データ通信とデータ格納のための ユニバーサルASN.1トランスレータ

小野智弘† 西山智† 堀内浩規†
小花貞夫† 鈴木健二†

OSIディレクトリやOSI管理等の応用では，ASN.1で定義されたデータ型を持つ情報を永続的に格納し，しかも，運用中にデータ型の定義を変更する必要のある場合が多い．これらの応用プログラムを効率的に開発するためには，データ通信処理とデータ格納処理の双方で共通に使用可能なASN.1符号化/復号ツールが必要となるが，既存のツールはデータ格納処理には機能面，性能面で十分に対応できない．本論文では，データ通信処理とデータ格納処理の双方で共通に使用可能なASN.1符号化/復号ツールであるユニバーサルASN.1トランスレータを提案する．本トランスレータは運用中にASN.1によるデータ型定義の動的な変更が可能であり，処理の高速化のために，1) インタフェースの統一によるコンパイラとインタプリタの効率的な併用，2) 部分符号化/復号機能の提供，3) 一意符号化機能の提供の3つの機構を持つ．

Universal ASN.1 Translator for Data Communication and Data Storage

Chihiro Ono,† Satoshi Nishiyama,† Hiroki Horiuchi,†
Sadao Obana† and Kenji Suzuki†

Many applications such as OSI Directory and OSI Management store data specified by Abstract Syntax Notation One (ASN.1), and some of them need to dynamically change their abstract syntax. In order to realize efficient development of such data storing applications, we need a powerful ASN.1 encoding and decoding tools applicable to both data communication and data storage. Unfortunately, existing ASN.1 compilers and ASN.1 interpreters are not designed for data storage. This paper proposes a powerful ASN.1 encoding and decoding tool, that is, a universal ASN.1 translator applicable to both data communication and data storage. In order to achieve high performance, the translator provides the following three mechanisms: 1) efficient hybrid use of a compiler and an interpreter by unifying interfaces such as routines, local representation of abstract syntax and value; 2) useful partial encoding/decoding functions; and 3) a unique encoding function.

1. はじめに

抽象構文記法1（ASN.1）[1),2)]は，開放型システム間相互接続（OSI）の応用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．また，ASN.1はOSI応用のみでなく，ネットワーク管理やディレクトリ等のインターネット応用でも用いられている．これらのOSI応用やインターネット応用の多くは，ASN.1で定義されたデータ型を持つ情報（以下，ASN.1データと呼ぶ）を永続的に格納する．このような応用には，OSIディレクトリ[3)]のディレクトリ情報ベース（DIB），OSI管理[4)]の管理情報ベース（MIB），メッセージ通信処理システム（MHS）[5)]のメッセージストア（MS），ライトウエイトディレクトリアクセスプロトコル（LDAP）[6)]に基づくDIB，簡易ネットワーク管理プロトコル（SNMP）[7)]に基づくMIB等がある．特にDIBやMIB等の応用は，オブジェクトクラスや属性型の追加・変更のために，運用中にASN.1データの型定義（以下，ASN.1定義と呼ぶ）を動的に変更する必要のある場合が多い．

このため，データを格納する応用（以下，データ格納応用と呼ぶ）を作成する際には，応用プログラムのデータ通信処理部分のASN.1符号化/復号処理と，動的にASN.1定義を変更する必要のあるデータ格納処

† 国際電信電話株式会社研究所
KDD R&D Laboratories

理部分のASN.1符号化/復号処理の双方で共通に使用可能で，しかも，部分符号化/復号機能や一意符号化機能等のデータ格納処理に特有な機能を有するASN.1符号化/復号ツールが必要となる．しかしながら，既存のASN.1インタプリタ[8)~12)]で，そのような機能をすべて備えるものはない．

本論文では，統一された応用プログラムインタフェース（API）とデータ格納処理に必要な機能を持ち，データ通信処理とデータ格納処理の双方で共通に使用可能なASN.1符号化/復号ツールである，ユニバーサルASN.1トランスレータを提案する．以降，2章でトランスレータの要求条件をあげ，3章でトランレータの設計概要を述べる．また，4章でトランスレータの使用法を例示し，5章でトランスレータの評価を述べる．

2. ユニバーサルASN.1トランスレータの提案

データ格納応用のデータ格納処理部分を作成する際には，**図1**に示すように，以下の2つの方針がしばしばとられる．

方針1 動的に変更されうるASN.1定義に対してインタプリタを，また，変更されないASN.1定義に対してコンパイラを使用：
ASN.1定義を動的に変更する必要のある応用では，ASN.1データの符号化/復号処理のためにインタプリタを使用する必要がある．さらに，高速化の観点から，変更のないASN.1定義に対しては，コンパイラを使用する必要がある．

方針2 市販のDBMSを使用：
開発の効率化のために，データ格納応用はデータ格納処理部分のベースに市販のデータベース管理システム（DBMS）を使用する．これは市販のDBMSが，応用開発者が容易に実現することの困難な一貫性制御，安全性制御，信頼性制御等の豊富なデータ管理機構を提供するためである．この場合，以下の手法を用いて処理の高速化を図る．

手法1 ASN.1にかかわる処理をDBMSの外部で実施：
ASN.1ではSEQUENCE，SET等の構造形が定義されているが，このうち，CHOICEやSET等のASN.1定義からDBMSスキーマのデータ構造への変換は非効率的であり，変換コストも大きい．さらに，DBMSを用いてこれらの構造を持つASN.1データに対する高速インデックス機構を実現することも困難である．このため，ASN.1にかかわるすべての処理をDBMSの外部で行う．

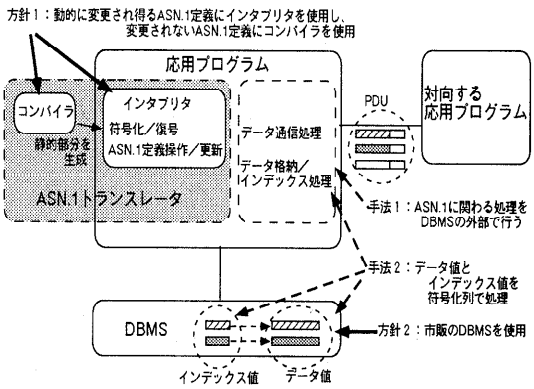


図1 データ格納処理部分の方針
Fig. 1 Strategies for data storage.

手法2 ASN.1データとインデックス値を符号化列で格納・操作：
データ通信処理とデータ格納処理で同一の符号化したオクテット列（符号化列）を使用することにより，データ格納・操作にともなう符号化/復号処理を減少させる．また，インデックス値を符号化列で格納し，データ格納処理における条件値とインデックス値との比較を符号化列どうしで行う．

このことから，データ通信処理のためのASN.1符号化/復号処理に加えて，上記の2つの方針を支援するASN.1符号化/復号ツールが必要となる．既存のASN.1インタプリタ[8)~12)]は，データ通信処理にかかわるASN.1符号化/復号処理に適用可能であるが，ASN.1データの格納処理にかかわるASN.1符号化/復号処理を想定して設計されておらず，以下の理由でデータ格納処理には適用できない．まず，既存のコンパイラとインタプリタを同一の応用プログラムで使用した場合，コンパイラとインタプリタがそれぞれ提供する異なるデータ構造や関数等のAPIが混在するため，開発が繁雑・非効率となる．また，双方のAPIで定義されるデータ構造間の変換コストも大きい．さらに，既存のコンパイラやインタプリタの提供する単純な符号化/復号機能（全体符号化/復号）では，特定部分のみの復号等のデータ格納処理にともなう符号化列での処理の実行には十分ではない．

そこで筆者らは，ASN.1定義の動的な変更が可能で，データ通信処理とデータ格納処理の双方で共通に使用可能なユニバーサルASN.1トランスレータを提案する．本トランスレータでは，データ格納処理のための方針に対応した以下の3つの機構が必要となる．

(1) コンパイラ/インタプリタの効率的併用
[機構1]

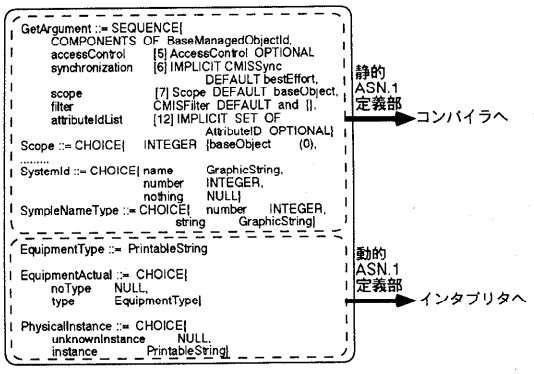


図 2 ASN.1 定義の例（OSI 管理）
Fig. 2 Example of abstract syntax (OSI Management).

(2) 多様な符号化/復号機能の提供
- 部分符号化/復号 [**機構 2**]
- 一意符号化 [**機構 3**]

2.1 コンパイラ/インタプリタの効率的併用

データ格納応用では，一般に，応用の ASN.1 定義を，運用中に動的に変更されうる部分（動的 ASN.1 定義）と，されない部分（静的 ASN.1 定義）に分けられる．プロトコルデータ単位（PDU）のようなデータ通信処理のための ASN.1 定義は，静的 ASN.1 定義となる．また，ASN.1 データを格納するための ASN.1 定義のうち，OSI 管理[13]で標準的に定義されている logID のような属性型も静的と見なすことができる．一方，ユーザが定義するオブジェクトクラスやその属性型は追加・変更が生じる可能性があるため，動的 ASN.1 定義となる．以下に，**図 2** で示す ASN.1 定義の例をあげる．

静的 ASN.1 定義の例 （図 2 参照）
- 応用 PDU のための ASN.1 定義
  （GetArgument 等）
- 運用中に変更されないデータの ASN.1 定義
  （SystemId 等）

動的 ASN.1 定義の例 （図 2 参照）
- 運用中に変更されうるデータの ASN.1 定義
  （EquipmentType 等）

ユニバーサル ASN.1 トランスレータは，静的 ASN.1 定義の符号化/復号にコンパイラの出力した関数を用い，動的 ASN.1 定義の符号化/復号にインタプリタを用いる．そこで，トランスレータでは，応用プログラムの開発者がコンパイラとインタプリタの 2 種類の異なったデータ構造を扱う繁雑さを避けるために，関数，ASN.1 定義内部表現，ASN.1 データ内部表現をコンパイラとインタプリタで共通化する必要がある．

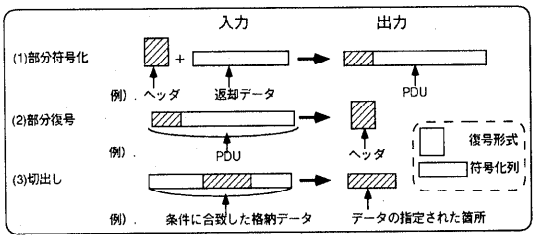


図 3 部分符号化/復号の例
Fig. 3 Example of partial encoding and decoding.

2.2 部分符号化/復号

ユニバーサル ASN.1 トランスレータでは，以下の符号化/復号機能が必要である．

(1) 部分符号化（**図 3** (1)）
符号化列の部分と復号形式の部分が混在するデータを符号化する機能である．本機能は，通信相手の応用プログラムへ送り返す符号化した PDU を作成する際に，復号形式のプロトコルヘッダ部分を，すでに符号化されている返却データに付加する場合等に使用する．

(2) 部分復号（図 3 (2)）
符号化列の特定箇所のみを復号する機能である．本機能は，符号化された PDU を受信した際に，プロトコル処理を行うためにプロトコルヘッダの部分のみを復号する場合等に使用する．

(3) 切出し（図 3 (3)）
符号化列の特定箇所のみを切り出す機能である．本機能は，通信相手の応用プログラムからの検索条件を満たすデータから，指定された箇所を切り出す場合等に使用する．

2.3 一意符号化

データ格納応用では，**図 4** に示すように，条件値とインデックス値を符号化列どうしで比較する．一般に，データ格納応用は，データ通信では基本符号化規則 (BER)[2]に基づく符号化を用いる．ところが，BER では SET 型における要素の出現順が任意である等，符号化形式が一意に定まらないため，BER 符号化列どうしで比較することはできない．そこで，BER 符号化列を一意に識別可能な符号化列に変換する機能が必要となる．

3. ユニバーサル ASN.1 トランスレータの設計

3.1 概 要

図 5 に，ユニバーサル ASN.1 トランスレータのソフトウエア構成，応用プログラム実行形式の生成過程，ならびに，生成された応用プログラムの構成を示す．

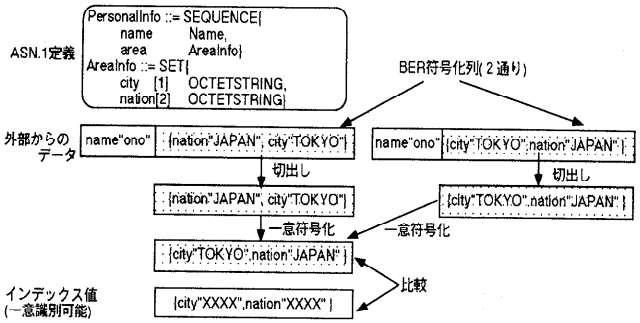


図4 インデックス機構
Fig. 4 Index mechanism.

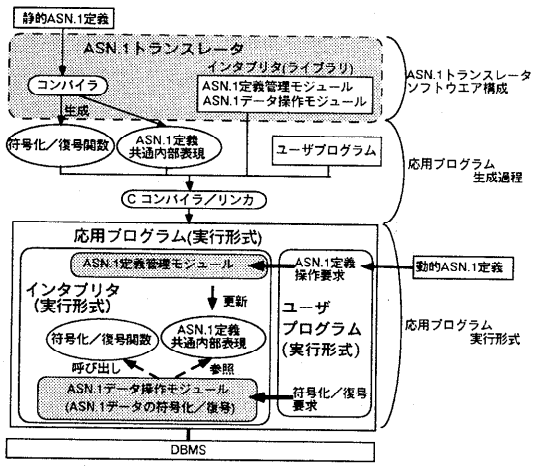


図5 ソフトウエア構成
Fig. 5 Software structure.

表1 符号化/復号関数
Table 1 Common routines for encoding and decoding functions.

関数	
whole_encode()	データ全体の BER または DER 符号化
whole_decode()	BER または DER 符号化列全体の復号
partial_encode()	BER または DER 符号化列部分と復号形式部分の混在したデータの BER または DER 符号化
partial_decode()	BER または DER 符号化列の指定された箇所のみの復号
trim_off()	BER または DER 符号化列の指定された箇所のみの切出し
ber_to_der()	BER 符号化列から DER 符号化列への変換

表2 ASN.1 定義操作用関数
Table 2 Common routines for updating abstract syntax.

関数	
select()	ASN.1 定義要素の取得
insert()	新しい ASN.1 定義要素の追加
update()	ASN.1 定義要素の更新
delete()	ASN.1 定義要素の削除

トランスレータは，コンパイラとインタプリタから構成される．

3.1.1 コンパイラ

コンパイラは静的 ASN.1 定義の型参照名ごとに，符号化/復号用の関数と，3.4 節で述べる ASN.1 定義共通内部表現の型参照名ノードを生成する．これらはインタプリタで使用される．

3.1.2 インタプリタ

インタプリタは ASN.1 定義管理モジュールとデータ操作モジュールから構成されるライブラリであり，実行形式では応用プログラム中に組み込まれる．

実行中に ASN.1 定義の更新要求を受け取ると，ASN.1 定義管理モジュールが構文を検査し，ASN.1 定義の要素を更新する．ASN.1 データの符号化/復号要求に対しては，ASN.1 データ操作モジュールが 3.4 節で述べる ASN.1 定義共通内部表現をたどりながら，ASN.1 データを符号化/復号する．型参照名が動的である間は動的型参照名ノードをたどり，符号化/復号処理を行う．静的型参照名ノードに到達した場合は，コンパイラの生成した符号化/復号関数を利用して符号化/復号を行う．

3.2 符号化/復号関数

ユニバーサル ASN.1 トランスレータは，**表1** に示すように，既存のコンパイラやインタプリタが提供する全体符号化/復号を行う関数に加え，2 章で述べた部分符号化/復号（機構 2）や，一意符号化（機構 3）に対応した関数を提供する．一意符号化には一意識別符号化規則（DER）[2]を使用し，BER 符号化列を DER 符号化列に変換する（DER 変換）．

3.3 ASN.1 定義操作用関数

ASN.1 定義管理モジュールは，動的 ASN.1 定義の管理を行い，**表2** に示す関数を提供する．

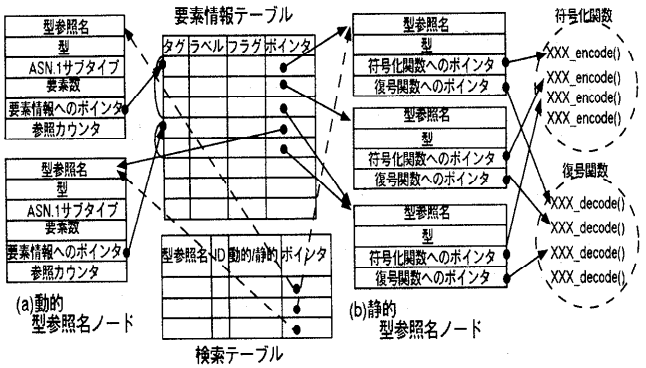


図 6 ASN.1 定義共通内部表現
Fig. 6 Common local representation of abstract syntax.

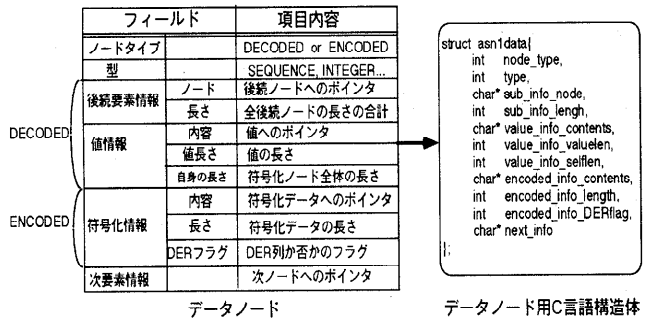


図 7 ASN.1 データ共通内部表現
Fig. 7 Common local representation of ASN.1 data value.

3.4 ASN.1 定義共通内部表現

コンパイラとインタプリタが提供する関数は，図 6 に示す共通の ASN.1 定義内部表現を使用する．

コンパイラが生成する ASN.1 定義共通内部表現は，静的 ASN.1 定義のみを含み，その後，インタプリタが動的 ASN.1 定義を追加・更新する．

図 6 に示すように，ASN.1 定義共通内部表現は，「静的型参照名ノード」と「動的型参照名ノード」の 2 種類の型参照名ノードを持つ．さらに，型参照名ノード間の関連を管理する「要素情報テーブル」，該当する型参照名ノードへのポインタを管理する「検索テーブル」を持つ．静的型参照名ノードは静的型参照名に対応し，型参照名（図 2 の GetArgument 等），型（図 2 の GetArgument に対応する SEQUENCE 等），対応する符号化/復号関数へのポインタ等の情報を含む．また，動的型参照名ノードは動的型参照名に対応し，型参照名（図 2 の EquipmentActual 等），型（図 2 の EquipmentActual に対応する SEQUENCE 等），ASN.1 サブタイプ（値の範囲等），包含する要素数（図 2 の EquipmentActual では 2 個），要素情報テーブルの個々の要素へのポインタ，当該ノードを参照している型参照名ノードの数等の情報を含む．

3.5 ASN.1 データ共通内部表現

部分符号化/復号（機構 2）を実現するために，トランスレータでは，図 7 に示す ASN.1 データ共通内部表現を使用する．

動的 ASN.1 定義の型参照名は，あらかじめ決定できないため，ASN.1 データ共通内部表現は型参照名に依存しない「データノード」から構成される．個々のデータノードは各データの型参照名に対応し，C 言語の構造体にマッピングされる．各データノードは符号化された部分と復号形式の部分を混在させるために，以下のフィールドを持つ．ノードタイプはノードの記述子で，データがすでに復号されていれば DECODED，符号化されていれば ENCODED の値をとる．後続要素情報は後続するデータノードへのポインタである．値情報は復号された値へのポインタで，ノードタイプが DECODED の場合に使用される．符号化情報は符号化データの値へのポインタで，ノードタイプが ENCODED の場合に使用される．次要素情報は同じ親データノードを持つ次のデータノードへのポインタである．

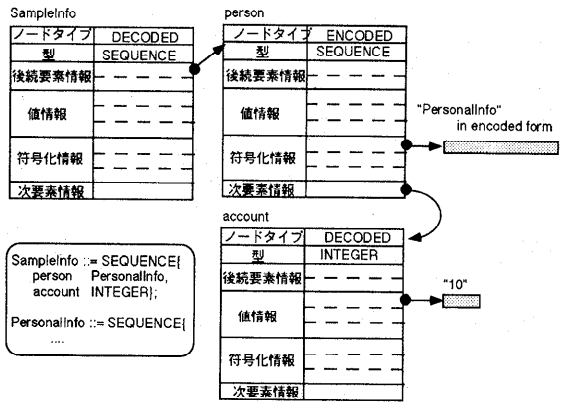


図 8 ASN.1 データ共通内部表現の例
Fig. 8 Example of common local representation of ASN.1 data value.

ASN.1 データ共通内部表現の例を図 8 に示す．SampleInfo データノードは，符号化されたデータノード person と復号されたデータノード account の 2 つの後続するデータノードを持つ．

4. ユニバーサル ASN.1 トランスレータの使用例

ユニバーサル ASN.1 トランスレータを使用した OSI 管理の応用における M-GET 操作の処理の例を図 9 に示す．応用プログラムは PDU を受信すると，まず，プロトコルヘッダを解析するために partial_decode() を使用する．続いて，フィルタリング時に条件値とインデックス値を符号化列どうしで比較するために，BER 符号化された条件値を DER 符号化列に変換する ber_to_der() を使用する．この DER 符号化された条件値を用いて検索条件に合致したデータを特定する．特定したデータを取得する際に，DBMS 上のデータから検索要求された箇所を切り出すために trim_off() を使用する．最後に，返却する PDU を生成する際に，符号化されている返却データに復号形式のプロトコルヘッダを加えて符号化するために partial_encode() を使用する．

5. 評価と考察

3 章で述べた設計に基づき，ユニバーサル ASN.1 トランスレータを C 言語で実装し，SUN SPARC Server 1000 上で評価実験を行った．ここでは，ユニバーサル ASN.1 トランスレータによる符号化/復号関数のみからなる応用プログラム実行形式を作成し，以下の性能評価を行った．評価では，図 10 に示す 5 段のデータ（総タグ数はバイト数の約 1/5 となる）を使用した．また，各々の結果は 1000 回の試行の平均をとった．

```
/* decoding protocol headers */
.....
partial_decode()
.....
/* filtering */
.....
ber_to_der()          /* converting condition value into DER form */
  (comprare condition values with index values in encoded form)
/* fetching the hit data one by one */
  for(i=0;i< NUMBER_OF_HIT_DATA;i++){
    fetch();
    .....
    trim_off()        /* trimming off the requested attributes */
    .....
  }
/* adding protocol headers to return value */
.....
partial_encode()
.....
```

図 9 ユニバーサル ASN.1 トランスレータ使用例（M-GET 操作）
Fig. 9 Example of using universal ASN.1 translator (M-GET Operation).

```
SampleInfo ::= SEQUENCE{
    data1     DataOctet,
    data2     Values1   }
Values1 ::= SEQUENCE{
    data3     DataOctet,
    data4     Values2   }
Values2 ::= SEQUENCE{
    data5     DataOctet,
    data6     DataOctet }
DataOctet ::= SEQUENCE OF OCTET STRING
    --5bytes(3 for contents, 1 for tag, 1 for length)
```

図 10 評価で使用するデータの ASN.1 定義
Fig. 10 Abstract syntax of data used in evaluation.

5.1 データ量とタグ数に対する応答時間の変化

図 11 にデータ量とタグ数に対する応答時間の変化を示す．ここでは，動的 ASN.1 定義に対する全体符号化/復号の場合について測定した．データはタグ数がデータ量の 1/5 の場合と 1/10 の場合の 2 種類を使用した．タグ数が 1/5 のデータでは，各データがタグ 1 byte，長さ 1 byte，コンテンツ 3 byte といった，最悪に近い応答時間を測定していることになる．

応答時間（T msec）は，ほぼ，データ量（X bytes）とタグ数（Y）に比例し，以下に示す近似式で表現できる．たとえば，タグを 200 個持つ 1 Kbyte のデータの復号の応答時間は 11.5 msec となる．

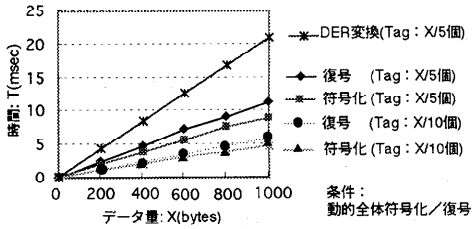


図 11 データ量とタグ数に対する応答時間の変化
Fig. 11 Time against data size and number of tags.

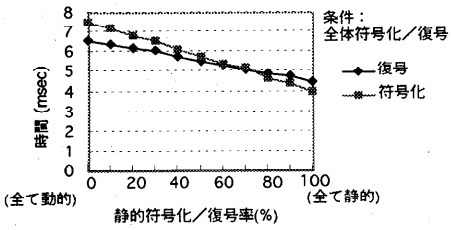


図 12 静的符号化/復号率の変化に対する応答時間の変化
Fig. 12 Response time against ratio of static encoding or decoding.

$$T \approx 5.0 \times 10^{-4}X + 5.5 \times 10^{-2}Y + 0.01$$

この式より，応答時間はデータ量よりもタグ数に大きく依存し，データ量が巨大でもタグ数が少なければ応答時間はそれほど劣化しないといえる．

5.2 コンパイラとインタプリタ併用の効果

図 12 は，ASN.1 定義中の静的定義の割合を示す静的符号化/復号率に対する応答時間の変化を示す．ここでは，550 byte のデータの全体符号化/復号の場合を測定した．

すべてが静的 ASN.1 定義である場合と，すべてが動的 ASN.1 定義である場合の符号化/復号の応答時間を比較すると，符号化の場合は静的の 7.5 msec に対して動的の 4 msec と 30%の減少となり，復号の場合は静的の 6.5 msec に対して動的の 4.5 msec と 45%の減少となった．したがって，共通の ASN.1 定義内部表現（3.4 節），ASN.1 データ内部表現（3.5 節）を使用して実現したコンパイラとインタプリタの併用が有効であるといえる．

DIB や MIB のようなデータ格納応用を作成する場合，多くのオブジェクトクラスや属性型が ITU-T 勧告や ISO 国際標準で標準化されており，これらが利用可能である．通常，応用プログラム開発者は応用に特化した部分の ASN.1 定義のみを加えればよい．したがって，使用する ASN.1 定義は，ほとんどが静的と見なせる標準で定義されたものであり，開発者が追加する動的 ASN.1 定義に該当する部分は少ないため，コンパイラとインタプリタとの併用の効果を大きく受けられる．

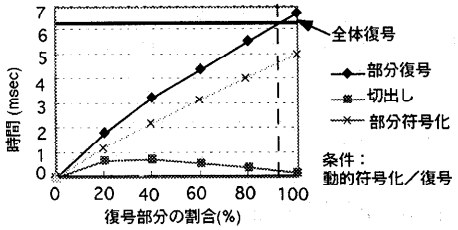


図 13 復号部分の比率に対する応答時間の変化
Fig. 13 Response time against ratio of decoded part.

5.3 部分符号化/復号の効果

図 13 では，復号部分の割合に対する応答時間の変化を示す．ここでは，動的 ASN.1 定義の 550 byte のデータを符号化/復号する場合を測定した．

復号部分の割合が 95%以下であれば，部分復号の方が全体復号に比べて高速となるため，ほとんどの場合に本機能が有効であるといえる．さらに，プロトコルヘッダの解析やインデックス値等の復号部分が小さい場合には本機能の効果がさらに大きくなる．

また，切出し関数については，任意の大きさのデータの切出し時間が 1 msec 以下であった．これは，本関数の内部処理が切出し箇所の特定のみであるためである．これにより，巨大なデータを扱う場合でも高速な切出しが可能となるといえる．

5.4 DER 変換の効果

DER 変換は，今回，BER 符号化されたデータをいったん復号し，DER で符号化する仕組みで実装した．したがって，応答時間は図 11 に示すように，両者の和となる．しかしながら，本関数が応用プログラム内での符号化条件値と符号化インデックス値との比較等（図 9 参照）の限定された回数しか使用されないことを考えると，応答時間の増加は無視できる程度であるといえる．

5.5 データ格納応用での全応答時間の考察

図 11～13 の結果を用いて，データ格納応用でのユニバーサル ASN.1 トランスレータの全応答時間を考察する．ここでは，データ格納応用において，2 章に述べた機構 2 と機構 3 を使用しない場合，**表 3** に示す代替方法を使用することとした．また，既存のコンパイラ[14]とインタプリタ[12]における，550 byte の全体符号化/復号の応答時間を**表 4** に示す．

以下に，OSI 管理の応用プログラムにおける M-GET 操作（処理の流れは 4 章と同様）の全応答時間を推定する．ここでは，データ量は 1 kbyte，条件数が 2 個，条件に合致するデータ数が 3 個と仮定する．

表 5 は，a) 機構 1，2，3 を用いたユニバーサル

表3 機構2，3の関数の代替方法
Table 3 Alternative solutions to routines of mechanism 2 and 3.

機構2，3の関数	代替方法
partial_encode()	符号化列の全体復号（whole_decode()）の後，データの必要部分を追加し，全体符号化する（whole_encode()）.
partial_decode()	符号化列の全体復号（whole_decode()）の後，指定部分を取り出す.
trim_off()	符号化列の全体復号（whole_decode()）の後，指定部分を取り出し，全体符号化する（whole_encode()）.
ber_to_der()	符号化列の全体復号（whole_decode()）の後，DER で全体符号化する（whole_encode()）.

表4 既存のツールの応答時間
Table 4 Response time for existing tools.

		符号化時間	復号時間
既存のインタプリタ		2.8 msec	9.1 msec
既存のコンパイラ		0.7 msec	1.8 msec
ユニバーサル ASN.1 トランスレータ	インタプリタ	7.5 msec	6.5 msec
ユニバーサル ASN.1 トランスレータ	コンパイラ	4.0 msec	4.5 msec

表5 データ格納応用における全応答時間の推定
Table 5 Estimation of total response time in data storing application.

	応答時間
a) ユニバーサル ASN.1 トランスレータ（機構1，2，3使用）	7.5 msec
b) ユニバーサル ASN.1 トランスレータ（機構2，3不使用）（代替方法（表3）を使用）	66.0 msec
c) 既存のコンパイラとインタプリタ（等価な関数での代替方法（表3）を使用）	34.5 msec

ASN.1 トランスレータ，b) 代替方法を用いたユニバーサル ASN.1 トランスレータ，c) 既存のコンパイラとインタプリタを用いた代替方法の3つの場合の推定応答時間を示す．a) と b) では，4章の例では，a) が b) より 8.8 倍高速であり，これは，機構2，3の効果によるものである．データ量，条件数，合致したデータ数が増加するにつれて，その差は拡大する．また，a) と c) では，表4に示すように個々の応答時間はインタプリタの復号時間を除いて，既存のコンパイラとインタプリタの方が短いが，4章に示す応用プログラムでの実際の利用形態を考慮に入れると，a) が c) より 4.6 倍高速となる．この結果は，機構1，2，3が効果的であることを示している．さらに c) の場合，実際には既存のコンパイラとインタプリタのデータ構造間の変換に処理時間を要するため，その差はさらに拡大する．

以上の評価結果より，本トランスレータは，既存の ASN.1 ツールに比べてデータ格納応用においては，処理の高速化に有効であることが分かる．

6. ま と め

本論文では，データ通信処理とデータ格納処理の双方で共通に使用可能な ASN.1 符号化/復号ツールである，ユニバーサル ASN.1 トランスレータを提案した．本トランスレータは，1) 動的な ASN.1 定義にインタプリタを使用し，静的な ASN.1 定義にコンパイラを使用する，2) 市販の DBMS を使用し，高速化のために ASN.1 データとインデックス値を符号化列で格納・操作する，という方針に基づくデータ格納応用のソフトウェア開発を支援する．具体的には，1) インタフェース統一によるコンパイラとインタプリタの効率的な併用，2) 部分符号化/復号機能の提供，3) 一意符号化機能の提供の3つの機構を具備している．評価を通じて，これらの機構がデータ格納応用の全応答時間の高速化に有効であることを示した．

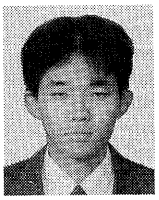
謝辞 日頃ご指導いただく，国際電信電話（株）研究所村上仁己所長に感謝いたします．

参 考 文 献

1) ITU-T Rec. X.680: Information Technology – Abstract Syntax Notation One (ASN.1): Specification of Basic Notation (1992).
2) ITU-T Rec. X.690: Information Technology – ASN.1 Encoding Rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER) (1992).
3) ITU-T Rec. X.500-X.521: Data Communication Networks Directory (1988).
4) ITU-T Rec. X.701: Systems Management Overview (1992).
5) ITU-T Rec. X.400-X.420: Data Communication Networks Message Handling Systems (1988).
6) RFC 1777: Lightweight Directory Access Protocol (1995).
7) RFC 1157: Simple Network Management Protocol (1990).
8) Canaschi, F. and merelli, E.: An Architecture for an ASN.1 encoder/decoder, *Computer Networks and ISDN Systems*, Vol.14, pp.297–303 (1987).
9) Gora, W. and Speyerer, R.: Automatic code generation for ASN.1-based protocols, *Proc. ENE'88*, pp.191–204 (1988).

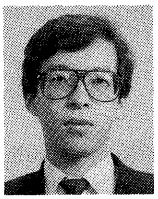
10) Ohara, Y., Suganuma, T. and Senda, S.: ASN.1 tools for semi-automatic implementation of OSI application layer protocols, *Proc. 2nd ISIIS*, pp.63–70 (1988).
11) Nakakawaji, T., et al.: Development and Evaluation of Apricot (Tools for Abstract Syntax Notation One), *Proc. 2nd ISIIS*, pp.55–62 (1988).
12) 長谷川亨，野村眞吾，堀内浩規：ASN.1支援ツールの開発—コンパイラおよびエディタ，情報処理学会マルチメディア通信と分散処理研究会，DPS39-4, pp.1–8 (1988).
13) ITU-T Rec. X.721: Structure Of Management Information: Definition of Management Information (1991).
14) Hasegawa, T., Nomura, S. and Kato, T.: Implementation and Evaluation of ASN.1 Compiler, *Journal of Information Processing*, Vol.15, No.2, pp.157–167 (1992).
15) 小野智弘，西山 智，堀内浩規，小花貞夫：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，情報処理学会マルチメディア通信と分散処理研究会，DPS79-14, pp.71–76 (1996).

(平成9年5月12日受付)
(平成9年9月10日採録)



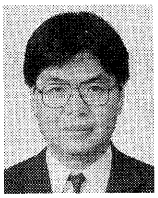
小野 智弘（正会員）

昭和43年生．平成4年慶応義塾大学電気工学科卒業．平成6年慶応義塾大学計算機科学専攻修士課程修了．同年国際電信電話（株）入社．以来，研究所にて，ネットワーク管理，通信支援データベース等の研究に従事．平成8年度情報処理学会全国大会大会奨励賞を受賞．電子情報通信学会会員．



西山 智（正会員）

昭和36年生．昭和59年東京大学工学部電気工学科卒業．同年国際電信電話（株）入社．平成3年米国テキサス大学オースチン校計算機科学科修士課程修了．現在，同社研究所ネットワーク管理グループ主任研究員．この間，データベース，ネットワークアーキテクチャ，国際ビデオテックス通信，分散処理技術およびプロダクションシステムの研究に従事．平成5年電子情報通信学会学術奨励賞受賞．電子情報通信学会会員．



堀内 浩規（正会員）

昭和35年生．昭和58年名古屋大学工学部電気工学科卒業．昭和60年同大学大学院情報工学専攻修士課程修了．同年国際電信電話（株）入社．現在，同社研究所ネットワーク管理グループ主任研究員．この間，ネットワークアーキテクチャ，OSIプロトコル実装方式，通信プロトコルの形式記述技法，ネットワーク管理の研究に従事．平成4年度電子情報通信学会学術奨励賞，平成8年度情報処理学会全国大会優秀賞を各受賞．電子情報通信学会会員．

小花 貞夫（正会員）

昭和28年生．昭和51年慶應義塾大学工学部電気工学科卒業．昭和53年同大学大学院修士課程修了．同年国際電信電話（株）入社．現在，同社研究所ネットワーク管理グループリーダ．工学博士．この間，パケット交換方式，ネットワークアーキテクチャ，OSIプロトコル実装，データベース，ビデオテックス，分散処理，ネットワーク管理の研究に従事．電子情報通信学会会員．

鈴木 健二（正会員）

昭和20年生．昭和44年早稲田大学理工学部電気通信科卒業．昭和44年から45年までオランダのフィリップス国際工科大学に招待留学．昭和51年早稲田大学大学院博士課程修了．工学博士．同年国際電信電話（株）入社．現在，同社研究所副所長．この間，磁気記録，パケット交換方式，ネットワークアーキテクチャ，高速・分散処理の研究に従事．平成4年度電子情報通信学会業績賞，平成7年度科学技術庁長官賞を各受賞．平成5年度より電気通信大学大学院情報システム学研究科客員教授．本会理事．電子情報通信学会，IEEE各会員．