2T－3　ASN.1高能率圧縮符号化規則(EPER)におけるサブタイプ対応の符号化規則の拡張

堀内浩規　小花貞夫　鈴木健二
国際電信電話株式会社　研究所

1. はじめに

ASN.1のための新たな符号化規則として、従来の基本符号化規則(BER)より符号化データ長を短くして効率的な通信を可能とする圧縮符号化規則(PER:Packed Encoding Rules)[1]があるが、ビットのシフト演算を頻繁に行って符号化/復号処理時間を増大させる等の問題点がある[2,3]。そこで、筆者等は先に上記問題点を解決する高能率圧縮符号化規則(EPER: Efficient Packed Encoding Rules)を新たに提案し、PERと比較して符号化処理時間を1.2~3.0倍、復号処理時間を1.2~5.7倍に向上させ、EPERの有効性を実証した[2,3]。しかしながら、ASN.1の抽象構文定義において、サブタイプにより整数型の値の範囲等に制約条件が付加される場合には、EPERの符号化/復号処理の一層の効率化が可能であることが想定される。本稿では、このための符号化規則の拡張を提案する。

2. EPERとASN.1サブタイプの概要

2.1 EPERの概要

PERでは、Unaligned転送構文使用時のシフト演算の頻発による処理速度の低下、及び、Aligned転送構文使用時のパディングによる符号化データ長の増大という問題点がある。これらの問題点を解決するため、ASN.1の各型に対応する符号化データをビット単位の要素とオクテット単位の要素を、それぞれ、ビット･フィールドとオクテット･フィールドに分離して設定する(図1)。また、抽象構文定義からビット･フィールドの長さが予め求められない場合には、その長さを示すオフセット･フィールドを付加する。EPERにおける各型の符号化規則を表1に示す。

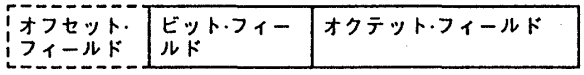

図1 符号化の構造

ビット･フィールドは、①Boolean型の場合、②Enumerated型で値の範囲が2から128の場合、③BitString型で値の長さが8ビットの倍数でない場合の剰余ビット、④Sequence型/Set型のPreamble、⑤Choice型のIndexで値の範囲が2から128の場合に使用する。

オフセット･フィールドは、①BitString型の場合、②Sequence型/Set型でオプショナルなビット･データの型を含む場合、③Choice型で選択肢中にビット･データの型を含む場合、④Sequence Of型/Set Of型で繰り返し要素の型がビット･データとなる場合に付加する。

2.2 ASN.1サブイタイプの概要

表1 EPERにおける各型の符号化規則

型	符号化規則
Boolean	BFに,1ビットで符号化. 値はTrue(0)/False(1).
Enumerated	列挙値を昇順に並べ換え,最小値を0として昇順に値を割当てる.割当てた値の範囲(R)が1の時,符号化しない.2≦R≦128の時BFに,R≧129の時OFに値-1を設定する.
Null	符号化しない.但し,Set型等の要素で,OPやDFの時,BFのPreambleに存在の有無示す.
Integer	-32≦値≦31の時 1オクテットで,-2^{35}≦値≦$2^{35}-1$の時2~5オクテットで,その他は6オクテット以上でOFに符号化する.
Real	LIと値からなり,OFに設定する.値はBERと同一の符号化規則.
Oct.,/Chr. String	LIと値からなり,OFに設定する.
BitStr.	値のビット長を示すLIと値からなる.LIはOFに設定する.値は,長さが8ビットの倍数でない剰余ビットをBFに設定し,その他はOFに設定する.
Object Identifier	LIと値からなり,OFに符号化する.値はBERと同一の符号化規則.
Any	LIと値からなり,OFに符号化する.
Sequence /Set	Preambleと1個以上の構成要素からなる.PreambleはOPとDFの要素の存在を表し,要素有(1)/無(0)の値をBFに設定する.OPやDFが無い場合には,Preambleは符号化しない.構成要素は,各型に従って,OF,BFに符号化する.
Choice	Indexと選択要素からなる.Indexは選択された要素を識別するための番号を割り当て,選択肢数が1個の時は符号化せず,2~128個の時はBFに,129個以上の時はOFに符号化する.選択成要素は,各型に従った符号化規則.
Sequence Of /Set Of	要素の個数を示すNumber部と要素の型を符号化したComponent部からなる。Number部はOFに符号化され,Component部は使用する型に従って、OFまたはBFに符号化する.Number部はLIと同一の符号化規則.

注) BF:ビット･フィールド,OF:オクテット･フィールド,
OP:オプショナル,DF:デフォルト,LI:値の長さ.

ASN.1における Integer、Octet String (Character Stringを含む)、Bit String、Sequence OfおよびSet Of型では、サブイタイプを使用して、その型が取りうる値の範囲、長さ、繰り返し要素数に関する制約を抽象構文上で記述できる。例えば、"INTEGER(0..128)"ではInteger型の値の範囲が0から128、"OCTET STRING(16)"ではOctet String型の値の長さが固定長で16オクテット、"SEQUENCE SIZE (20) OF"ではSequence Of型の繰り返し要素数が固定数の20であることを示す。

3. サブイタイプ対応の符号化規則の拡張提案

3.1 基本方針

(1) サブタイプによる制約条件が単一の値しかとらない場合(例えば、INTEGER(32))には、使用する値は自明なので符号化を行なわない。

Extension of Encoding Rules for Subtypes in EPER (Efficient Packed Encoding Rules for ASN.1)
Hiroki HORIUCHI, Sadao OBANA and Kenji SUZUKI, KDD R & D Laboratories

(2) サブタイプによる制約条件が値の範囲を限定する場合(例えば、INTEGER(0..128))には、それを活用した符号化を行なう。

3.2 EPERへの符号化規則の追加

EPERへの符号化規則の追加を表2に示す。

(1) Integer型に対する追加

単一の値の場合には符号化は行なわない。また、値の範囲が64K以下の場合には、LIは符号化せず、値が表現可能な最小現の符号化データ長を確保し、値のみをビット･フィールドまたはオクテット･フィールドに符号化する。

(2) Octet String, Bit String型に対する追加

値の長さが固定長の場合はLIの符号化を行なわない。また、長さの範囲が64K以下の場合はLIとして長さの下限値との差分を符号化する。

(3) Sequence Of, Set Of型に対する追加

要素の個数が固定の場合にはNumber部は符号化を行なわない。また、要素の個数が64K未満の場合にはNumber部は要素数の下限値との差分を符号化する。

表2 サブタイプを使用した符号化規則の追加

型	符号化規則
Integer (値の範囲(R)に制約有)	① 単一値(R = 1):符号化しない,② 上限値または下限値のみ:制約無しの場合と同一の規則.但し,下限値との差分を符号化.③ 値の範囲が64K以下:LIは符号化せず,値のみを符号化.値は,下限値との差分を符号化し,2 ≤ R ≤ 128の時はBFに最小ビット数で符号化,R ≥ 129の時はOFに符号化.④ 値の範囲が64Kを越える:②と同一の規則.
Octtet,/ Character String (値の長さに制約有)	①長さが固定長:LIは符号化しない.②長さの上限値が64K未満:LIは下限値との差分を符号化.符号化方法はInteger型の③と同一.③長さの上限値が64Kオクテット以上:制約無しの場合と同一の規則.
BitStr.(値の長さに制約有)	Octet String型と同一の規則.
Sequence Of /Set Of (繰り返し要素数に制約有)	①要素数が固定:Number部は符号化しない.②要素数が64K未満:Number部は,要素数の下限値との差分を符号化.符号化方法はInteger型の③の方法と同一.③要素の個数がが64K以上:制約を与えられない場合のSequence Of, Set Of型と同一の規則.

注) BF:ビット･フィールド,OF:オクテット･フィールド,

これにより、ビット･フィールドは、①Integer型の値の範囲、②Octet StringやBitString型のLIで長さの範囲、③Sequence Of, Set Of型のNumber部で繰り返し要素数、それぞれが、2から128の範囲に限定された場合にも使用する。また、オフセット･フィールドの使用は、上記のビット･フィールドを使用する要素が、①Choice型の選択肢の要素、②Sequence型/Set型でオプショナルな要素、③Sequence Of型/Set Of型で繰り返し要素、となる場合にも付加する。

4. 評価実験

3章の符号化規則の拡張に従って、作成済みのEPERのコンパイラ[3]を拡張し、抽象構文から生成した符号化/復号処理プログラムを使用して、符号化/復号処理時間および符号化データ長を測定した。抽象構文としては、図2に示すサブタイプが無いもの(Eg1)と有るもの(Eg2)を使用した。測定計算機としてはNEWS 3860を使用した。表3の実験結果より、追加したサブタイプを使用した符号化規則の符号化処理時間と復号処理時間は、サブタイプを使用しない場合と比較して、それぞれ、1.7倍、3.0倍に高速となった。また、符号化データ長は0.8倍に短くなった。

```
Eg1 :: = SEQUENCE OF SEQUENCE {
    first INTEGER, second IA5String, third INTEGER }
Eg2 :: = SEQUENCE SIZE (20) OF SEQUENCE {
    first INTEGER (0..128), second IA5String (16),
    third INTEGER (32) }
```

図2 評価実験に使用した抽象構文定義

表3 評価実験結果

抽象構文	符号化時間 (msec)	復号時間 (msec)	データ長 (oct.)
Eg1 (サブタイプ無)	0.12	0.06	421
Eg2 (サブタイプ有)	0.07	0.02	340

注) Eg1のSequence ofの要素は 20個.

5. 考察

(1) 符号化/復号処理時間と符号化データ長

サブタイプの使用による符号化/復号処理時間と符号化データ長の効果は、サブタイプにより定義された制約条件に依存するが、特に、値や長さが単一であったり、値の範囲が狭くかつ下限値が大きい場合には、相対値を符号化するため、その効果は大きい。また、サブタイプ定義における値の範囲が広くかつ下限値が小さく、実際の値が下限値近くにかたよって存在するような特殊な場合(例えば、サブタイプ定義の値の範囲が0から32Kの場合で、実際の値が0~31の時の符号化データ長は、従来のEPERでは1オクテット、本提案では2オクテットとなる)を除き、ほとんどの場合に効果があると考えられる。上記の特殊な場合には、抽象構文定義中の該当する型にサブタイプを使用しないことが望まれる。

(2) 符号化/復号プログラムの処理

サブタイプを使用した符号化規則の追加により、符号化規則及びコンパイラは、従来のEPERの場合より複雑となる。しかしながら、サブタイプを使用するか否かはコンパイラが抽象構文定義から判別するので、生成される符号化/復号処理プログラムでは、その判定を行なう必要は無く、それによる処理速度の低下は生じない。

6. おわりに

本論文では、高能率圧縮符号化規則(EPER)において、ASN.1サブタイプを用いて型に制約条件が付加された場合に、その制約条件を活用した符号化規則とすることで、符号化データ長や符号化/復号処理時間が向上することを示した。

最後に、日頃御指導をいただくKDD研究所 浦野所長、眞家次長に感謝致します。

参考文献

[1] ISO/IEC CD 8825-2 "Specification of ASN.1 Encoding Rules - Part2: Packed Encoding Rules", Dec. 1992.
[2] 堀内,小花 鈴木, "ASN.1における圧縮符号化規則(EPER)の改善提案",情処第48回全大 2D-4, Mar. 1994
[3] 堀内,小花 鈴木, "ASN.1のための高能率圧縮符号化規則(EPER)の提案と評価",情処研究報告 DSP66-25, July 1994.