川正俊，植村俊亮：”XML 文書における要素名と文書構造を利用した情報フィルタリング”，信学技報，pp.9-15，2001