仮想電子辞書システムの設計と構築

Design and Construction of Virtual Electronic Dictionaries

芳野 学†, 都司 達夫‡, 宝珍 輝尚‡
Yoshino MANABU†, Tsuji TATSUO‡, Houchin TERUHISA‡
†福井大学 工学研究科 ‡福井大学工学部
Fukui University, 3-9-1 Bunkyo, Fukui-Shi 910, Japan

キーワード: 電子辞書, データベース, ビュー
Keywords: electronic dictionary, database, view

あらまし: 母体とする複数の辞書をその論理構造にしたがって構成要素に分解し，項目結合可能性の制限の下にユーザが必要な項目のみを自由に組み合わせて新たに定義できる「仮想電子辞書」の提案とその設計・実装について述べる．これらの仮想電子辞書の基本概念の説明とともに，定義構文，検索構文を提案し，その実装方式を説明する．

Summary: In this paper, we describe design and implementation of virtual electronic dictionary system. In our system, one or more dictionaries are decomposed into their components according to their logical structures and some of them are corrected and combined logically into a virtual dictionary under the connectivity condition. Along with the description of the fundamental ideas underlying our virtual dictionaries, defining and retrieving primitives of virtual dictionaries are introduced and their implementation is explained.

1 はじめに

記憶メディアの大容量化と低廉化およびマルチメディア技術の進展にともなって，各種電子辞書の出版は盛んに行われている．特に，画像や写真，音声を取り込んだ「マルチメディア辞書」の発展は著しい．本来，辞書にはきわめて豊富で多角的な情報が内包されており [1]，これらの情報は「辞書の文法」にしたがって，構造的に記述されている．手で引く場合にはこの構造の強制に縛られており，辞書が潜在的に保有している豊富で多角的な情報を手軽に利用することは不可能である．電子辞書の存在意義はこのような構造の強制を解放し，ユーザの望む情報をユーザの望む配列で提示することができることであると考えられる．これには，望む情報を特定するために，辞書の文法にしたがった構造化を行う必要がある．電子辞書を含めた電子化出版物の構造化には，たとえば EPWING[8] などの規格が存在しており，それにしたがった，電子辞書も出回り始めている．また，[2] では辞書のハイパーテキスト化の方式が提案されている．ところが，辞書の論理構造（スキーマ）を前面に押し出し，ユーザレベルで既存辞書からニーズにかなった，仮想的な辞書を定義し利用するための枠組みはあまり，見当たらない．

そこで，本報告では，上記のような目的に沿った「仮想電子辞書」の設計と構築について述べる．仮想電子辞書として，一種類の辞書内で有用な情報のみを再配列して得られる「部分辞書」と複数の辞書内の有用な情報を「結合対」を介して結合して得られる「結合辞書」を提案する．さらに，これらの仮想辞書に関連する諸概念を提案するとともに，仮想電子辞書の定義・操作言語を提案し，その実装方式を説明する．

2 仮想辞書の設計

市販電子辞書における相互検索機能を使えば，複数辞書にまたがった場合の検索操作性は向上するものの，個々の辞書を意識して，切り替えるといった操作が必要であり，煩わしい．ここでは，個々の辞書の実体とは別個にユーザが定

義する仮想辞書を設計する．仮想辞書は説明項目の実体は持たずに，辞書実体のどの部分を使用するかの定義情報のみ有する．ここでの仮想辞書の考え方は，データベースシステムにおけるビュー機能に相当している．各実体辞書の文法構造を表現する辞書スキーマ群に対するビュースキーマとして仮想辞書のスキーマをユーザが定義できる．仮想辞書に対する検索要求は辞書実体に対する３で述べる検索ライブラリ中の検索関数呼び出しにより，実行される．以降では特に「仮想辞書」と断らない限り，単に辞書といった場合は実体辞書を指すこととする．なお，本研究で用いた辞書の元データは三省堂（株）から提供を受けた，新クラウン和英辞典 [5]，ニューセンチュリー英和辞典 [6] であり，Webster の英語シソーラス [7] を参考にした．

2.1 結合対

「仮想辞書の自動生成」を行う場合には，項目同士の結合操作がポイントとなる．例えば，英和辞書から，「反意語仮想辞書」を作成する場合，英和辞書の見出し説明中に現れる「反意語」が結合対となり，（同じ）英和辞書の見出しと結合してその反意語の詳しい情報を与える仮想辞書が定義される．なお，辞書の見出しは必ず，結合対となり得る．ここでは，種々の辞書の辞書構造と各項目のデータの分類を掲げて，仮想辞書生成時の結合対について考察する．

(1) 和英辞書

和英辞書の論理構造を表１に示す．[ ] 付きの項目は辞書の部分構造を決定するための項目であり抽象項目と呼ぶこととする．また，[ ] なしの項目は辞書に記載されているデータであり，具体項目と呼ぶ．

それぞれの具体項目のデータは，結合可能性の観点から表２のように分類される．「見出し」以外の項目には記号 (1byte) も含まれている．ここでは，「英訳の例文」「用例の例文」「関連語の例文」をまとめて「例文」としている．また，用例や例文は（和単語列，英単語列）の対として与えられるが，ここでは例えば，用例の場合，和単語列成分は「用例和」，英単語列成分は「用例英」として別個の項目としている．

なお，以降の結合可能性分類表における右端の欄の”－”, ”○”, ”△”, ”×” は，”結合は無意味”, ”結合可能”, ”データに処理を施すことで結合可能”, ”結合は困難” をそれぞれ表している．見出しは例えば国語辞書など，仮名和単語や片仮名和単語を見出しとする辞書とそのまま結合可能である．”△” のついたものは，適当な構文解析や整形によって，有意な単語に分解することにより，他の結合対と結合可能である．「漢字訳」は，見出しとほとんど１対１の関係にある．しかし，たとえば「いたみ」のように「痛み」と「傷み」のような同音異義語があるが，和単語を結合対とする他の辞書と結合可能である．「英訳」は多くの場合単一英単語であり，英単語を見出しとする他の辞書と結合可能である．英単語列の場合には，英語辞書の成句と結合したり，名詞＋名詞や形容詞＋名詞のような複合語の場合には最後の名詞が有意であり，結合対となり得る．「関連語和」の場合には，見出しを含む複合語である場合が多く，たとえば，国語辞書の参考語と結合可能であり，より詳しく，その日本語の意味を知ることができる．また，用例や例文はいずれも，単語列として他の説明項目と全く一致することはほとんどない．ただし，構文解析によりその中の名詞や動詞について他と結合することは可能である．

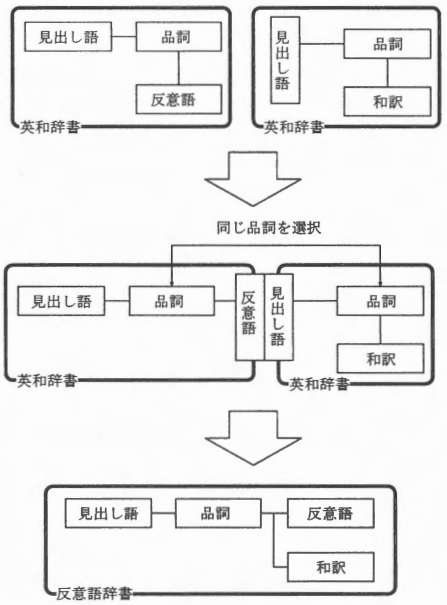

図 1: 結合対と結合条件

表１　和英辞書の構造

[和英辞書] → 見出し [見出し項目]
[見出し項目] → 漢字訳 [英訳項目]
[英訳項目] → 英訳 [対訳項目]
[対訳項目] → [例文] [用例] [関連語]
[例文] → 例文和 例文英
[用例] → 用例和 用例英 [例文]
[関連語] → 関連語和 関連語英 [例文]

表２ 和英辞書の結合対分類表

分類	分類レベル	文字種	結合可能性
見出し	辞書	和単語	○
漢字訳	見出し項目	和単語	○
英訳	英訳項目	英単語 (列)	△
用例和	英訳	和単語 (列)	×
用例英	英訳	英単語 (列)	×
関連語和	漢字訳	和単語	△
関連語英	漢字訳	英単語 (列)	△
例文和	英訳, 用例, 関連語	和単語 (列)	×
例文英	英訳, 用例, 関連語	英単語 (列)	×

(2) 英和辞書

英和辞書の論理構造を表３に示し，結合対分類表を表４に示した．発音，重要度，UC フラグ，語形変化については，他の辞書項目と結合して意味のある仮想辞書の定義は考えにくいと判断してここでは，結合可能性を「－」とした．品詞種別については，国語辞書などと組合わせて品詞別の仮想辞書を定義することができるので，「○」とした.「類義語」や「反意語」は説明の形式に統一性はないが，そのなかの英単語を取り出すことにより，英語見出しや他の英語辞書の類義語と結合し得る．さらに,「解説」や「文法情報」については説明が不定型であり，有意な結合は行えないとして,「×」とした.

表３　英和辞書の構造

[英和辞書] → 見出し [見出し項目]
[見出し項目] → 重要度 発音 [品詞項目]
解説 文法情報
[品詞項目] → 品詞 語形変化 UC [語義項目]
他の品詞形 反意語 類義語 解説
文法情報 [成句項目]
[語義項目] → UC 語義 [用例] 解説
文法情報 [下位語義]
[用例] → 用例英 用例和
[下位語義] → 語義 [用例] 解説 文法情報
[成句項目] → 重要度 成句 意味 [用例]

表４ 英和辞書の結合対分類表

項目	分類レベル	文字種	結合可能性
見出し	辞書	英単語	○
重要度	見出し項目 成句項目	数値	－
発音	見出し項目	発音記号	－
品詞	品詞項目	品詞識別子	○
語形変化	品詞項目	英単語列	－
UC フラグ	品詞項目 語義項目	UC フラグ	－
語義	語義項目, 下位語義	和単語列	△
用例英	用例	英単語列	×
用例和	用例	和単語列	×
反意語	品詞項目	英単語列	△
類義語	品詞項目	英単語列	△
他の品詞	品詞項目	英単語	△
解説	全レベル	不定	×
文法情報	成句レベル以外	不定	×
成句	成句項目	英単語列	○
意味	成句項目	和単語列	×

(3) 英語シソーラス辞書

英語シソーラス辞書の構造を表５に示す.

表５　英語シソーラス辞書の構造

[英語シソーラス辞書] → 見出し [見出し項目]
[見出し項目] → 品詞 [品詞項目]
[品詞項目] → 語義 用例 [語義項目]
[語義項目] → [類義語項目] [関連語項目]
[成句項目] [反対語項目] [反意語項目]
[類義語項目] → 類義語
[関連語項目] → 関連語
[成句項目] → 成句
[反対語項目] → 反対語
[反意語項目] → 反意語

2.2　仮想辞書定義

2.1 で述べた結合対にしたがって，複数の辞書を結合した場合には，複数の辞書が単に結合

対を介して合わさっただけであり，実用にはならない．仮想辞書を実用的に使用するためには，ユーザ自身が仮想辞書を自分のニーズにしたがって，きめ細かく定義できる必要がある．このために，ここでは，本辞書システムにおける，仮想辞書定義の定義要素を以下に述べる．以降の説明においては例として，英和辞書と英語のみの英語シソーラス辞書を使って，和訳付き英語シソーラスの仮想辞書を作成する場合を取り上げる．

(1) 辞書構造の定義

複数の辞書を結合しても，それらの辞書の対応する項目がすべて必要であるわけではない．各辞書の項目を選択的に取り込み，それを可能な配置により再編成することにより，仮想辞書の構造を定義できる．和訳付き英語シソーラス辞書の場合，例えば，

```
和訳付き英語シソーラス辞書（見出し，
    見出し項目（品詞，品詞項目（
        （類義語，類義語項目（語義）），
        （反意語，反意語項目（語義））））
```

のようにその構造を入れ子で指定する．

(2) 母体となる辞書群の指定

結合の対象となる実体辞書または仮想辞書を１つ以上指定する．１つの場合には，結果として得られる辞書を仮想部分辞書といい，２つ以上の時は仮想結合辞書という．和訳付き英語シソーラスの仮想辞書の場合には，母体辞書群は英和辞書，英語シソーラス辞書である．

(3) 母体辞書との対応付けの定義

(1) で定義した仮想辞書の具体項目が実体辞書のどの項目に関連づけられるかを定義する．実体辞書の論理構造（スキーマ）にしたがって，見出しから始まって辞書項目に至るパスを実体辞書の項目として指定する．たとえば，図２の英和辞書スキーマにおいて，”見出し項目. 品詞項目. 成句項目. 成句”とすれば，以下の (5) の選択条件を満足するすべての成句を指示することになる．また，”見出し項目. 品詞項目. 成句項目”とすれば，選択条件を満足するすべての成句について，その「重要度」,「成句（の綴り）」,「意味」,「用例」を指示する．和訳付き英語シソーラス辞書の場合，例えば，

```
見出し = ET. 見出し，
見出し項目. 品詞 = ET. 見出し項目. 品詞，
見出し項目. 品詞項目. 類義語 =
    ET. 見出し項目. 品詞項目. 語義項目. 類義語，
見出し項目. 品詞項目. 類義語項目. 語義 =
    EJ. 見出し項目. 品詞項目. 語義，
見出し項目. 品詞項目. 反意語 =
    ET. 見出し項目. 品詞項目. 語義項目. 反意語，
見出し項目. 品詞項目. 反意語項目. 語義 =
    EJ. 見出し項目. 品詞項目. 語義
```

となる．ここで，ET は英語シソーラス辞書，EJ は英和辞書を表す．

(4) 結合対の指定

２つの母体辞書を D_1, D_2 とする．D_2 の項目 $e_1 \ldots e_n$ が結合可能であり，D_1 の見出しと結合する時，”D_2 は項目 $e_1 \ldots e_n$ を介して D_1 と結合可能である”といい，$\langle D_1, D_2.e_1 \ldots e_n \rangle$ と表記してこれを結合対という．和訳付き英語シソーラス辞書の場合には，結合対は

⟨EJ, ET. 見出し項目. 品詞項目. 語義項目. 類義語 ⟩
⟨EJ, ET. 見出し項目. 品詞項目. 語義項目. 反意語 ⟩

となる．結合対による２つの辞書の結合は関係データベースにおける結合演算に相当する．

(5) 選択条件の指定

和訳付き英語シソーラスの仮想辞書の場合，基本的には，英和辞書も英語シソーラス辞書も品詞ごとに分けられているが，互いに異なる品詞を結合しても，有意な情報とはいえないし，組み合わせの数も膨大になる．このような場合には，「選択条件」として，”ET. 見出し項目. 品詞項目. 品詞 = EJ. 見出し項目. 品詞項目. 品詞”というようにして結合の条件を指定する．他に，動詞ばかりの仮想辞書を作成する場合には，”ET. 見出し項目. 品詞項目. 品詞＝動詞 and EJ. 見出し項目. 品詞項目. 品詞＝動詞 ”の指定になる．

(6) 見出し指定と見出しフィルタ

見出しの指定を行う．見出しとして複数の辞書項目を指定してもよい．見出しには既定として，高速アクセスのための索引が付与される．通常，結合条件の選択演算により，仮想辞書のレコード集合は選択的に定義されるが，見出し集合に働くフィルタによっても選択することができる．見出しフィルタとして，正規表現が利用できる．また，逆引き辞書 [4] のように，見出しの読みを逆順にして，並べ替えることもできる．

2.3 仮想辞書の定義構文

仮想辞書には，仮想部分辞書と仮想結合辞書がある．いずれの場合にも「辞書構造の定義」，「母体辞書の指定」と「母体辞書との対応付けの定義」は必須である．

(1) 仮想部分辞書

仮想部分辞書では辞書の結合は行われず，単一辞書について，ユーザの必要とする項目のみを再編成する．数ある辞書項目をすべて検索対象とはせずに，使用目的に沿って選んだ辞書項目のみをよりコンパクトにまとめた辞書を単一辞書から抽出することが目的である．たとえば，英和辞書の中から，成句のみを抽出して成句辞書を作成したい場合には，

```
DEFINE  成句辞書 (見出し，見出し項目
                  (品詞，成句))
FROM    EJ
ASSIGN
  見出し = EJ. 見出し，
  品詞 = EJ. 見出し項目 . 品詞，
  成句 = EJ. 見出し項目 . 品詞項目 .
          語義項目 . 成句項目 . 成句
WITH ENTRIES  見出し
```

となる．DEFINE 句では，仮想辞書名の下に辞書項目とその構造が定義される．実体辞書では「品詞項目」の下に 3 つの項目があり，「語義項目」には 5 つの項目があったが，成句のみを選ぶことにより，これら 2 つの抽象項目はなくなり，構造は浅くなっている．FROM 句は母体辞書の指定であり，AS 以降は母体辞書による定義内容である．ASSIGN 句は仮想辞書の各辞書項目が実体辞書のどの項目に関連づけられるかを定義する．WITH ENTRIES 句は仮想辞書の見出しを含む 1 以上の複数の辞書項目を指定する．物理的にはこれらを複合キーとして，索引が生成される．

見出しに対する索引は，実体辞書の索引をそのまま使用できないこともないが，実体辞書のすべての見出しが成句項目を持っているわけではない．したがって，仮想辞書の見出しに対してフィルター検索をかけた場合，無駄な探索を行ってしまうので，仮想辞書用に新たに別途作成する．

(2) グルーピング

次に，英作文を支援する仮想辞書の構成を考える．英文作成において，実際にもっとも役に立つ情報は用例であろう [3]．例えば次のように，和英辞書から例文，用例，関連語を集めて，用例対訳辞書 A を定義する．

```
DEFINE  用例対訳辞書A (見出し,
                  用例項目 (用例和，用例英))
FROM  JE
ASSIGN
  見出し =  JE. 見出し，
  (用例和，用例英) =
(<JE. 見出し項目 . 英訳項目 . 対訳項目 . 例文和，
JE. 見出し項目 . 英訳項目 . 対訳項目 . 用例和，
JE. 見出し項目 . 英訳項目 . 対訳項目 . 例文和，
JE. 見出し項目 . 英訳項目 . 関連語 . 関連語和，
JE. 見出し項目 . 英訳項目 . 関連語 . 例文和>，
<JE. 見出し項目 . 英訳項目 . 対訳項目 . 例文英，
…… (用例和 と同様) >)
WITH  ENTRIES  見出し
```

ASSIGN 句の「用例和」は和英辞書の 5 つの和文の辞書項目を 1 つにまとめた新たな単一項目である．「用例英」も同様の新たな単一項目である．このような操作を辞書定義レベルのグルーピング (GROUPING) という．このときの問題は，「用例和」と「用例英」の対応のさせ方である．一般に 2 つのグルーピング A $\langle a_1, a_2, \cdots, a_n \rangle$ および B $\langle b_1, b_2, \cdots, b_n \langle$ があり，仮想辞書の DEFINE 句において (A, B) と並置され，母体辞書において (a_i, b_i) ($i = 1,2, \cdots, n$) と並んでいるとき，(A, B) の項目間は (a_1, b_1) (a_2, b_2) $\cdots$ (a_n, b_n) と対応づけられる．2 つのグルーピングの項目間がこのように対応づけられることを ASSING 句において，( ) で囲むことによって表す．

ところで，用例は英和辞書にも記載されており，その仮想辞書，“用例対訳辞書B (見出し，用例項目 (用例英，用例和))” が用例対訳辞書Aと同様にして定義できる．

(3) 仮想結合辞書

和訳付き英語シソーラス辞書の定義をまとめると，

```
DEFINE  和訳付き英語シソーラス辞書 (見出し,
          見出し項目 (品詞，品詞項目 (
            (類義語，類義語項目 (語義)) ,
```

(反意語, 反意語項目 (語義)))))
FROM ET, EJ
ASSIGN
見出し = ET.見出し,
見出し項目.品詞 = ET.見出し項目.品詞,
見出し項目.品詞項目.類義語 =
ET.見出し項目.品詞項目.語義項目.類義語,
見出し項目.品詞項目.類義語項目.語義 =
EJ.見出し項目.品詞項目.語義,
見出し項目.品詞項目.反意語 =
ET.見出し項目.品詞項目.語義項目.反意語,
見出し項目.品詞項目.反意語項目.語義 =
EJ.見出し項目.品詞項目.語義
WITH ENTRIES 見出し, 見出し項目.品詞
CONNECTED BY
<EJ, ET.見出し項目.品詞項目.語義項目.類義語>
<EJ, ET.見出し項目.品詞項目.語義項目.反意語>
WHERE
EJ.見出し項目.品詞 = ET.見出し項目.品詞

となる. 結合対の指定には, CONNECTED BY 句を導入した. また, WITH ENTRIES 句により, ET の見出しと品詞の対が指定されており, この対をキーとして索引が生成される. 仮想結合辞書は, 複数の辞書を結合して得られるが, 必ず, FROM 句において複数の辞書と, CONNECTED BY 句において結合対を指定する必要がある.

(4) 見出しなしの仮想辞書

既述の仮想辞書には必ず見出しが定義されていた. ユーザは見出しを指定することにより, 関連する辞書項目を直接検索することができる. ところが, 場合によっては, 見出しにはよらず, 定義されたデータを"縦断的"に検索したい場合がある. たとえば, 2.3 で述べた用例対訳辞書は便利であるが, 調べたい言葉の見出しに関するものだけしか知ることができない. たとえば, ある言葉の用例の和英対訳を辞書全体にわたって検索できれば, 手引きでは調べ尽くせない有用な例文を数多く得ることができる. この仮想辞書は既定義の仮想辞書である用例対訳辞書Aを母体辞書として, 次のように定義できる.

DEFINE 用例対訳辞書和英 (用例和, 用例英)
FROM 用例対訳辞書A
ASSIGN
用例和 = 用例対訳辞書A.用例項目.用例和,
用例英 = 用例対訳辞書A.用例項目.用例英

このとき, 用例対訳辞書Aの定義における用例和と用例英のグルーピングとそれらの対応関係は用例対訳辞書和英においても有効であることに注意されたい.

見出しなしの仮想辞書はいわゆる全文検索機能に対応するが, 検索の対象を仮想辞書で定義された項目のみに限定することが可能であり, 効率的な検索が行える.

(5) 辞書の縦列結合

見出しなしの複数の仮想辞書を縦続接続することができる. 縦列結合する仮想辞書は構造等価でなければならない. 2つの辞書が構造等価であるとは, DEFINE 句で指定される (木) 構造が同一であること, および, 具体項目のデータ型が一致していることことである.

(4) の例では, 和英辞書を母体とする用例対訳辞書Aのみが検索の対象となっているが, 英和辞書を母体とする (2) の用例対訳辞書Bの用例も併せて検索したい. 辞書を縦列結合してあらたな仮想辞書を作り出す構文は以下の通りである.

DEFINE 用例対訳辞書和英 (用例和, 用例英)
FROM 用例対訳辞書A, 用例対訳辞書B
ASSIGN 用例和 =
[用例対訳辞書A.用例項目.用例和 |
用例対訳辞書B.用例項目.用例和]
用例英 =
[用例対訳辞書A.用例項目.用例英 |
用例対訳辞書B.用例項目.用例英]

[… | … | …] は縦列結合を表す. 縦列結合は結合対による結合によらず, したがって, 結合対の指定は不要であること, やはり, 母体辞書におけるグルーピングとそれらの対応関係が保存されることに注意されたい.

(6) 実体辞書定義文

仮想辞書と同様に実体辞書についても実体辞書定義文を同様に定める必要がある. これは, 定義構造を仮想辞書のものと同一にし, 操作上区別なしに扱えるようにするためである. これにより, たとえば, 母体辞書定義として, 仮想辞書と実体辞書を混在させることができる. 実体

辞書定義文は実体辞書の構造を定義するとともに，それぞれの項目に対応する辞書の実データを検索するライブラリ関数のアドレスを関連づけする．項目の検索にあたってはこの関数を呼び出せばよい．

```
DEFINE 英語シソーラス辞書 (見出し語,
        見出し項目 (品詞, 品詞項目 (語義,
        語義項目 (用例, 類義語, 関連語,
        成句, 反対語, 反意語))))
ASSIGN
    見出し語= thesaurus_key(),
    見出し項目. 品詞 = thesaurus_part(),
    見出し項目. 品詞項目. 語義
        = thesaurus_sense(),
    見出し項目. 品詞項目. 語義項目. 用例
        = thesaurus_example(),
    見出し項目. 品詞項目. 語義項目. 類義語
        = thesaurus_synonym(),
    見出し項目. 品詞項目. 語義項目. 関連語
        = thesaurus_relate(),
    見出し項目. 品詞項目. 語義項目. 成句
        = thesaurus_idiom(),
    見出し項目. 品詞項目. 語義項目. 反対語
        = thesaurus_contrast(),
    見出し項目. 品詞項目. 語義項目. 反意語
        = thesaurus_antonym()
```

2.4 仮想辞書の検索

仮想辞書の検索はFIND文により行う．仮想辞書のDEFINE文の構造定義中の項目に具体値を代入して，条件付けする．具体値の指定には，正規表現が使用できるほか，論理和｜も使用できる．例えば，和訳付き英語シソーラス辞書の場合，

```
FIND 和訳付き英語シソーラス辞書 ("*take",
        見出し項目 ("動詞", 品詞項目 (
        (類義語，類義語項目 (語義)),#)))
```

とすれば，"take"で終わる動詞の英単語の (類義語，類義語項目 (語義)) がリストアップされる．#はその項目以下の項目がすべてマスクされ，検索条件にも検索対象にもならないことを示す．以下はこの検索文による検索結果である．

```
見出し語：ｔｅｓｔ
品詞：動詞
 類義語 (１)：ｃｈｅｃｋ
  和訳 (１)：1. 阻止する, 防ぐ, 抑制する
          2. テストする, 検査する, 調べる
          3. 預ける 4. 王手をかける
  和訳 (２)：1. 一致する, 符合する
          2. 検査する
          3. 小切手を振り出す
          4.[キングに] 王手をかける
 類義語 (２)：ｅｘａｍｐｌｅ
  和訳 (１)：1. 調べる, 検査する, 審査する
          2. 試験する
          3. 診察する, 検診する
          4. 尋問する
 類義語 (３)：ｐｒｏｖｅ
  和訳 (１)：1. 説明する, 立証する
          2. 試す, 試験する
  和訳 (２)：1. 判明する, わかる
 類義語 (４)：ｄｅｍｏｎｓｔｒａｔｉｏｎ
  和訳 (１)：1. 実証する, 明らかにする
          2. 宣伝する, 説明する
          3.[感情などを] 顔に出す
  和訳 (２)：1. デモする
 類義後 (５)：ｔｒｙ
  和訳 (１)：1. 試す, 使ってみる
          2. 試みる, やってみる
          3. しようと努める, 努力する
          4. 試練を与える, 無理させる
          5. 審判を行う, 裁判する, 審理する
  和訳 (２)：1. 試みる, やってみる, 努力する
```

また，

```
FIND 和訳付き英語シソーラス辞書 ("*take",
        見出し項目 ("動詞", 品詞項目))
```

の「品詞項目」のように，抽象項目のままにしておくと下位の項目がすべて検索対象になる．

見出しなしの仮想辞書の検索も同様にFIND文により行うことができる．例えば，

```
FIND 用例対訳辞書和英 ("*考え*", 用例英)
```

は用例和に“考え”を含むような用例の和英対訳表を作る．用例対訳辞書和英は２つの仮想辞

書を母体としており，これらは”対応づけられたグルーピング”を含む．したがって，グルーピングされた各項目ごとに対応関係が存在するので，和文に対して英文が一意に定まる．

3 仮想辞書の実体化

仮想辞書は実体辞書に対するビューであり，その検索は次の理由から一般的にオーバヘッドが大きい．

(a) 実体辞書にあった抽象項目が仮想辞書定義において，消失した場合でも，実際の検索は実体辞書を用いるために，抽象項目を介して行う必要がある．

(b) 仮想結合辞書の結合処理コストが非常に高い．検索時に毎回結合処理を行うことは実用的でない．

仮想辞書の定義と実体辞書の辞書項目検索ライブラリルーチンを用いて，実体辞書と同様にその物理データ構造を二次記憶中に作り出す操作を「仮想辞書の実体化」という．これにより，上記のオーバヘッドは回避できる．

なお，2.3(1) で述べたように，仮想辞書の見出しに対する索引だけは，仮想辞書定義時にすでに作成されており，実体化しなくても使用できる．また，(b) に対しては，結合によって連結される相手辞書のレコードの参照情報が結合索引として，仮想結合辞書定義時に作成される．これらの対策により，項目データも含めて完全に実体化しなくても，見出し指定検索の場合には，実用に耐えうるパフォーマンスは得られる．ただし，見出しなしの仮想辞書の場合には，定義項目の全検索が必要であり，それ用に最適化した辞書データの内部データ構造を自動生成することが必要となる．実体化は MATERIALIZE 文で行う．

MATERIALIZE 用例対訳辞書和英

4 仮想辞書システム

4.1 モジュール構成

仮想辞書システムは (1) ユーザからの仮想辞書定義や，仮想辞書検索の指示を受け，それぞれを DEFINE 文、FIND 文に変換する GUI，(2)DEFINE 文，FIND 文を構文解析し，対応する内部データ構造を 1 次記憶中に作成するパーザ，(3) 内部データ構造を参照しながら仮想辞書操作を行うインタープリタ，(4) 母体辞書 DB，母体辞書定義文 DB 仮想辞書定義文 DB，(5)GUI からの要求にしたがって実際に母体辞書に指定された実体辞書や仮想辞書の検索を行うライブラリ，(6) 仮想辞書実体化サブシステム，などからなる．

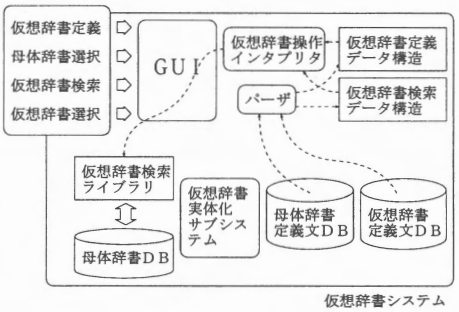

図 2: 仮想辞書システム

(1) 仮想辞書の定義

仮想辞書の定義を行なう場合，ユーザーは仮想辞書定義文を作成する．仮想辞書定義文を受け取った仮想辞書定義文パーザはこれを解析し，1 次記憶中に仮想辞書定義内部データ構造に変換する．

(2) 仮想辞書の検索

(1) と同様にユーザは仮想辞書検索文を作成する．定義文と同様に仮想辞書検索パーザにより，仮想辞書検索内部データ構造に変換され仮想辞書インタプリタに渡される．検索データを受け取ったインタプリタは主メモリ上で被検索辞書の辞書定義データを探して，必要な検索ルーチンを決定する．決定したらインタプリタは必要な検索ルーチンを母体辞書サーバに指示し，母体辞書サーバはその指示にしたがって母体辞書から検索を行なう．検索結果はインタプリタを通して，GUI などから出力される．

4.2 GUI

前節で述べた言語ベースのインターフェースでは項目指定のためのパス表現が多くてかつ長たらしいので，仮想辞書はやはり使いにくい．仮想辞書の定義や操作には GUI ベースの分かりやすいインターフェースを提供することが必要で

ある．特に本システムでは辞書スキーマを前面に押し出す必要がある．すなわち，仮想辞書の定義にあたっては，母体辞書のスキーマをグラフィカルに提示し，定義に必要な項目をその中から切り出し，可能な範囲で望みどうり組み合わせる機能を提供することにより，新辞書のスキーマ構築を支援する必要がある．

本研究では，このようなGUIをCGIを用いて実装することで，Web上で仮想辞書の定義，検索を可能とした．

GUIは大きく分けて，仮想辞書定義部と仮想辞書検索部とに分けられる．

(1) 仮想辞書定義部（付録 (1) 参照）

仮想辞書定義部では，最初に仮想辞書の辞書構造を作成する．

辞書構造の作成は，

1. 仮想辞書中のどこに項目を追加するかを選択．
2. 母体辞書を選択．
3. 母体辞書中の辞書項目を選択．
4. 追加された項目の名前を変更する．

という手順で行なわれる．辞書構造を決定すれば自動的に母体辞書との対応付けも行なわれる．辞書構造が決定されたら次は仮想辞書の見出し，結合対，結合条件を指定する．完成した仮想辞書定義文は仮想辞書定義文DBに登録され，検索時に利用される．また，他の仮想辞書の母体辞書として利用することができる．

(2) 仮想辞書検索部（付録 (2) 参照）

仮想辞書検索部では，最初に検索を行なう仮想辞書あるいは実体辞書を選択する．次に選択した辞書の辞書構造を参照しながら検索対象と検索条件を指定する．検索条件は任意の項目を選択するだけで自動的に作成される．検索は作成された仮想辞書検索文に従って行なわれるが，一度検索した検索文中の検索条件のパラメータを変更しながら再検索を行なうこともできる．

5 おわりに

仮想電子辞書の設計について述べた．本設計に基づき現在実装中である．既存の実体辞書の問題点として，記述が不統一でスキーマに沿った構造化ができない項目説明があったり，項目として切り出す時の終端子として決めたマーカが必ずしも一定でなかったり，スキーマとしてこちらが想定していない項目が出現したりする例外が出現し，構造化できないものがあった．可能な限り手作業で対応したが，なお十分ではない．これらのいくつかは，言葉が持つ多様性を十分に説明するためには，ある程度不可避であると思われる．電子辞書の規格化（たとえば，[8]）がこのあたりも含めて十分に検討され，標準化されたならば，逆に辞書の編集段階から，規格準拠を考慮でき，統一のとれた電子辞書を構築できる．本研究における仮想電子辞書の場合には，このような標準化された実体辞書が母体辞書として使用できるような環境が望まれる．この点に関して，現在，我々はXML(eXtensible Mark up Language)を用いて，仮想辞書の定義，検索，辞書情報の記述を行なうための取り扱いを検討中である．付録 (2)(3) に，仮想辞書定義のためのDTDとそれに基づく”和訳付き英語シソーラス辞書”のXMLによる定義を示す．

謝辞

和英辞書および英和辞書の元データは三省堂（株）より提供していただいた．謝意を表する．

文 献

[1] 吉田 將: 辞書構築における諸問題，情報処理，27, 8, pp.933-939, 1986.

[2] 内藤，山下，松山，柵木: オンライン辞書のハイパーテキスト化手法，情報処理学会論文誌，34, 2, pp.320-329, 1993.

[3] 武田明子，古郡延治：例文をもとにした英文書作成支援システム，情報処理学会論文誌，35, 1, pp.53-61, 1994.

[4[岩波書店辞典編集部：逆引き広辞苑，岩波書店.

[5] 山田和男 編： 新クラウン和英辞典，三省堂.

[6] 木原，福村，芦川 編： ニューセンチュリー英和辞典，三省堂.

[7]NeXT Computer Inc.: Chap.5 in ” NeXT Applications”, 1990.

[8]EPWING コンソシアム: http://www.epwing.or.jp

[9]http://www.m-w.com/cgi-bin/netdict

付　録

（1）GUIによる仮想辞書の定義

（2）GUIによる仮想辞書の検索

(3) 仮想辞書定義のための DTD

```
<?xml encoding="EUC-JP">

<!-- 仮想辞書の宣言 -->
<!ELEMENT Define ( Head, From, Body ) >

<!-- 仮想辞書のヘッダ情報 -->
<!ELEMENT Head ( DicName, Locate, Author, Update, Note ) >
<!ELEMENT DicName ( #PCDATA ) >          -- 辞書名 --

<!-- 仮想辞書の所在 -->
<!ELEMENT Locate ( #PCDATA ) >           -- 辞書の存在する URL --

<!-- 作成者 -->
<!ELEMENT Author ( #PCDATA ) >           -- 作成者名 --
<!ATTLIST Author
          Mail            #IMPLIED       -- 作成者のメールアドレス --
>

<!-- 最終更新日 -->
<!ELEMENT Update ( #PCDATA ) >           -- 更新日 --

<!-- 備考 -->
<!ELEMENT Note ( #PCDATA ) >             -- 備考 --

<!-- 母体辞書情報 -->
<!ELEMENT From ( ParentDic* ) >
<!ELEMENT ParentDic ( #PCDATA ) >        -- 母体辞書名 --
<!ATTLIST ParentDic
          Alias          #IMPLIED        -- 母体辞書の別名 --
          Locate         #REQUIRED       -- 母体辞書の存在する URL --
>

<!-- 仮想辞書の構造および項目情報 -->
<!ELEMENT Body ( Struct, Assign, Connect, Index ) >
<!ELEMENT Struct ( ( Concrete | Abstract | Assign | Connect )* )  >
<!ELEMENT Concrete >
<!ATTLIST Concrete
          Name          #REQUIRED        -- 具体項目の名前 --
>
<!ELEMENT Abstract ( ( Concrete | Abstract | Assign )* ) >
<!ATTLIST Abstract
          Name          #REQUIRED        -- 抽象項目の名前 --
>

<!-- 辞書項目の関連情報 -->
<!ELEMENT Assign ( ( Relate, Connect, Group ) | Entity* ) >
<!ATTLIST Assign
          Name          #REQUIRED        -- 関連情報を付加する辞書項目名 --
          Type   ( NUM, PENUM, UC, PRON, WORD, AWORD, SENT, ASENT, ETC ) "WORD"
                                         -- データ型に関する定義 --
>

<!ELEMENT Relate ( #PCDATA ) >           -- 対応する母体辞書項目パス --
<!ELEMENT Index ( Key* ) >
<!ELEMENT Key ( #PCDATA ) >            -- 索引を付加する項目パス --
<!ELEMENT Connect ( Entry* ) >
<!ELEMENT Entry ( #PCDATA ) >          -- 結合辞書の見出し語に代入する項目 --
<!ATTLIST Entry
          Name           #REQUIRED       -- 結合を行なう母体辞書 --
>
```

```
<!ELEMENT Where ( Expr ) >
<!ELEMENT Expr ( #PCDATA) >
<!ATTLIST Expr
          Name          #REQUIRED       -- 条件を付加する項目パス --
          Op    ( EQ, NEQ ) "EQ"        -- 演算子 --
>
<!ELEMENT Entity ( #PCDATA ) >          -- 実体 --
```

(4)XML を用いた仮想辞書定義文

```
<Define>
  <Head>
    <DicName>和訳付き英語シソーラス辞書</DicName>
    <Locate>http://pear/~yoshino/kenkyu/jet.xml</Locate>
    <Author Mail="yoshino@pear">Manabu Yoshino</Author>
    <Update>1999 11/8</Update>
  </Head>
  <From>
    <ParentDic Alias="ET" Locate="http://pear/~yoshino/kenkyu/et.xml">英語シソーラス辞書
</ParentDic>
    <ParentDic Alias="EJ" Locate="http://pear/~yoshino/kenkyu/ej.xml">英和辞書</ParentDic>
  </From>
  <Body>
    <Struct>
      <Concrete Name="見出し語"/>
      <Abstract Name="見出し語項目">
        <Concrete Name="品詞"/>
        <Abstract Name="品詞項目">
          <Concrete Name="類義語"/>
          <Abstract Name="類義語項目">
            <Concrete Name="類義語和訳"/>
          </Abstract>
          <Concrete Name="反意語"/>
          <Abstract Name="反意語項目">
            <Concrete Name="反意語和訳"/>
          </Abstract>
        </Abstract>
      </Abstract>
    </Struct>
    <Assign>
      <Relate Name="見出し語">ET. 見出し語</Relate>
      <Relate Name="品詞" Type="PENUM">ET. 見出し語項目. 品詞</Relate>
      <Relate Name="類義語">ET. 見出し語項目. 品詞項目. 語義項目. 類義語</Relate>
      <Relate Name="類義語和訳">EJ. 見出し語項目. 品詞項目. 語義</Relate>
      <Relate Name="反意語">ET. 見出し語項目. 品詞項目. 語義項目. 反意語</Relate>
      <Relate Name="反意語和訳">EJ. 見出し語項目. 品詞項目. 語義</Relate>
    </Assign>
    <Index>
      <Key>見出し語</Key>
      <Key>見出し語項目. 品詞</Key>
    </Index>
    <Connect>
      <Entry Name="EJ">ET. 見出し語項目. 品詞. 類義語</Entry>
      <Entry Name="EJ">ET. 見出し語項目. 品詞. 反意語</Entry>
    </Connect>
    <Where>
      <Expr Name="EJ. 見出し語項目. 品詞">ET. 見出し語項目. 品詞</Expr>
    </Where>
  </Body>
</Define>
```