# 記述名表現方式と意味ネットワークを利用した記述名解決法†

古宇田 フミ子†† 田中 英彦††

計算機資源の識別法として，利用者の立場を重視し，意味を持たせた名前，すなわち，対象自体の持つ機能を説明する名前（記述名）を用いた方式を実現することを目的とし，記述名で表現可能な内容の要件と，表現された記述名に向いた新しい名前解決法を提案する．現在の名前付けは，管理に必要な属性，例えば，所属等の属性を用いた方式はあるが，利用時に必要となる対象が持つ固有の機能を記述した方式によるものは見当らない．使用目的を表す機能記述による名前も必要不可欠となる．そこで，本論では，記述名を，計算機資源固有の機能を表し，使用可否の状況の説明をも持つものと規定し，主に，基本的で重要と考えられる前者の表現と，この名前解決法を考察した．機能の記述表現では，機能の記述の抽象度のクラス，記述要素間の関係表現，記述量，省略等が自由に選べるような柔軟な表現を目指し，この下で，正しく対象を示すような名前解決法を，対象に固有な性質と一時的に成り立つ性質を共に表現可能な意味ネットワーク KL-ONE を利用して構成した．記述要素の抽象度の差を半順序関係として表し，軸概念等を導入して，記述の柔軟性に対処した名前解決方式を示した．この結果，与えられた記述名と対象との関係を数理的に明示でき，資源固有の機能を述べる記述名利用の可能性を示すことができた．

## 1. はじめに

機能記述により，対象の識別を行う名前である記述名（descriptive name）を計算機資源に適応する．記述名表現法を検討し，その表現を可能とする，名前情報ベース（ディレクトリ）構成法と記述名解決法を意味ネットワークを用いた方式により，提案する．

これまで，計算機資源の識別法は，管理に都合よい名前が付けられることが多く，物理的または，論理的構成法を反映した階層構成を採るものが多い．例えば，Clearinghouse[9] は組織名などを含んだ三層構成の識別名を持ち，Grapevine[1] は二層構成の識別名である．この場合，名前の構成要素順は固定され，省略ができない．値を正確に知っている必要がある．

Sollins[13] は，人間が日常使う名前についての考察から，名前に関する五つのカテゴリを示し，これに基づく分散処理向きの名前管理法を提案した．この中で，名前付けの考え方の比較のための共通指標として，割付，解決，使用，曖昧さまたは唯一性，意味の度合，の五つの観点を挙げていることが注目される．

この指標によると，これまでの識別名は，名前の割付が固定的で，省略できないので，唯一性は保障されやすい．解決法も木構造を採る等，比較的容易であるが，使用時に，正確に知らないと使えない等の制限がある．管理構造に重点を置いた名前付けなので対象自体の持つ機能の意味は明確でない．

上記の問題点は，名前を管理用の観点から構成したために起こる．本論では，名前を管理用の観点ではなく，使用時の利便性を考慮し，意味の度合を重視した名前，すなわち，対象の機能を説明する名前により，計算機資源の識別が可能な名前管理法を新たに示す．これまでの識別名では実現できなかった表現の順序や省略の自由度等，曖昧さの表現の可能性についても探る．これらを考慮する必要があることから，解決法は従来の識別名の方式とは異なる構成となる．記述名の定義は，IS 7498-3 名前とアドレス[7]を利用する．

IS 7498-3 によると，「A name that identifies a set of one or more objects by means of a set of assertions concerning the properties of the objects of the set」である．7498-3 では，定義のみでこれ以上詳しい規定はない．

以下では，記述名の表現のための要件と形式を2章で，記述名の情報データベースの基本的構成法を3章で議論する．さらに，従来の識別名解決法とは根本的に異なる考え方を用いた記述名の解決法を4章で展開し，5章でこれらの検証と考察を述べる．

## 2. 記述名表現の要件と記述名管理法

### 2.1 記述名表現に必要な項目

IS 7498-3 の定義では，記述名は，対象の特性，性質の記述を用いて対象を識別するための名前となる．この定義では，対象の特性，性質の記述の範囲や表現法の規定がない．そこで，考察範囲を計算機資源に限

† Representation of Descriptive Name and the Resolution Method with a Semantic Network Structure by FUMIKO KOUDA and HIDEHIKO TANAKA (Department of Electrical Engineering, Faculty of Engineering, University of Tokyo).

†† 東京大学工学部電気工学科

定し，以下の条件で表現法を検討する．

計算機資源を利用する立場からは，1) 資源の持つ処理機能や役割が分かること（例，コピー機能を持つ，蓄積用等の説明），2) この処理や役割が一時的に使用可能なのか，常に可能か，等の使用可否の条件が分かること，の二点は，重要な情報である．属性を用いた識別名の例として，CCITT の X. 500[8] がある．X. 500 では，その使用目的から，管理上必要な属性，例えば，国，組織，等を扱っている．しかし，目的とする対象固有の属性記述は行っていない．本論で提案する記述名は資源自体の持つ機能の記述表現を目的とする．

資源の機能は必ずしも画一的には表現できないから，記述名では表現の柔軟性も考慮する必要がある．

処理機能の記述では，異機種環境でも共通に使用可能な一般的，抽象的な値で指定し，細かな差異には言及しない場合と，機種の差に応じ引数の指定等，具体的に指示する必要がある場合がある．ある機能の記述に種々の抽象度のクラスを持たせた記述表現が必要となる．抽象度によらず機能を共通に規定する概念を属性型と呼び，抽象度により変化する個々の値を属性値と呼ぶ．例えば，機能ファイル全体は，「ファイルタイプ」は「能動的」と指定し，より具体的なコマンドファイル全体は「ファイルタイプ」は「コマンド」と指定し，この中で，組込みコマンドだけを知りたい時は，「ファイルタイプ」は「組込みコマンド」と指定するであろう．この時，属性型は「ファイルタイプ」であり，「能動的」「コマンド」「組込みコマンド」は属性値と見ることができる．一つの属性型に抽象度のレベルが異なる属性値が存在し，各々に抽象度の異なる記述となっている．

属性型には，計算機の利用者名では，アクセス権，等が考えられ，ファイルでは，版，コード形式，処理機能，入力形式，等が考えられる．属性値は，型に応じて種々に決まる．他の例を見ると，Profile[11] では，属性型をタグ (tag) と呼び，name, address, phone, mail, login, home, 等を用いている．

対象の特性 (property) の成立の期間や場所が，一時的か，恒常的かの指定も必要になる．

特性を複数記述する場合，暗黙の了解として，どの条件も満たす and の関係を示す．特性間の関係記述で，and 以外の関係（包含関係，従属関係，等）記述も可能であれば，目的とする対象の持つ性質自体の関係（例えばコンパイラの版と使用可能な共通ファイルとの関係等.）も規定できる．したがって，特性間の関係記述ができることが望ましい．

一方，これらの機能をすべて指定した記述を用いなければ対象が発見されないわけではないから，記述要素の量を適度に選び，記述の省略を自由にできるようにすることも必要になる．

### 2.2 記述表現で表される対象

記述名が識別子として機能するためには，記述表現されたものがどの対象を指すかが明確にならねばならない．記述名は意味を重視するものであり，必ずしも識別の唯一性を目指すわけではないから，ある記述で表される対象が複数ある場合は，集合として，これらのいずれをも示し得るという立場を採りたい．

記述表現に柔軟性が必要なことを考慮すると，記述の抽象度により示される対象がどのように変わるか，省略や記述要素順が異なる表現の時，どんな条件ならば，同じ対象を表し得るかが問題となる．この解決法は本論の論点の一つである．

### 2.3 記述名に必要な形式

属性を記述する名前の表現形式は，特性のタイプを示す属性型と特性の値である属性値との関係で表現し，{属性型 operator 属性値} の形式を持つことが多い．例えば，Profile は tag＝value を組み合わせた形式であり，X. 500 では属性の表現はこの形式を持つ．L. L. Peterson の yellow-pages service[10] では，〈type, operator, value〉の形を取っている．operator は，＝に留まらず，〈や〉も表現できる（例 time＞3 msec）．X. 500 でもフィルタを用いると同様な表現が可能である．

本論では，考察の第一歩として，{属性型＝属性値} のみを調べ，属性型同士の関係も表記する．

記述名表現法は，対象に固有な性質を表すことを目的とする．属性型と属性値を等号で表し，属性型同士の関係が共通性 (and) ならばこれらを並べ，従属や包含関係ならばこの関係を陽に示し，これらの順序は問わず省略も自由な形式を基本形とする．一時的な関係である述語表現も記述可能なものと考える．対象の持つ機能記述の表現法は重要と考えるので，以下では，記述名は基本形のみ考える．

記述名の例を示す．印刷コマンド全体を指すものは，

{file_type＝command,functional_description＝print, input＝data_file, output＝print_device, class＝user, capability＝access_mode, access_check (class, file),

function (description, input, output), etc.}

その中の一つは,

{file_type=built_in_command, functional_description=lzrprint, access_check (class: subject, file: object), function (description: print, input: data_file, output: lapr1), etc.}

という記述になる.

### 2.4 記述名管理（解決）方式の選択

名前解決法は，与えられた名前から対象を引き出す機構である．名前解決では，変換を行う拠り所となる，一般にディレクトリと呼ばれる情報源の表が必要となる．この表の作り方と解決法は一体であり，どのような表を用いるかで解決法が変わると考えられる．

筆者らは，記述名の要件の考察から，記述名解決には，記述の抽象度のクラスや記述要素間の関係が分かり，記述で省略された属性を復元でき，記述の要素順に依存しない解決法を図れるものがディレクトリとして必要であると考えた．以下に，これまでに行われている主な名前解決法に触れ，選択の参考としたい．

識別名の名前解決では，変換表，コンテクスト写像(context mapping)，木構造のディレクトリ等を利用している．いずれも，必要に応じて，途中に中間名などを用いた識別子間の変換である．この場合の解決法は，変換の対応関係を辿ることで対象を見いだすものである．例えば，UNIX OS における i-node では，{アドレス，識別子} の組で表された変換表を持つ．Watson らのコンテクストマッピング[16]は，あるレベルの識別子をコンテクストと写像函数とにより，次のレベルの識別子に変換し，最終的には対象を求める方式である．Comer ら[3]は，分散処理のモデルを環境とリンクから成るものとし，その上で，名前解決機構を示した．名前にはいくつかのレベルがあり，名前解決では，名前のレベルだけでなく環境も変わる，としている．名前変換では，名前間の写像を表すコンテクストと名前の対（qualified name）を用い，コンテクスト間の関係を示す naming network を利用してプリミティブ名を引き出している．

属性を用いた名前の場合は，属性を解釈する必要があるので，単なる変換だけでは不十分となる．

例えば，X. 500 では，各ノードが属性型と属性値の集合から成る DIT から作られる木構造の DIB (Directory Information Base) をディレクトリとして利用している．この名前解決は，根を始点として属性の種類を見ながら下方へ向かう．DIB では，木構造のレベルごとに都市，組織等の異なる範疇を表し，search, modify 等のオペレーションで操作する．そのため，識別名 (Distinguished name) は，木構造の構成要素順を反映したものとなる．

Terry[14] は，分散処理での管理の分散を図るために，属性を利用した，構造化していない名前を扱った．名前サーバのオーソリティ属性の集合から成る構造データベースをクラスタ化条件を用いてオーソリティ属性から成るコンテクストに分割し，管理を分散した．木構造の一レベルに対応するクラスタ化条件により属性を文字列として名前のレベルを決め，分類している．名前解決は，コンテクスト間を繋ぐ，コンテクスト結合を辿り (resolution chain) 名前サーバのオーソリティ属性を求めるものである．

Profile は，属性に基づく名前 (attribute_based naming) サービスを扱っている．属性から主管理者 (principal) への写像関係をデータベースとして構成し，記述名の解決には，このデータベースと四種の解決関数 (interpret function) を用いている．ユーザインタフェースで利用者の必要性に応じて，解決関数の種類を指定する必要がある．この名前体系の数学的考察はアリゾナ大の技術報告[4]に詳しい．

これらは，属性からなる名前の名前解決であるが，Terry の場合は，属性を文字列として扱っている点で，識別名解決に近い．X. 500 の木構造の場合は，レベルごとに異なる解釈を持つ構成になるので，要素順が大きく影響する．また，ノード間の関係が場所ごとに意味が異なるので，要素を追加する時の配置場所や，ノードの持つ機能の継承の表現にも向かない．Profile のデータベース型は，記述の省略や記述要素順に依存しない解決法であるが，属性の抽象度の差異が明確でなく，記述要素間の関係が見えないことから意味解釈のための関数が必要となる．

記述の抽象度は，抽象的なものから具体的なものまで，同じ型を継承しているので，記述名解決には，何らかの継承表現が可能で，しかも解決に有効なものを使いたい．筆者らは，ノードの表す意味は常に一定で，ノードの持つ機能の継承表現や機能間の関係表現が可能で，木構造の関係やデータベースとしての側面も持ち合わせる意味ネットワーク表現を記述名解決に利用することとした．

意味ネットワークは，1968年，Quillian により，初めて提唱された．意味ネットワーク表現は，概念，特性や，これらの関係を知識として表すもので，節を概

念とし，リンクが節間の関係を表す継承可能なグラフ表現である．リンクを上に辿るとより抽象性が高まり，下に向かうとより具体的になる．葉ノードは個々のものを示す．特性の継承は，複数の上位ノードから起こることもある（multiple inheritance，多重継承性）．リンクは木構造の性格も持ち，対象の属性に関する表現を行い属性の種類に応じて分類が可能である．

ところが，初期の意味ネットワーク表現では，例外と標準値（デフォルト）の表現に難があり，今日においても，この点の扱い方について種々の解決法が試みられている[12),15)]．その中で，KL-ONE[2),5)]は，標準値を排し，例外を認めず構造性を強く持った意味ネットワークである．KL-ONE は，必要性（necessity）を述べた記述部と，状況性（contingency）を述べる宣言部を明確に分けた構造を持つ．記述部は，ノードが概念（concept）から成り，概念間のリンクで概念の持つ構造を継承する意味ネットワークである．概念の持つ構造記述要素は，役割（role）とその制約の記述としての補概念（filler，値と数（value/number restrictions）がある．），役割値写像（role value map），構造記述（structural description，引数表現）から成り，これらの関係により概念の必要条件としての内部構造を説明する．概念には，クラスに当たる一般概念（generic Concept）とインスタンスに相当する個別概念（individual Concept）があり，個別概念は葉ノードのみ可能である．下位概念（subConcept）と上位概念（superConcept）は包含関係のリンク（SuperC link）で結ばれ，下位概念では上位概念の持つ役割と補概念の構造が例外なしに継承される．ある概念で役割を加えることもできる．加えられた役割も下位概念に継承される．そのため，下位概念ほど上位概念からの制約が強い．例外は認めない．これを構造継承意味ネットワーク（Structured Inheritance Network, SINet）と呼ぶ（**図 1**）．

宣言部は存在と記述の相互参照を示すいわゆる一階述語の構成となる．KL-ONE では，nexus, description wire, context から構成される．これは，一時的関係，例えば，「対象の存在場所は某所である．」，「対象の管理者は誰である．」等の表現に当たる．

### 2.5 記述名と情報管理ベース（SINet）の対応付け

KL-ONE の記述部の概念の持つ構造は，役割，補概念とこれらの関係表示による概念の属性記述となっている．役割は補概念に対する型を示すと考えられるから，SINet の役割を記述名の属性型に，SINet の補概念を属性値に対応させる．すると，属性型と属性値は役割と補概念との関係で表現され，属性型同士の関係は役割間の関係（役割値写像や構造記述）に対応付けられる．

この対応付けにより，記述名で表される内容に対応する概念が SINet 上で決まる．

## 3. 意味ネットワークの構造解析とその利用

### 3.1 意味ネットワーク KL-ONE 記述部（SINet）の構造

この節では，KL-ONE 記述部で観察されることを述べそれに若干の解釈を加える．以後，概念の表記法として，注目している概念を pcon, pcon の上位概念を supcon, pcon の下位概念を subcon,（いずれも集合を表す場合は大文字で始める．）とする．

（1） KL-ONE では，前述のとおり，概念は一般概念と個別概念に区別され，これより細かい分類は行っていない．一般概念では，物理的に存在するものを表したものと，形のないものを述べた概念が混在する．本論の目的は，計算機資源として存在するものの記述名を構成することにあるので，便宜上，一般概念を実在するものを示す実体概念と形のないものを表す抽象概念とに細かく分ける．主に実体概念に注目したいので，以後，実体概念のリンクを幹と呼び，幹について考える．幹として，ファイル系とユーザ系を考える．

（2） 一般概念のリンクの上下は概念の持つ構造に関して，包含関係を表す．二種類の関係が提示されている．

a）一つの一般概念が複数の下位概念に繋がる場合（制限，restriction），各下位概念では，上位の概念の持つ役割に対する補概念値はより具体的な値になる．枝分れすることにより，一つの概念の性質をより具体的に記述，分類したことになる．

b）一つまたは複数の概念が一つの下位の概念に繋がる場合（接合，conjunction）は，下位の概念は（一つまたは複数の）上位概念の性質を持ち合わせた（多重継承性）概念となる．

（3） 役割と補概念の組は概念の持つ機能を説明する．役割を一つ固定すると，補概念の値はリンクの上位にあるものほど抽象度が高く，下位ほど具体性を帯びる．そこで，一つの役割に対する補概念の抽象度の関係を，より抽象度の高いものを前に，具体的なも

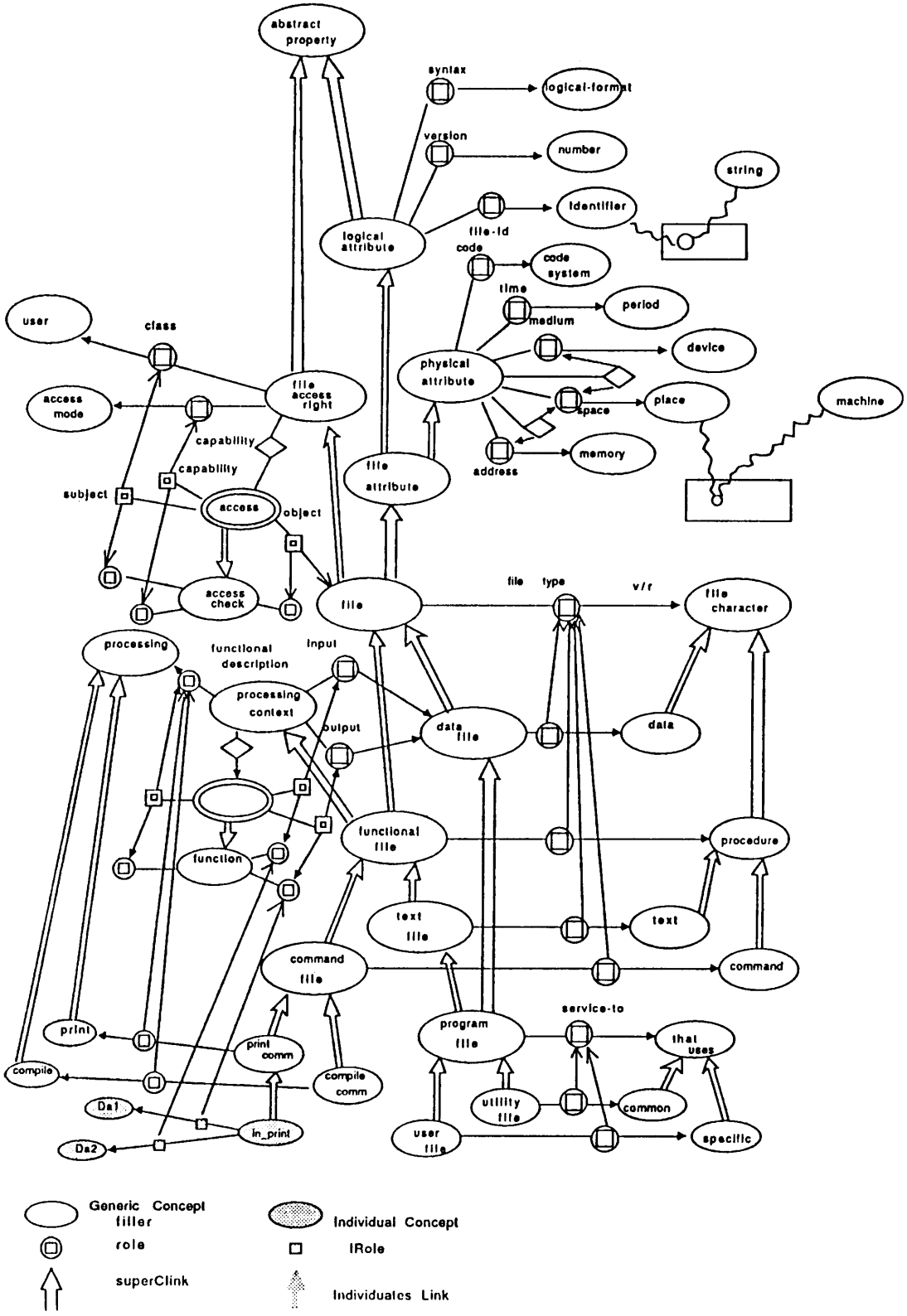


図 1 意味ネットワークの例
Fig. 1 Example of SINet.

のを後ろと見て，リンクに沿った前後関係（表記 $\leqq$ ($<$, $=$)）を新たにつける．（2）の構造から，以下の関係が分かる．ただし，役割を rol ($i$)，補概念を fil ($i, j, k$)，($i$：役割の識別番号，$j$：半順序の始点からの順序数，$k$：補概念自体の識別番号）とし，注目している概念 pcon の持つ役割を rol($i$)，その補概念を，fil($i, j, k$)，Subcon の対応補概念を fil($i, j+1, k+m$)($m=1, .., n$) とする．

a）この半順序関係は，一つの役割と，これに関する補概念の組から成り，この始点は新たに役割が生ずるある概念上である．終点は個別概念に対応する補概念である．

b）概念間のリンクが接合の場合，半順序関係は，各役割について，

(rol($i$), fil($i, j, k$))=(rol($i$), fil($i, j+1, k+1$))

c）概念間のリンクが制限で $n$ 個の枝に分かれる時，

（i） rol($i$) の補概念がより具体的に説明される，すなわち，制限が生ずる場合，半順序関係は，$m=1, .., n$ として

(rol($i$), fil($i, j, k$))$<$(rol($i$), fil($i, j+1, k+m$))

（ii） rol($i$) が制限に関係しない役割の場合，すなわち，同じ概念にある他の役割に制限による分類が生ずる時，補概念値は変化しないから，半順序関係は，

(rol($i$), fil($i, j, k$))=(rol($i$), fil($i, j+1, k+m$))

となり同じ役割と補概念の値を持つ枝に分割される．

d）多重継承性では，上位概念で枝分れした同じ役割が異なる補概念値を持って複数の上位概念から伝わってくる可能性がある．この場合，両方の性質を持ち合わせる，という趣旨から，複数の補概念値を許すことにする．すなわち，二つの上位概念の半順序関係が各々，(rol($i$), fil($i, j, k$)) と (rol($i$), fil($i, m, n$)) の時，(rol($i$), fil($i, j, k$)$\cup$fil($i, m, n$)) とする．

役割ごとに各々の半順序系列を持つので，SINet は，始点の異なる複数の役割による半順序系列を持ち，これらにより枝分れした構成を持つ幹がいくつか集まったものと見なせる．そのため，概念間のリンク関係と一つの役割に対する半順序関係の枝分れの関係は必ずしも一致しない．例えば，図1で，ファイル概念では，役割「版」を上位概念から継承し，新たな役割として「ファイルタイプ」が生ずる．「版」は同じ補概念値を採りながら，ファイル概念の下位概念，データファイル，機能ファイルに分かれる．

（4） 個別概念自体の識別子を識別名とすることができる．この識別子で対象を指すと見なす．個別概念には，時には指す値が変わる宣言（assertion）関係が観察される．この識別子で従来の名前管理と関係がつく．

なお，一般概念自体の名前は，この概念の持つ役割や機能の総称（generic name）を表すと考える．

### 3.2 意味ネットワーク KL-ONE 記述部の分割表現

SINet では上位概念の節で加わった役割構造は必ず下位概念まで継承される．概念に役割構造を新たに加えたり，役割をより具体的に制限するような記述により，概念に機能が追加される．下位概念ほど，機能の記述量は増す．もしも，上位概念と下位概念との関係が分かれば，同じ記述を繰り返す必要はなく，記述量を減らすことができると考える．

そこで，継承関係と機能追加が明示でき，これに関する役割の半順序関係は値が変化したところ（半順序関係の始点と値が変化した場所）のみ表示するように，概念を一単位として，SINet の幹の概念を分割する．すなわち，一般概念の場合は，1）上・下位概念間のポインタとこれらの概念との継承関係，すなわち，接合なのか制限なのか，制限ならば，上位概念のどの役割を制限し，具体化したかの記述（この役割と新たな補概念値表示 restrole），どの役割をどのように制限すると下位概念になるか（下位に続く半順序系列）の記述．接合や多重継承性の時は上位概念へのポインタは二つ以上あるが，前節（3）に従って，変化しない役割，補概念値の組は記述しない．2）注目している概念の持つ構造や，ここで新たに加わった役割の記述，すなわち，役割の半順序関係の始点表示 (new-role) をする．以上の要素から成る単位に分割する．

一方，個別概念の場合は下位概念がないこと，個の集合を示す必要があることから，1）上位概念へのポインタと継承されてくる制限役割，2）個別概念の制限役割に対応する補概念の記述，個別概念に対応する識別名へのポインタを一単位とした．

この分割の一単位を各々，一般概念機能単位（generic conceptual functional unit, G-CFU），個別概念機能単位（individual conceptual functional unit, I-CFU）と呼ぶ．両方を纏めた時は CFU と呼ぶ．

## 4. 記述名解決法

### 4.1 記述名と SINet の関係付けの解釈

#### 4.1.1 SINet 上の役割，補概念の組と軸概念

SINet 上で，ある概念Cの持つ役割と補概念値の組は，この役割に関する半順序系列上のある点になる．この役割と補概念の組は，この半順序系列で，より後にある要素を最も抽象化したものである．そこで，この役割と補概念値の組に対応する概念Cを用いると，半順序系列のより後にある役割と補概念値の組を持つ概念全体を代表すると見なせる．役割の半順序系列で補概念値が変化しない時は，複数の概念が一つの役割と補概念値に対応するので，この中の概念リンクの中で最初の枝分れ直前の概念を一つ選ぶ．すると，役割と補概念の組に対応する概念が，必ず一つ決まる．この概念を軸概念（pvt：pivot concept），軸概念とこれより下位の概念全体を属性解集合 sol( ) と呼ぶ（図 2）．

軸概念を pcon，下位概念の集合を Subcon とし，pcon に対する役割，補概念を各々 rol($i$)，fil($i, j, k$) とした時，軸概念，属性解集合は，

pvt(rol($i$)，fil($i, j, k$))＝pcon

sol(rol($i$)，fil($i, j, k$))＝{pcon，Subcon}

sol(rol($i$)，fil($s, j, k'$))＝$\phi$（ただし，$i \neq s$）

となる．

#### 4.1.2 半順序関係と軸概念

軸概念になり得るのは前項の決め方から，半順序の始点，制限により $n$ 個に分岐する時と，終点である．概念間のリンクが制限で $n$ 個の枝に分かれ，rol($i$) に対応する概念で制限が生じ，補概念値がより具体化される時，半順序関係は，$m=1, \cdots, n$ として

(rol($i$)，fil($i, j, k$))＜(rol($i$)，fil($i, j+1, k+m$))

であり，変化直前の (rol($i$)，fil($i, j, k$)) に対応する概念が軸概念となる（4.1.1 項参照）．

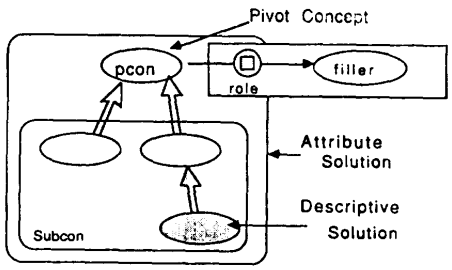


図 2 軸概念，属性解集合と，記述解集合
Fig. 2 Pivot Concept, Attribute Solution and Descriptive Solution.

概念は異なる役割を持ち得るので，この概念が異なる役割と補概念の組の共通の軸概念となることもある．個別概念は常に軸概念となり得る．

CFU では，new-role があるか，restrole がある場合に軸概念になる．ある概念で new-role，restrole 共に生じない（零個）ならばこの概念は軸概念にならない．

#### 4.1.3 記述名で表される SINet 上の概念の集合

記述名の要素の属性型と属性値の組は，2.5 節より，役割と補概念の組に対応付けられる．この対応付けを利用すると，記述名の一つの属性型と属性値の組に対応する SINet 上の軸概念が決まり，同時に SINet 上の属性解集合 sol( ) が定まる．既に，2.3 節で記述名の形式を要素間の従属性や包含関係とこれらの共通部分からなると定義した．属性型間に従属関係がある場合は，概念の持っている性質の関係となるので，軸概念には影響を受けない．したがって，記述名解決は記述名のすべての属性型と属性値に対応する属性解集合 sol( ) を求めることに帰着する．特に，最終の軸概念に対応する属性解集合に含まれる個別概念は，対象自体を表すと見なせる（3.1 節）ので，この個別概念の集合を記述名で指す対象と見なすことができる．これを記述解集合 dsol( ) と呼ぶ．この考え方を用いると，与えられた記述名に対し，(属性型，属性値) -> (役割，補概念) -> 属性解集合 -> 記述解集合 -> {対象} の対応付けにより記述名解決が可能となる．

#### 4.1.4 記述名の要素と軸概念

記述名の要素に対する軸概念を $Pn$ ($n=1, \cdots i, \cdots j, \cdots m$) として，これらの関係を調べる．

定理 1 二つの軸概念 $Pi$，$Pj$ の関係は以下のいずれかである．

1) $Pi$ と $Pj$ が同じ概念である．2) $Pi$ と $Pj$ が上下のリンクで（概念を介して）繋がっている．3) $Pi$ と $Pj$ がともに同じ下位概念 $Ck$ に繋がる．4) 二つの軸概念に共通な下位概念がない．

証明） 軸概念 $Pi$，$Pj$ を作る役割と補概念の組（各々 $(ri, fi)$，$(rj, fj)$ とする．）に対する半順序系列を各々，$L1$，$L2$ とする．

a） $L1$，$L2$ が同じ役割 ($ri=rj$) に対する系列ならば，$Pi$，$Pj$ は，前後関係となる（記述名の要素で同じ属性型に異なる複数の属性値を持つ場合に起こる．）．

b） $L1$，$L2$ が異なる系列で，ある概念 $C1$ を共通に通る（$ri$ と $rj$ が多重継承で $C1$ に加わる場合や同じリンクに $ri$ と $rj$ がある場合）とすると，$Pi$，$Pj$ は，$C1$ を通して前後関係を持たせることができる．

$Pi$ は，$C1$ より上位概念か，等しいか，下位概念かのいずれかの関係があり，$Pj$ との関係も同様となる．この関係を ≪(等しい時は＝とする．≪は推移律を満たす.) で表すと，($Pi$ と $Pj$ に関係が付く時，$Pi$, $Pj$ は対称なので) $Pi$ を $Pj$ より下位でないとして，

A) $Pi=Pj \ll C1$, $Pi=Pj=C1$, $C1 \ll Pi=Pj$,

B) $Pi=C1 \ll Pj$, $Pi \ll C1=Pj$, $Pi \ll C1 \ll Pj$, $C1 \ll Pi \ll Pj$, $Pi \ll Pj \ll C1$,

C) $Pi \ll C1$, $Pj \ll C1$, $Pi$ と $Pj$ は無関係

D) $C1 \ll Pi$, $C1 \ll Pj$

これを $Pi$ と $Pj$ のみの関係で見ると，A) のグループは $Pi=Pj$, B) は $Pi \ll Pj$, C) は $Pi$, $Pj$ は関係がないが，共通の後の要素 (概念 $C1$) を持つ．D) は $Pi$ と $Pj$ に共通の後の要素がない，のいずれかとなる．

c) $L1$, $L2$ が異なる系列で，どの概念上でも交わらない時は，$Pi$, $Pj$ は関係が付かない．

$L1$, $L2$ の関係は上記 a)～c) ですべてを尽くしている．以上より，$Pi$ と $Pj$ の関係は，1)～4) のいずれかとなる． □

定理2　関係(≪)による二つの軸概念 $Pi$ と $Pj$ に共通な後の最初の軸概念 $Pk$ (後軸概念) は以下のようになる．

1) $Pi$ と $Pj$ が同じ概念の時は軸概念は変わらない ($Pi=Pj=Pk$). 2) $Pi$ と $Pj$ が上下のリンクで繋がっている時は，$Pj$ が新しい軸概念となる ($Pi \ll Pj=Pk$). 3) $Pi$ と $Pj$ 間には関係がなく，$Pi$ と $Pj$ がともに同じ下位概念に繋がる場合，その共通な下位概念で，軸概念の条件を満たす最初の概念 $Pk$ が新しい軸概念となる ($Pi \ll Pk$ または，$Pi \ll Pk$). ただし，$Pk$ は (記述省略等で) 必ずしも与えられた記述名の要素に対する軸概念とは限らないが，これも記述名の軸概念と見なす (導出軸概念). 4) 二つの軸概念に共通な下位概念がない時は，新たな軸概念は決まらない．

証明) 1), 2), 4) は定理 1 より明らか．

3) は，定理 1 の $C1$ が軸概念の条件を満たせば，$Pk=C1$. 軸概念にならない時は，$C1$ は new-role も restrole も持たないので，下位概念は一つ ($C2$) である．$C2$ が軸概念の条件を満たせば，$Pk=C2$. そうでない時は，$C2$ について繰り返す．個別概念は軸概念となるのでこの処理は必ず終了し，最初の後軸概念が見つかる． □

定理3　軸概念 $Pi$ と $Pj$ の各々の属性解集合の共通部分は，$Pi$ と $Pj$ の関係に応じて決まる新たな共通な後の最初の軸概念 $Pk$ の属性解集合と等しい．

証明) $Pi$ の属性解集合を sol ($Pi$) と表すと，定義より

$$Ca \in \mathrm{sol}\ (Pi) <=> Pi \ll Ca \quad (\text{または } Pi=Ca) \qquad (☆)$$

A) $Pi$ と $Pj$ が等しい時，$Pi=Pj=Pk$

B) $Pi \ll Pj$ ならば，$Pk=Pj$ となる．一方，(☆) より，$Pj \in \mathrm{sol}\ (Pi)$ である．また，$Ca \in \mathrm{sol}\ (Pj)$ ならば，$Pj \ll Ca$ だから，$Pi \ll Pj$ と併せて，$Pi \ll Ca$ より $Ca \in \mathrm{sol}\ (Pi)$, すなわち，$\mathrm{sol}\ (Pj) \subset \mathrm{sol}\ (Pi)$ である．これより，$\mathrm{sol}\ (Pi) \cap \mathrm{sol}\ (Pj) = \mathrm{sol}\ (Pj)$ となる．

C) $Pi$ と $Pj$ には関係がないが，両者に共通な最初の後の軸概念 $Pk$ が存在する場合，$Pi \ll Pk$, $Pj \ll Pk$ である．すなわち，$Pk \in \mathrm{sol}\ (Pi)$, $Pk \in \mathrm{sol}\ (Pj)$, となり，これより，$Pk \in \mathrm{sol}(Pi) \cap \mathrm{sol}\ (Pj)$, すなわち $\mathrm{sol}\ (Pk) \subset \mathrm{sol}\ (Pi) \cap \mathrm{sol}\ (Pj)$ である．

$Pu \in \mathrm{sol}\ (Pi) \cap \mathrm{sol}\ (Pj)$ となる軸概念 $Pu$ をとる． (☆)

より，$Pi \ll Pu$, $Pj \ll Pu$ である．もし，$Pu \ll Pk$ とすると，$Pu$ は $Pi$ と $Pj$ に共通な後ろの最初の軸概念となり，$Pk$ が最初であることに反する．すなわち，$Pk \ll Pu$ となる．したがって，$Pu \in \mathrm{sol}\ (Pk)$ すなわち，$\mathrm{sol}\ (Pi) \cap \mathrm{sol}\ (Pj) \subset \mathrm{sol}\ (Pk)$.

以上より，$\mathrm{sol}\ (Pk) = \mathrm{sol}\ (Pi) \cap \mathrm{sol}\ (Pj)$ となる．

D) $Pi$ と $Pj$ が交わらない時は，$\mathrm{sol}\ (Pi) \cap \mathrm{sol}\ (Pj) = \phi$ となり共通な属性解集合はない． □

定理4　与えられた記述名が何らかの対象を表すための必要十分条件は，記述名の各要素に対して定まる軸概念 $Pi$ ($i=1, \cdots, m$) 間の関係が，定理 1 の 1)～3) の関係になることである．

証明) =>(背理法) 定理 1 4) の関係が生ずると定理 3 D) より，共通な属性解集合を持たない．

<= 定理 2 の 1)～3) から，等しい軸概念は同じと見なし，異なる二つの $Pi$, $Pj$ に対しては，必ず，$Pi \ll Pj$ または，第 3 の軸概念 $Pk$ があり，$Pi \ll Pk$, $Pj \ll Pk$ である．これらを併せると軸概念は≪の関係で逆木構造の半順序となる．このことは，最終の軸概念 $Pe$ があり，他の軸概念と必ず関係が付く (☆☆) ことを示している．$Pe$ に対応する dsol ( ) が求める対象となる． □

定理5　定理 4 の条件を満たす記述名では，最終の軸概念 Pend が一組決まり，しかも，Pend は途中の軸概念の比較順序によらない．

証明) 前半は，(☆☆) である．後半は，同じ記述名

Aで二つの異なる比較順を各々 $C1$, $C2$ とし，$C1$ により得られた最終軸概念 $Pe1$ が $C2$ では最終軸概念にならないとすると，Aのある要素に対応する軸概念 $Pc2$ で，$Pe1 \ll Pc2$ となるものがある．すると，$Pc2$ は $C1$ では最終ではないことから定理4の証明より $Pc2 \ll Pe1$ となる．したがって，$Pe1 \ll Pc2 \ll Pe1$ となり，$Pe1 = Pc2$ である． □

定理6 異なる表現の記述名が同じ対象を表すための条件はこれらの最終軸概念が等しいことである．

証明) =>同じ対象を表す二つの記述名の表現をA，Bとする．A，Bに対する最終軸概念を各々 $Pae$, $Pbe$ とする．$Pae$ と $Pbe$ が，定理1 4) の関係とすると，AとBは違うものを表すことになる．$Pae$ と $Pbe$ が，定理1 3)の関係とすると，定理2 3)よりこれらに共通な後の軸概念が存在し，両者とも最終軸概念にならない．$Pae \ll Pbe$，とすると，sol ($Pbe$)$\subset$sol ($Pae$) となり，AとBは同じ対象を表さない．$Pbe \ll Pae$ も同様．したがって，$Pae = Pbe$ である．

<= 二つの表現に等しい最終軸概念を $Pe$ とする．これに対応する dsol ( )が同じ対象を表す． □

### 4.2 記述名解決方式の考え方と解決法

記述名解決法は，与えられた記述名の要素（属性型と属性値）に対応する軸概念がSINet上のどこにあるかを調べ，これらの軸概念から前節の定理群を利用して≪の関係で最後の軸概念を見つけ出すことである．

属性からSINetに対応付けることで，軸概念を探してもよいが，定理5より比較順によらないことを考慮して，発想を逆にし，概念の持つ役割と補概念の組が記述名の属性型と属性値に一致するか，という考え方をとる．概念は複数の役割を持ち得るので，一つの概念のどの役割についても調べる．ある概念で，a）役割と補概念値の組が記述名の属性型と属性値の組と一致した時は，これが軸概念となる．この軸概念と，これまでに決まっている後軸概念（ただし，後軸概念の初期値＝abstract_property）とを定理2により比較し，両者に共通な後の最初の軸概念を新しい後軸概念とする．定理1 4）の関係であったら終了する．次に調べる概念は≪の関係で後ろの順序となるべきだから，新しい後軸概念の下位概念に限定する．b）概念上で役割のみが属性型と一致する場合，属性値と一致する補概念はこの役割の半順序系列上にあるから，この概念の上位概念と下位概念を調べる．b-1）上位概念は new-role を持つところまで遡る．途中で補概念と属性値が一致すれば，この概念を軸概念とし，a）と同様の処理を行う．b-2）一致しない時は，下位概念を次に調べる候補のリストに加える．これらの処理には，軸概念と調べる候補概念の管理表と，記述名の要素と半順序系列の関係を管理する表を用いる．前者は調べる概念の二重チェックも行い，探索のループを防止する．c）一つの幹内で軸概念がない時は，後軸概念は初期値のまま変化しない．この時は，別の幹に移る．管理表で処理の二重チェックを行っているので，他の幹に移動する．

以上より，解決法は，管理表を見ながら，与えられた記述名の属性型に対応する役割の半順序系列に沿って，新しい後軸概念を決めることを，記述名の属性全部について調べるか，または，管理表で調べるべき概念がなくなるか，または，定理1 4）の関係が生ずるまで繰り返す．後二者の場合は，求める対象は存在しない．

## 5. 検証と考察

### 5.1 記述名解決法の検証

#### 5.1.1 実験項目

同じ対象を表すと考えられる記述名を形態を変えて与え，探索の入口を幾つか変えた場合，SINet上で探索概念の数と軸概念決定処理の様子を調べる．

記述名の形態は2.3節の意味の基本形について1) 記述の抽象度の違い，2) 記述の省略の度合の違い，3) 記述の要素順の違い，4) 間違い記述名の場合，を考える．

探索の入口は，I．目的の対象が存在する幹に入った場合，1) ある役割の半順序系列の始点より前，2) ある役割の半順序系列の始点上，3) ある役割の半順序系列の途中の点，4) ある役割の半順序系列の終点，5) ある役割の半順序系列の終点より後，II．6) 目的の対象とは異なる幹に入った場合，の各々について調べる．

検証では，図1のSINetを用いて，一般概念 program_file と個別概念 in_print を記述する記述名 UDN[$i$] を以下のように設定した．

program_file の記述では，最終軸概念が program_file になるようにし，記述順と記述の省略について調べる．

UDN[1]：属性型と属性値の組を，SINet上で下位概念の役割と補概念の組に対応するものから順に省略なしに並べた記述．15種類の属性型と属性値の組からなる．

UDN[2]: 属性型と属性値の組を，ランダムに省略なしに並べた記述．15種類の属性型と属性値の組からなる．

UDN[3]: UDN[1]の記述を省略し2/3(10種類)にしたもの．

UDN[4]: UDN[2]の記述を2/3にしたもの．in_printの記述では記述の抽象度を比較する．UDN[5]とUDN[7]は最終軸概念がin_print，UDN[6]のそれは，functional_fileになるように設定した．

UDN[5]: 7種類の機能を記述したもの．

UDN[6]: UDN[5]の記述の属性値をより抽象化したもの．属性型と属性値の組は7種類で，同じである．

UDN[7]: UDN[5]の記述の属性値を最も具体化した記述．

UDN[8]: 間違えた記述名．

#### 5.1.2 結　　果

UDN[1]とUDN[2]，または，UDN[3]とUDN[4]との関係比較により，これらの表す対象は記述要素順に影響されず同一であり，探索の入口が記述名のある属性型に対応する半順序系列上ならば，概念の探索数も同じとなった．概念の探索数はいずれも17であった．

UDN[1]とUDN[3]，または，UDN[2]とUDN[4]との比較から，記述の省略に影響を受けず同じ対象を表し，概念の探索数も同じとなった．探索数は17であった．

UDN[5]とUDN[7]は，記述の抽象度が異なるが，同一の対象in_printを指した．共に最終軸概念がin_printであった．UDN[6]は，抽象度の高い記述のみで，最終軸概念が上位概念のfunctional_fileであり，より広い対象を指す結果になった（**図3**）．

探索経路は，探索入口によって異なる．入口がある役割の半順序の系列内ならば，探索数は変わらず，目的とする対象の持つどの属性型の半順序系列外の時は，最初の軸概念を発見するまでの探索が増えた．

多重継承性のため，同じ役割に対する二つの異なった属性値を継承することが起こり得るが（検証例ではprogram_fileがfile_typeに対しdataとprocedureの二つの値を継承している），どちらの系列を用いても問題はなく，3.1節(3)d）を裏付けた．

### 5.2 記述名表現と名前解決法の考察

#### 5.2.1 記述の同値性：表現の柔軟性と制約

記述名は表現に柔軟性があることから，異なる表現で同じ対象を示し得る条件や，このような表現に対する名前管理法の構成法が問題であった．4章の定理群を用いると，記述表現が異なる場合にも同じ対象を指すことが可能なことが分かる．異なる表現の記述名が同じ対象を表す条件は，定理6より，これらの記述名の要素の最終軸概念が一致することである．この条件を満たせば，記述量，記述要素順，記述の抽象度が異なっていても同じ目的の対象を指すことができる．例えば，記述量を省略した場合（5.1節のUDN[1]とUDN[3]等）や記述名の大部分の要素の記述クラスが抽象的な場合においても，目的とする対象を規定する属性型と属性値の組の記述が含まれれば，その対象を指すことができる．

#### 5.2.2 記述の抽象度の差と示される対象

記述名の要素の最終軸概念のSINet上の位置により指し示す対象の集合が異なることが分かる．記述の抽象度は属性型に対応する役割の半順序系列の前後で表される．このことから，抽象度の高い記述の場合，最終軸概念が半順序系列の前の方に位置し，SINetの上位に留まるので，最終軸概念に対する属性解集合の要素が多い．具体的記述の場合は，下位になるので属性解集合の要素が減り，対象は限定される（例えば，5.1節のUDN[5]，UDN[6]，UDN[7]の場合）．

#### 5.2.3 半順序性を利用した記述名解決法

探索経路は，4.2節のa)～c）を用いるので，定理4の逆木構造の形になる．役割の半順序系列に乗っている概念や後軸概念を決め，その下位概念のみを調べるので，全数探索ではない．下位概念ほど絞り込まれる．上位概念に関する多重継承性の関係は，b-1）で確認するので調べ忘れは生じない（図3）．もしも木構造ならば，多重継承性がなく，この処理は不要になる．

このアルゴリズムでは探索の入口は必ずしも根でなくてよい．入口が記述名のある属性型の半順序系列内ならば，軸概念が見つかり，後軸概念が更新される．目的の対象が存在しない幹から入った場合，後軸概念は初期値のままなので他の幹に移ることができる．最初の軸概念が見つかる前と後とで，同じアルゴリズムで処理でき，区別する必要のないことも分かった．

探索の入口が記述名のある属性型に対応する半順序系列上のどこかならば，概念の探索数が同じになる理由は，探索をSINetの構造に沿って，経路上の概念の役割を必ず調べる方法を採ったため，と考えられる．

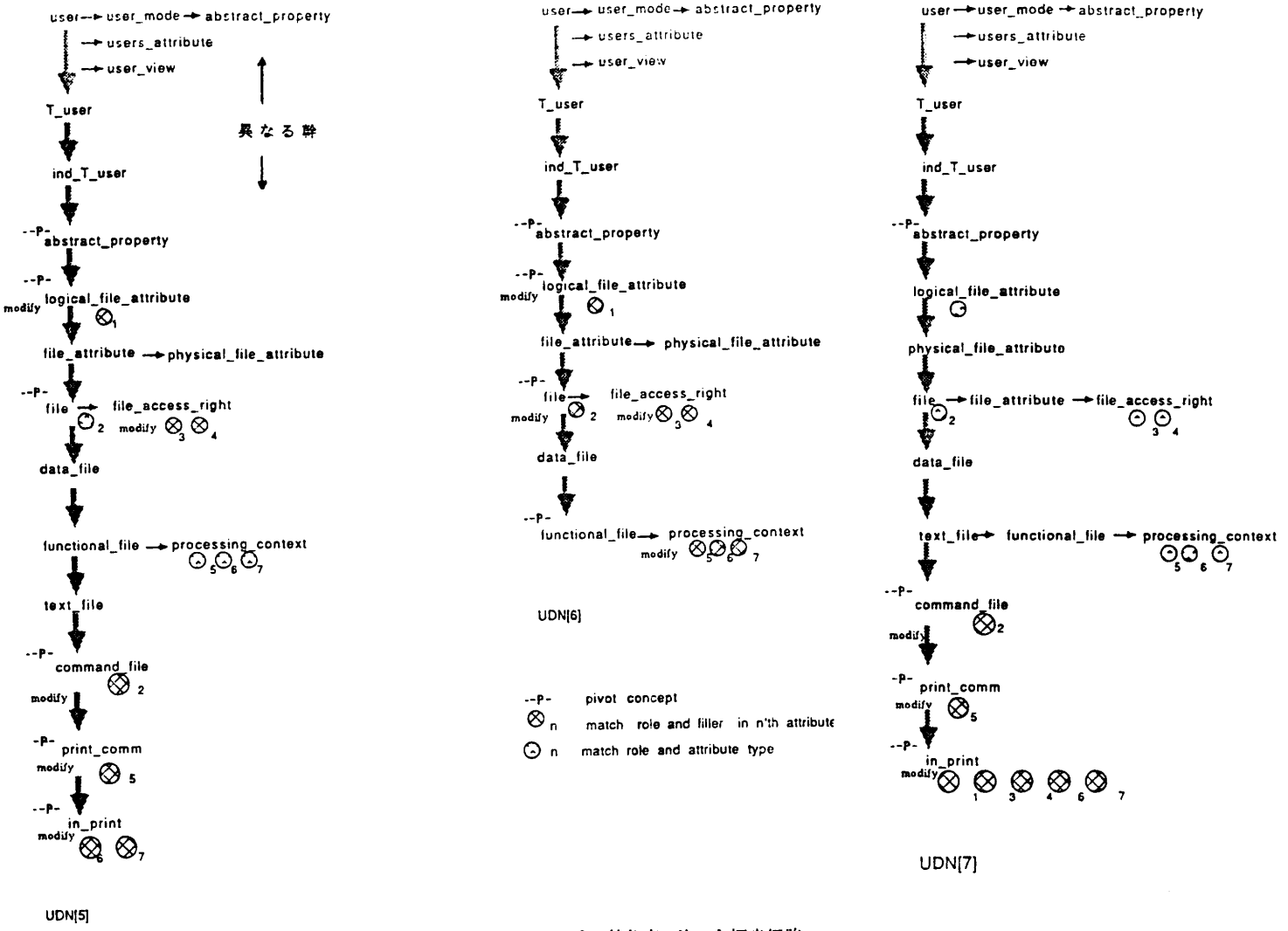


図 3 記述の抽象度の違いと探索経路
入口が異なる幹の場合.

Fig. 3 Searching paths on the condition of different abstraction-level description.
Entry point is incorrect trunk of SINet.

### 5.3 記述名に関する今後の課題

本論では，対象が固有に持つ性質記述を行う記述名のみを考察した．2.3 節でも述べたように，記述名には X. 500 等で考察されているような一時的に成り立つ属性の記述も必要となる．KL-ONE はこの両方を表現し得る構造を記述部と宣言部により持つことから，目標とする記述名表現とその名前解決法に役立つと考えられる．対象が一時的に成り立つ述語関係に関する属性記述については，KL-ONE の宣言部を利用することで実現可能と考えられるが，今後の課題である．述語関係は連想メモリ上でのマーカ伝搬[6] やコネクションにおけるウェーブ伝播等，実現法の研究がなされている．これらも考察のヒントとしたい．

3.2 節の CFU は，役割と補概念値の組が変化する場所のみの記述なので，記述名要素との比較により直ちに軸概念か否かが分かる．記述名解決法にこの考え方を利用し，軸概念の導入等に効果があったが，SINet 分割の規模等に対する評価は，残された課題である．

## 6. おわりに

計算機環境における諸資源の機能を規定するための記述に必要不可欠となる記述名を表現形式，抽象度，記述の省略(量)の面から考察した．記述名表現では，抽象度や記述量等，表現内容の詳しさを変えることにより，意味の度合や指す対象の集合を変化させたり，異なる表現形態で同じ対象を指し得ることが，記述要素の持つ半順序性を通して計数的に理解された．その結果，このような柔軟性がある記述に対する名前解決法を，記述要素の半順序関係が表現可能な意味ネットワークを利用して，新しく提案することができた．

このような記述名は，今後，ますます重要になると予想される，使いやすい資源管理辞書を構成する上で，鍵となる要素である．本論文では，この目的に至る第一段階の基本システム構成とその基本アルゴリズムを示した．今後の課題は，これを基に，より柔軟なシステム構成を検討することである．

## 参考文献


1) Birrell, A. D., Levin, R., Needham, R. M. and Schroeder, M. D.: Grapevine: An Experience in Distributed Computing, *Comm. ACM*, Vol. 25, No. 4, pp. 260-274 (1982).
2) Brachman, R. J.: An Overview of the KL-ONE Knowledge Representation System, *Cognitive Science*, Vol. 9, No. 2, pp. 171-216 (1985).
3) Comer, D. E. and Peterson, L. L.: A Model of Name Resolution in Distributed Systems, *The 6th International Conference on Distributed Computing Systems*, pp. 523-530 (1986).
4) Debray, S. and Peterson, L. L.: Reasoning about Naming Systems, TR 87-16, p. 20, Department of Computer Science, University of Arizona (1987).
5) *Proceedings of the 1981 KL-ONE Workshop*, Fairchild Technical Report, No. 618, FLAIR Technical Report, No. 4 (1982).
6) 樋口哲也，古谷立美，半田剣一，楠本博之，国分明男：意味ネットワークマシン (IXM) プロトタイプの開発，情報処理学会研究会知識工学と人工知能，61-4 (1988).
7) Final Text of DIS 7498-3, Information Processing Systems—OSI Reference Model—Part 3: Naming and Addressing, ISO/IEC JTC1/SC 21 N 2872 (1988. 7. 20).
8) Information Processing Systems—Open Systems Interconnection—Directory—, ISO/IEC/DIS 9594 Part 1-Part 8 (1988).
9) Oppen, D. C. and Dalal, Y. K.: The Clearinghouse: A Decentralized Agent for Locating Named Objects in a Distributed Environment, *ACM Trans. Off. Inf. Syst.*, Vol. 1, No. 3, pp. 230-253 (1983).
10) Peterson, L. L.: A Yellow-Pages Service for a Local-Area Network, *Proceedings of the SIGCOMM '87 Workshop*, ACM, pp. 235-241 (1987).
11) Peterson, L. L.: The Profile Naming Service, *ACM Trans. Comput. Syst.*, Vol. 6, No. 4, pp. 341-364 (1988).
12) Shastri, L.: Default Reasoning in Semantic Networks: A Formalization of Recognition and Inheritance, *Artif. Intell.*, Vol. 39, No. 3, pp. 283-355 (1989).
13) Sollins, K.R.: Distributed Name Management, thesis of Doctor of Philosophy at Massachusetts Institute of Technology, p. 164 (1985).
14) Terry, D. B.: Structure-free Name Management for Evolving Distributed Environments, *The 6th International Conference on Distributed Computing Systems*, pp. 502-508 (1986).
15) Touretzky, D. S.: *The Mathematics of Inheritance Systems*, p. 220, Pitman, London (1986).
16) Watson, R. W.: Identifiers (naming) in Distributed Systems, Distributed Systems—Architecture and Implementation, *Lecture Notes in Computer Science*, Vol. 105, pp. 191-210, Springer-Verlag (1981).


(平成元年8月31日受付)
(平成2年6月 4 日採録)

**古宇田フミ子**（正会員）

1952 年生．1974 年お茶の水女子大学理学部数学科卒業．東京大学工学部電気工学科に勤務し，分散処理に従事している．電子情報通信学会会員．

**田中　英彦**（正会員）

昭和 18 年生．昭和 40 年東京大学工学部電子工学科卒業，昭和 45 年同大学院博士課程修了．工学博士．同年東京大学工学部講師，昭和 46 年助教授，昭和 62 年教授．昭和 53 年～54 年ニューヨーク市立大学客員教授，現在に至る．計算機アーキテクチャ，並列推論マシン，知識ベース，オブジェクト指向プログラミング，分散処理，CAD，自然言語処理，等の研究を行っている．‘計算機アーキテクチャ’，‘VLSI コンピュータⅠ，Ⅱ’，‘ソフトウエア指向アーキテクチャ’（いずれも共著），‘情報通信システム’著．電子情報通信学会，人工知能学会，日本ソフトウエア科学会，IEEE，ACM 各会員．