

ASN.1 データベースにおける

5 R-4

高速データ検索のためのデータ格納方式

小野 智弘 西山 智 堀内 浩規 小花 貞夫
国際電信電話株式会社

1. はじめに

抽象構文記法1(ASN.1)^[1]は、OSI応用層で扱うプロトコルやデータ要素の情報をコンピュータや端末の機種に依存することなく交換するための、「データ型の記法」と「符号化規則」を定めている。ディレクトリやネットワーク管理等のOSIの応用では、ASN.1で定義されたデータ型を持つ情報(ASN.1データ)をデータベースに格納する必要がある。筆者らは、現在、これらの応用の開発を容易にするために、汎用的に利用可能な、ASN.1データを効率的に扱えるデータベース(ASN.1データベースと呼ぶ)を開発している^{[2][3]}。本稿では、ASN.1データベースにおける高速データ検索のためのデータ格納方式について述べる。

2. ASN.1 データベースの概要

ASN.1データベース^[2]は、「ASN.1で定義された任意のデータ型を持つ情報を直接扱うデータベース」であり、以下の特徴を持つ。

- (1) ASN.1データの符号化/復号処理をインタプリタ形式で行なうことにより、データベース運用中にデータ型の変更が可能
- (2) データを通信路上で転送される符号化列で格納することにより、高速なデータ操作を実現

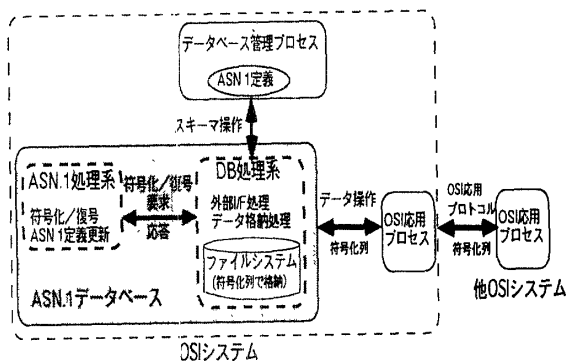


図 1: ASN.1 データベースのソフトウェア構成

ASN.1データベースは図1に示すように、OSI応用プロセスからローカルなデータ操作(検索、格納、更新、削除)を通してアクセスされる。また、ソフトウェア構成は、ASN.1データの符号化/復号を行なうASN.1処理系^[3]と、そのASN.1処理系を利用して外部インタ

フェース処理やデータ格納処理を行なうDB処理系から構成される。

3. データ格納に関する基本方針

高速なデータ検索を実現するためのデータ格納方式を設計するに当たり、以下の方針を立てた。

3.1 データ格納構造

ASN.1データベースでは、データ検索の高速化を図るために、BER(Basic Encoding Rules)^[1]形式で符号化されたデータを、符号化列が一意に定まるDER(Distinguished Encoding Rules)符号化列へ変換し、格納する。この場合、物理的配置は以下の2通りが考えられる。

- (1) 同じ型を持つデータを一括して格納し、データ型の入れ子関係については構成要素のデータ間をポインタで参照し合う(図2(a))。
- (2) データ格納要求操作で指定されたデータ型の単位で一まとめに格納する(図2(b))。

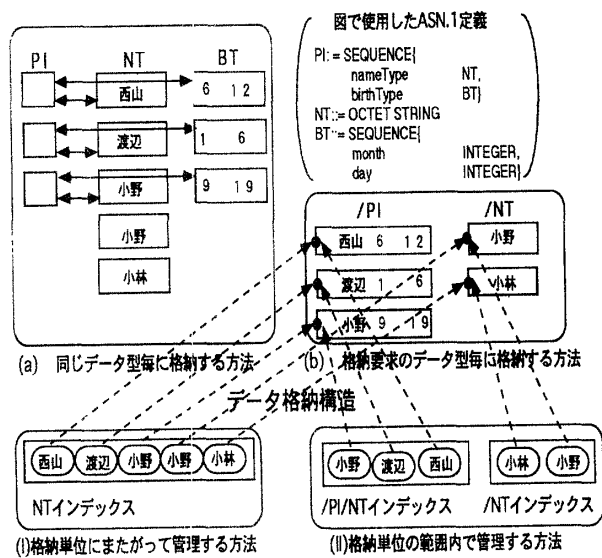
前者の場合、格納要求のあったデータ型から他の型をまたがって特定の型のデータを検索をするという要求には対応しやすいが、格納要求のあったデータ型全体を読み出す際にポインタをたどる必要があるため、検索時の処理負荷が重い。また、OSI応用の検索要求では操作対象が格納要求のデータ型と同じ単位となることが多いため、後者(図2(b))の方式で格納することとする。

3.2 インデックス機構

OSI応用での検索要求では格納されたデータ型の範囲内で条件が設定されることが多いため、インデックスは、格納されたデータ型単位をまたがって管理する(図2(I))のではなく、格納されたデータ型の範囲内で管理する(図2(II))。また、データ型の入れ子構造で階層的に定義されるASN.1データの特徴を考慮し、高速化のために以下の2つの機構を持たせる。

1) ハッシュ関数の導入 データ量の大きくなる、ASN.1構造型(SEQUENCE、SET、SEQUENCE OF、SET OF)のデータに対するインデックスの格納領域、比較時のコストを減少させるために、符号化列にハッシュ関数を施したハッシュ値でインデックス値を保持し、比較を行なう機能。

2) 間接的なインデックスの利用 検索条件で指定された型に直接インデックスが存在しない場合でも全数探索を回避するために、対象となるデータ型がASN.1構



インデックス形式

図 2: データ格納構造とインデックス形式

造形で、その構成要素のいずれかの型にインデックスが存在すれば、それを間接的に利用して探索する格納データの数を削減する機能。例えば図2の/PIに対する条件の検索の場合、/PI/NTにインデックスが存在すれば、それを間接的に利用する。

4. データ格納方式の設計

4.1 データ格納構造の実現

データの格納構造については、格納要求のあったデータ型単位毎(図2では/PI、/NT)にB木を用意し、データを格納する。ASN.1では各データ型のデータの長さは不定である。今回はデータを固定長でファイルシステムに格納することとし、データが固定長を超える場合は分割して格納する。B木ではデータ識別子(OID)と分割したデータを連結するための1バイトからなるヘッダを「キー」とし、DER符号化列形式の格納データを「値」とする。

格納したデータに含まれるデータ要素の位置(インデックスを付与するため等)は、各階層のデータ型の並びからなる「パス」で管理する。パスは、

/Level(1)/Level(2)/../Level(k)/../Level(n)

と記述する。Level(k)は該当するデータ型のタグ、識別子または出現位置で表される。特にLevel(1)は格納したデータ型単位を示し、データ内での位置は、後続するLevel(k)の並びで管理する。図2では、/PI/NTはPT型単位で格納されたデータ内のNTを指し、/NTはNT型単位で格納されたデータを指す。

4.2 インデックス機構の実現

インデックスはそれぞれをB木で格納し、図2に示すようにパスで管理する。インデックスのB木では、格

納データ内のパスで特定されるデータ型の値を「キー」とし、格納データのOIDを「値」とする。高速化のための、ハッシュ関数の導入と間接的インデックスの利用は以下により実現する。

4.2.1 ハッシュ関数の導入による検索の高速化

ASN.1 構造形に対するインデックス値は、データ(DER符号化列)のタグ、長さも含めた該当部分を切り出し、ハッシュ関数を施したハッシュ値で格納する。

比較演算子は一致か不一致が使用可能で、以下の手順で検索を行なう

- (1) 操作要求に含まれる符号化条件値(BER形式:C.b)をDER符号化列(C.d)へ変換する。
- (2) C.dのハッシュ値(C.h)を算出する。
- (3) C.hと、インデックス値(I.h)を比較し、合致したデータのOIDの集合を求める。
- (4) ハッシュ値同士が一致しても実際のデータは一致するとは限らないため、求めたOID集合に対応する実際の格納データの該当部分(D.d)を用いて比較し判定する。

なお、ASN.1単純形は一致、不一致のみでなく大小や部分一致等の比較演算子を用いるため、これらに対してはハッシュ値を用いず、値自身を復号してインデックス値として格納し検索する。

4.2.2 間接的なインデックスを用いた検索の高速化

ASN.1データベースでは、同じデータ型に対するインデックスをパス毎に別々に管理しており、さらにパスから要素内の位置を特定できるため、全てのASN.1構造形に対して以下の手順で間接的にインデックスを利用できる。検索手順を以下に示す。

- (1) 操作要求に含まれる条件の型から、間接的に利用できるインデックスの型を抽出する。
- (2) 間接インデックス毎に、DER化済みの条件値の該当部分を用いて合致するOID集合を求める。
- (3) 求めたOID集合のANDをとる。
- (4) 上記(3)の結果のOID集合から4.2.1節(4)と同様の方法で比較し判定する。

5. おわりに

本稿では、ASN.1データベースにおいて高速データ検索を達成するためのデータ格納方式を述べた。ここでは、格納要求単位でのデータ格納、インデックス機構におけるハッシュ関数の導入と間接的インデックスの利用が重要である。最後に日頃御指導頂く国際電信電話(株)研究所 村上所長に感謝します。

参考文献

- [1] ITU-T 勧告 X680 シリーズ"ASN.1"(1992)
- [2] 西山 他;"ASN.1 データベースの実現方式に関する一考察", 第49回情報全大 4W-11,(1994)
- [3] 小野 他;"ASN.1 データベースのためのASN.1符号化/復号処理系の設計と実装",DPS研究会資料 79-14,(1996)