OSI管理における管理情報ベース（MIB）のアクセス制御アルゴリズム

吉原貴仁† 堀内浩規†
杉山敬三† 小花貞夫†

OSI管理に基づくネットワーク管理システムの実際の運用では，管理情報ベース（MIB）のアクセス制御が重要である．MIBのアクセス制御では，アクセスの拒否と許可の単位（粒度）として，粗い順に（1）管理操作単位，(2）管理オブジェクトインスタンス（MOインスタンス）単位，および(3）属性単位の3つがある．これまでに粒度を管理操作単位とするアクセス制御アルゴリズムの報告があるが，粒度が管理操作単位の場合，アクセスが許可されているにもかかわらず，実際にはアクセスできないMOインスタンスが生じる．このため，個々のMOインスタンスに管理操作を発行する必要があり，結果として管理操作の発行回数の増加を招き，ネットワーク管理システム全体の効率が低下する問題がある．本論文では，この問題を解決するため，粒度をより細かいMOインスタンス単位とするMIBのアクセス制御を，粒度を管理操作単位とする従来のアルゴリズムと同程度の計算量（$O(n)$）で実現するアルゴリズムを提案する（nは名前木のMOインスタンスの個数）．

Algorithm for Access Control of Management Information Base (MIB) in OSI Management

KIYOHITO YOSHIHARA,† HIROKI HORIUCHI,† KEIZO SUGIYAMA† and SADAO OBANA†

Access control of Management Information Base (MIB) is significant in network management based on Open Systems Interconnection (OSI). There are three kinds of granularity for access control, each of which is (1) "management operation", (2) "MO instance", and (3) "attribute". When the granularity is "management operation", there might exist an MO instance to which cannot actually be accessed even if it is granted to access. It degrades the performance of a network management system, since an operation to each MO instance has to be issued, and this results in a plenty of operations. This paper proposes an algorithm for access control of MIB, which has the finer granularity of "MO instance" than that of the existing algorithm and has the time complexity of $O(n)$, almost the same order as that of the existing algorithm, where n represents the number of MO instances in a naming tree.

1. はじめに

OSI管理[1]に基づくネットワーク管理システムの実際の運用では，管理情報ベース（MIB）のアクセス制御が重要である[2)~4)]．たとえば，ネットワークの運用者には，その管理目的や権限に応じてMIBを提供する必要がある．また，CNM（Customer Network Management）[5),6)]を始めとする公衆網や私設網等の異なる管理ドメイン間の接続では，その重要性は顕著となる．

そこで，国際標準ITU-T Recommendation X.741 | ISO/IEC10164-9[7)]は，MIBのアクセス制御用の管理オブジェクト（MO）クラスとこれらを使ったアクセス制御の枠組みを規定している．MIBのアクセス制御では，アクセスの拒否と許可の単位（粒度）として，粒度の粗い順に（1）管理操作[8)]で対象となるすべての管理オブジェクトインスタンス（MOインスタンス）の単位（以下，管理操作単位と呼ぶ），(2）管理操作で対象となる個々のMOインスタンスの単位（以下，MOインスタンス単位と呼ぶ），および（3）属性単位がある．粒度が細かいほど，より小さい管理情報を単位とするきめの細かいアクセス制御が可能になる．

従来，管理操作単位を粒度とするアクセス制御アルゴリズムやその実装報告がある[9),10)]．しかしながら，粒度が管理操作単位の場合，アクセスが許可されてい

† 国際電信電話株式会社研究所
KDD R&D Laboratories

るにもかかわらず，実際にはアクセスできないMOインスタンスが生じる．このため，複数のMOインスタンスを指定するスコープを使わず，個々のMOインスタンスへ管理操作を発行する必要があり，結果として，管理操作の発行回数の増加を招き，ネットワーク管理システム全体の効率が低下するという問題がある．

本論文では，上記の問題を解決するため，粒度をより細かいMOインスタンス単位としながら，処理効率を低下させないMIBのアクセス制御アルゴリズムを提案する．以下，2章で国際標準によるMIBのアクセス制御の枠組み，3章で粒度を管理操作単位とする従来アルゴリズムの問題点を述べる．4章では，粒度をMOインスタンス単位とするアルゴリズムの提案を行い，5章で提案アルゴリズムの理論的な計算量および処理速度等の観点からの評価を行う．

2. 国際標準によるMIBのアクセス制御の枠組み

国際標準ITU-T Recommendation X.741 | ISO/IEC 10164-9は，以下に示すMIBのアクセス制御用のMOクラスとこれらを使ったアクセス制御の枠組みを規定している．

アクセス制御には，(1) 管理操作の発行元，(2) 管理操作に対して保護すべきMIB（保護対象）または開放すべきMIB（開放対象），および (3) 前記 (1) と (2) からアクセスの拒否または許可を決定するための規則が必要であり，それぞれに対して“initiators”，“targets”，および“rule”の各MOクラスを規定している．**図1**に保護対象と開放対象の例を，また，図1における“initiators”，“targets”，および“rule” MOインスタンスの属性値を**表1**に示す．

保護対象と開放対象は，ともに，MIBに対する値取得，値設定，および通知等の管理操作であるCMIS（共通管理情報サービス）[8]のスコープとフィルタを用いて指定する．スコープは管理操作対象となるMOインスタンスの範囲を，範囲の基点となるベースオブジェクトインスタンス（BOインスタンス）とあわせて名前木上で指定する．フィルタはスコープで指定された範囲内にあるMOインスタンスから管理操作の対象をさらに特定し，MOインスタンスの属性値の大小，一致，または属性自体の存在等を論理式で表す．**表2**に4種類のスコープの定義を，**図2**にこれらの例をそれぞれ示す．

表1のtargets1はBをBOインスタンスとするBaseTo1stLevelスコープだけで指定され，targets2はAをBOインスタンスとする2ndLevelOnlyスコープとGを除くフィルタで指定される．

図3に5種類の規則とこれらを使ったアクセスの拒否と許可の決定の枠組みを示す．アイテム拒否規則（item deny rule）とアイテム許可規則（item grant rule）の粒度には（1）管理操作単位，(2) MOインス

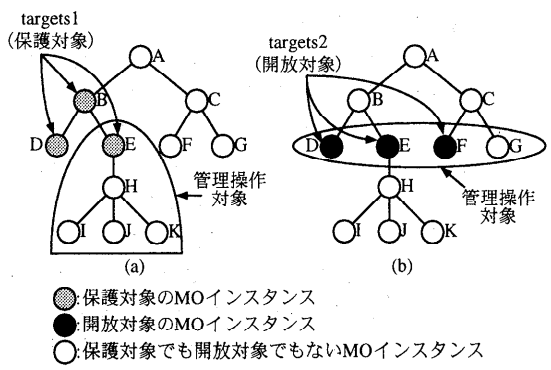


図1 保護対象と開放対象の例
Fig. 1 Example of access denied/granted MO instances.

表1 図1における“initiators”，“targets”，および“rule” MOインスタンスの属性値
Table 1 Attributes values of “initiators”, “targets”, and “rule” MO instances in Fig. 1.

MOインスタンス	属性値
initiators X	X
initiators Y	Y
targets1	MOインスタンスB，DおよびE
targets2	MOインスタンスD，EおよびF
rule1（アイテム拒否規則）	Xはtargets1にアクセス不可
rule2（アイテム許可規則）	Yはtargets2にアクセス可
rule3（デフォルト規則）	名前木のすべての管理情報へのアクセス不可

表2 4種類のスコープの定義
Table 2 Definition of four CMIS scopes.

種類	定義
BaseObject	BOインスタンスのみを範囲とする．
BaseTo*N*thLevel	BOインスタンスから第*N*レベル下位のMOインスタンス群までを範囲とする．BOインスタンス自身も含む．
***N*thLevelOnly**	BOインスタンスからちょうど第*N*レベル下位にあるMOインスタンス群を範囲とする．
WholeSubtree	BOインスタンスの下位レベルにあるMOインスタンス群すべてを範囲とする．BOインスタンス自身も含む．

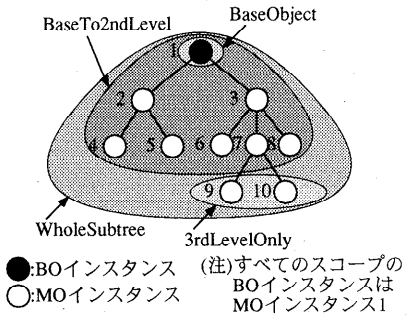


図2 4種類のスコープの例
Fig. 2 Example of four CMIS scopes.

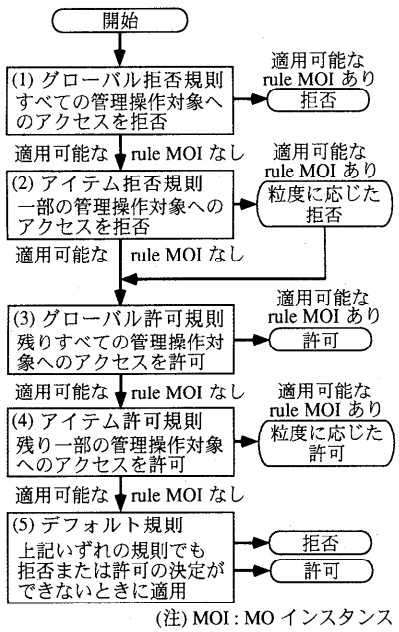


図3 国際標準に基づくアクセスの拒否と許可の決定の枠組み
Fig. 3 Framework of access decision based on International Standard.

タンス単位，および（3）属性単位がある．

標準には規定されていないが，いずれの粒度の場合でも，アクセス制御を実装するためには，アクセスの拒否と許可を決定するアルゴリズムが必要となる．

3. 従来のアルゴリズムとその問題点

粒度を属性単位とする，または，保護対象と開放対象の指定にフィルタを使うアルゴリズムを考える場合，フィルタで指定された個々の属性値の大小，一致，または属性自体の存在等を検査しなければならず，処理が複雑になり，処理速度の観点から実用的ではない[9),10)]．

一方，保護対象または開放対象の指定にフィルタを使わず，粒度を管理操作単位とする場合，管理操作対象が保護対象と交わりを持つか否か，ならびに，管理操作対象が開放対象に含まれるか否かを決定するだけで十分であるため，高速にアクセスの拒否と許可の決定ができる．すでにこのためのアルゴリズム[9),10)]が提案されている．これらのアルゴリズムは，管理操作のスコープと“targets” MO インスタンスのスコープ（BaseToNthLevel 等）の種類，スコープのレベル（N の値）および BO インスタンスの識別名の値から決定する．

しかしながら，粒度を管理操作単位とするアクセス制御では，管理操作対象と保護対象が交わりを持つと，アクセスが拒否されていないにもかかわらず，実際にはアクセスできない MO インスタンスが生じたり，また，管理操作対象が開放対象に含まれなければ，アクセスが許可されているにもかかわらず，実際にはアクセスできない MO インスタンスが生じる．

以下，図1 (a) または (b) に示した名前木，ならびに，表1の“initiators”，“targets”，および“rule” MO インスタンスを例にこれを説明する．図1 (a) で initiator X が E を BO インスタンスとする WholeSubtree スコープを持つ管理操作を発行する場合，表 1 には適用するグローバル拒否規則がないため，図3の (2) へ処理が進み，表1の rule1 が適用される．このとき，E は保護対象に含まれるため，この管理操作は拒否され，H，I，J，および K はアクセスが拒否されていないにもかかわらず，実際にはアクセスできない．また，図1 (b) で initiator Y が A を BO インスタンスとする 2ndLevelOnly スコープを持つ管理操作を発行する場合，表 1 には適用するグローバル拒否規則，アイテム拒否規則，またはグローバル許可規則がいずれもないため，図3の (4) へ処理が進み，表1の rule2 が適用される．このとき，G が開放対象に含まれないため，この管理操作は許可されず，D，E，および F はアクセスが許可されているにもかかわらず，実際にはアクセスできない．

さらに，**図4** に示すように，名前木の構造と“rule” MO インスタンスの指定によっては，アクセスが許可されるにもかかわらず，実際には複数の MO インスタンスを指定するスコープだけを使ったどのような管理操作のアクセスもできない．

このため，複数の MO インスタンスを指定するスコープを使わず，個々の MO インスタンスへ管理操作を発行する必要があり，結果として，管理操作の発行回数の増加を招き，ネットワーク管理システム全体の効率が低下するという問題が生じる．

これに対し，保護対象または開放対象の指定にフィ

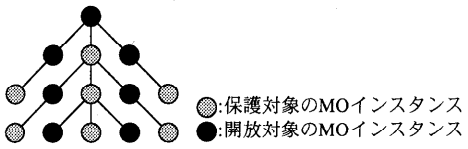


図4 スコープを持つ任意の管理操作がアクセス拒否される名前木の例
Fig. 4 Example of naming tree to which any operation with scopes is denied.

ルタを使わず，粒度をMOインスタンス単位とする場合には，上記の問題は生じないが，粒度を管理操作単位とするアルゴリズムと比較して処理は複雑になると考えられる．このため，処理効率を低下させず，粒度をより細かいMOインスタンス単位とするアルゴリズムが望まれる．

そこで以下に，保護対象または開放対象の指定にフィルタを使わないことを前提とし，粒度をMOインスタンス単位とする効率的なMIBのアクセス制御を実現するアルゴリズムを提案する．

4. 提案アルゴリズム

4.1 基本方針

粒度をMOインスタンス単位とする場合，処理が従来アルゴリズムと比較して複雑になるため，システム起動時およびアソシエーション確立時に前処理（以下，システム起動時前処理およびアソシエーション確立時前処理とそれぞれ呼ぶ）を行い，管理操作を受信してからアクセスの拒否と許可を決定する処理の負荷を軽減する方式[11]とする．

図5に提案アルゴリズムのフローチャートを，また，**図6**に提案アルゴリズムによるアクセスの拒否と許可の決定機構を示す．

図6で(1)管理操作が発行されると，(2)管理操作よりBOインスタンスとスコープの種類を抽出し，(3)これらをキーとして対応表を検索し，管理操作対象となるMOインスタンスを特定する．次いで，(4)特定したMOインスタンスとアソシエーション確立時前処理で収集した保護対象または開放対象との交わりを検出し，アクセスの拒否と許可を決定する．

また，M-CREATEやM-DELETE等の管理操作により，MOインスタンスが生成または削除される場合には，対応表の更新を行う．

4.2 前処理

4.2.1 システム起動時前処理

図7にシステム起動時前処理のフローチャートを示す．ここでは，すべてのMOインスタンスをBOインスタンスとするすべてのスコープの種類ごとに，スコープとそれが網羅するMOインスタンスの対応関係（以下，対応表と呼ぶ）を以下のステップにより求める．

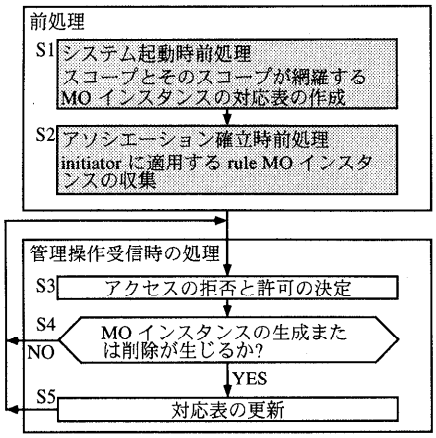


図5 提案アルゴリズムの概要
Fig. 5 Overview of proposed algorithm.

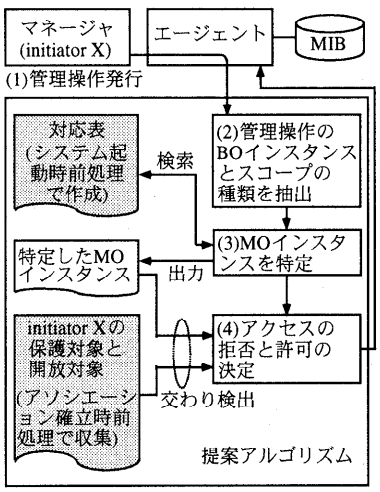


図6 アクセスの拒否と許可の決定機構
Fig. 6 Access decision mechanism.

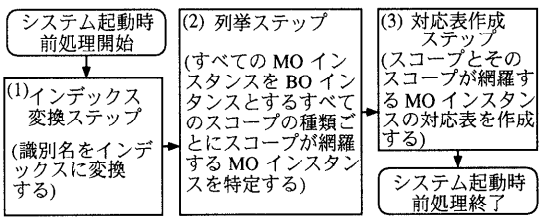


図7 システム起動時前処理のフローチャート
Fig. 7 Flow chart of system start-up phase.

4.2.1.1 インデックス変換ステップ

BOインスタンスの識別名は固定長ではないため，これを固定長のインデックスに変換し，対応表へのア

クセスを高速にする．ここでは，不定長ビット列を固定長ビット列に変換するハッシュ関数[12]を使う．

4.2.1.2 列挙ステップ

すべてのMOインスタンスをBOインスタンスとするすべてのスコープの種類ごとに，スコープが網羅するMOインスタンスを特定するため，大きさ $n \times n$（n は名前木のMOインスタンスの個数）の行列 A および C を以下のように定義する[13]．行列 A は名前木の構成を表し，MOインスタンス i がMOインスタンス j の1つ上位のとき，第 i 行第 j 列（(a_{ij})）の値が1である．行列 A^k は $k$thLevelOnlyスコープが網羅するMOインスタンスを，行列 C^k はBaseTo$k$thLevelまたはWholeSubtreeスコープが網羅するMOインスタンスを特定するために使う．BaseToNthLevelスコープとBaseToMthLevelスコープ（$N < M$）が網羅するMOインスタンスが同じ場合には，BaseToMthLevelスコープを対応表に登録しない．また，BaseToNthLevelスコープとWholeSubtreeスコープが網羅するMOインスタンスが同じ場合には，BaseToNthLevelスコープを対応表に登録しない．

$$(a_{ij}) \stackrel{\text{def}}{=} \begin{cases} 1 & (\text{名前木でMOインスタンス } i \text{ がMOインスタンス } j \text{ の1つ上位のとき}), \\ 0 & (\text{上記以外のとき}). \end{cases} \quad (1)$$

$$A^0 \stackrel{\text{def}}{=} E \ (\text{単位行列}),$$

$$A^k \stackrel{\text{def}}{=} A * A^{k-1} \quad (k \text{は名前木のレベル}, k \geq 1). \quad (2)$$

$$C^k \stackrel{\text{def}}{=} A^0 + A^1 + A^2 + \cdots + A^k. \quad (3)$$

4.2.1.3 対応表作成ステップ

前記インデックス変換ステップと列挙ステップより，スコープとそのスコープが網羅するMOインスタンスの対応表を作成する．図1の名前木を例とする場合の対応表を**表3**に示す．値が“1”のMOインスタンスが該当する行のスコープに含まれる．

4.2.2 アソシエーション確立時前処理

アソシエーション確立時に，そのinitiatorに適用するすべてのrule MOインスタンスを収集し，その後，rule MOインスタンスが示す保護対象または開放対象を対応表から求める．

4.3 管理操作受信時の処理

4.3.1 アクセスの拒否と許可の決定

図8にアクセスの拒否と許可の決定のフローチャートを示す．“∧”，“¯”はそれぞれビットごとの論理積および論理否定を表す．

表3 対応表の例
Table 3 Example of mapping table.

スコープ		MOインスタンス										
BOインスタンス（注1）	スコープの種類	・・・	E	・・・	H	I	J	K
省略	省略	省略						
E	BaseObject	*	1	*	0	0	0	0
E	BaseTo1stLevel	*	1	*	1	0	0	0
E	WholeSubtree	*	1	*	1	1	1	1
E	1stLevelOnly	*	0	*	1	0	0	0
E	2ndLevelOnly	*	0	*	0	1	1	1
省略	省略	省略						
H	BaseObject	*	0	*	1	0	0	0
H	WholeSubtree	*	0	*	1	1	1	1
H	1stLevelOnly	*	0	*	0	1	1	1
I	WholeSubtree	*	0	*	0	1	0	0
J	WholeSubtree	*	0	*	0	0	1	0
K	WholeSubtree	*	0	*	0	0	0	1

（注1）BOインスタンスはインデックスのビット列であるが，ここでは図1のアルファベットで表した．
（注2）“*”は値がすべて0である．

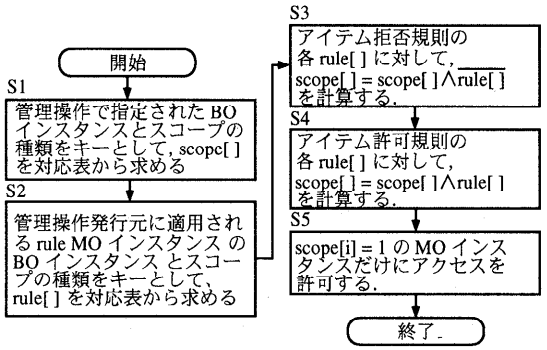


図8 アクセスの拒否と許可の決定のフローチャート
Fig. 8 Flow chart of access decision.

図8で，“scope[ ]”は管理操作で指定されるスコープが網羅するMOインスタンスを表す大きさ n（n は名前木のMOインスタンスの個数）の配列であり，“rule[ ]”はruleのMOインスタンスによって指定される保護対象または開放対象を表す大きさ n の配列である．図1(a)ならびに表1を例としてアクセスの拒否と許可の決定を示す．initiator XからBOインスタンスがEであるWholeSubtreeスコープを持つ管理操作が発行されるとする．

S1: 配列“scope[ ]”は次のようになる．

	A	B	C	D	E	F	G	H	I	J	K
scope[ ] =	0	0	0	0	1	0	0	1	1	1	1

S2: 表1のrule1が適用される．表1のrule1と対応表から配列“rule[ ]”は次のようになる．

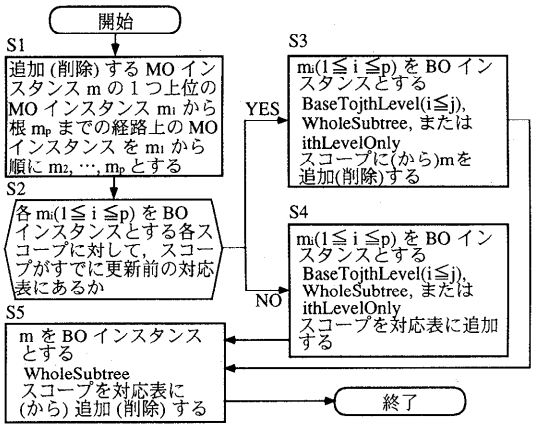


図9 対応表更新のフローチャート
Fig. 9 Flow chart of updating mapping table.

	A	B	C	D	E	F	G	H	I	J	K
rule[ ] =	0	1	0	1	1	0	0	0	0	0	0

S3: scope[ ] ∧ $\overline{\text{rule[ ]}}$ を計算する．

scope[ ] = scope[ ] ∧ $\overline{\text{rule[ ]}}$

	A	B	C	D	E	F	G	H	I	J	K
=	0	0	0	0	0	0	0	1	1	1	1

S4: initiator X に適用するアイテム許可規則はないため，S5 へ進む．

S5: S3 で，“scope[H]”，“scope[I]”，“scope[J]”，および“scope[K]”の値が 1 である．したがって，E へのアクセスを拒否し，H，I，J，および K へのアクセスを許可する．

4.3.2 対応表の更新

M-CREATE や M-DELETE 等の管理操作により，MO インスタンスが追加または削除される場合，**図 9** に示す更新処理のフローチャートに従い，対応表を更新する．

図 9 の S1 では 4.2 節の式 (1) で定義した行列 A を使って求める．行列 A の定義より，名前木で，MO インスタンス s の 1 つ上位の MO インスタンス t は $(a_{ts}) = 1$ を満たす．これより，追加または削除する MO インスタンスの上位の MO インスタンスを根に向かって順に列挙する．MO インスタンスの追加または削除がある場合，システム起動時前処理を再度実行するのではなく，先に列挙した MO インスタンスを BO インスタンスとするスコープだけを更新する．

5. 提案アルゴリズムの評価

5.1 計 算 量

5.1.1 提案アルゴリズムの計算量

表 4 に提案アルゴリズムの理論的な計算量を示す．ここで，n，L，および R はそれぞれ名前木を構成する MO インスタンスの個数，MO インスタンスの（ASN.1 符号化済み）識別名の最大長（byte），ならびに initiator に適用される rule MO インスタンスの個数を示す．

（1）**システム起動時前処理**（表 4 (1)）

すべての MO インスタンスの識別名をインデックスに変換するインデックス変換ステップの計算量は $O(Ln)$ となる．また，列挙ステップの計算量は $O(n^3)$ [13] である．したがって，システム起動時前処理の計算量は $O(Ln + n^3)$ となる．

（2）**アソシエーション確立時前処理**（表 4 (2)）

initiator に適用される rule MO インスタンスを検索する計算量は $O(R)$ となる．また，適用される rule MO インスタンスと対応表から保護または開放対象を特定する計算量は $O(Ln)$ である（5.1.1 項 (3) 参照）．したがって，アソシエーション確立時前処理の計算量は $O(LRn)$ となる．

（3）**アクセスの拒否と許可の決定**（表 4 (3)）

初めに，BO インスタンスの識別名とスコープの種類をキーとして，管理操作対象を特定する対応表検索の計算量を示す．BO インスタンスの識別名をインデックスに変換する計算量は，BO インスタンスの識別名の長さに比例し，$O(L)$ となる．また，このインデックスとスコープの種類をキーとする対応表検索は 4.2.1.1 項で参照したハッシュ関数を使うことにより定数時間内で終了する．したがって，対応表検索の計算量は $O(L)$ となる．

次に，上記で特定した管理操作対象とアソシエーション確立時前処理で収集した保護対象または開放対象との交わりを決定するための計算量を示す．ここでは，大きさ n の配列“rule[ ]”の要素ごとの論理否定および“scope[ ]”と“rule[ ]”の要素ごとの論理積を計算する．したがって，交わりを検出するための計算量は $O(n)$ となる．

以上により，アクセスの拒否と許可の決定の計算量は $O(L + n)$ となる．

（4）**対応表の更新**（表 4 (4)）

追加または削除する MO インスタンスの上位の MO インスタンスを根に向かって順に列挙する計算量は，枝分れのない名前木に（から）葉を追加（削除）する

表4 提案アルゴリズムの計算量
Table 4 Time complexity of proposed algorithm.

前処理	(1) システム起動時前処理	$O(Ln+n^3)$
	(2) アソシエーション確立時前処理	$O(LRn)$
管理操作発行	(3) アクセスの拒否と許可の決定	$O(L+n)$
MO インスタンス生成/削除	(4) MO インスタンス生成/削除による対応表の更新	$O(n)$

ときに最大 $O(n)$ となる．列挙した各 MO インスタンスの対応表の列およびこの MO インスタンスを BO インスタンスとする対応表のエントリの定数個の要素を更新する計算量は定数である．したがって，対応表の更新の計算量は $O(n)$ となる．

5.1.2 従来アルゴリズムとの比較

5.1.1 項（3）の結果より，管理操作を受信した際に実行する提案アルゴリズムの計算量は $O(L+n)$ であるが，一方，粒度を管理操作単位とする従来アルゴリズムの計算量は $O(LRn)$[14]である．ここで，L と R は名前木の深さにそれぞれ依存し，一般に，n の値にともない増加するため，いずれのアルゴリズムの計算量も n が支配的であり，$O(n)$ といえる．

5.1.3 前処理の効果

4.2 節で示した 2 つの前処理を行わず，管理操作が発行されるたびに，管理操作対象の MO インスタンスを特定し，これら個々の MO インスタンスが保護対象または開放対象に含まれるか否かをしらみつぶしに決定する単純なアルゴリズムを使って，粒度を MO インスタンス単位とするアクセス制御を行う場合と比較し，前処理の効果を示す．

単純なアルゴリズムでは，管理操作対象の MO インスタンスごとに，保護対象または開放対象に含まれるか否かをしらみつぶしに検査する．このため，計算量は管理操作対象に含まれる MO インスタンスの個数と保護対象または開放対象に含まれる MO インスタンスの個数の積 $O(n^2)$ となり，提案アルゴリズムの計算量が $O(n)$ であることを考慮すると，前処理の効果は大きい．

5.2 処理時間

提案アルゴリズムを C 言語により実装し，提案アルゴリズムの処理時間を SUN SPARC Server 670MP（CPU 数 1）上で計測した．計測に用いた名前木は根をレベル 0 とする深さ 9 までの標準二分木とした．

図 10 に名前木の MO インスタンスの個数に対するシステム起動時前処理時間を，ならびに，**図 11** に名前木の MO インスタンスの個数に対するアソシエーション確立時前処理時間を示す．次いで，**図 12** に名前木の MO インスタンスの個数に対するアクセスの拒否と許可の決定時間を，**図 13** および **図 14** にそれぞれ名前木の MO インスタンスの個数に対する MO インスタンス生成/削除時の対応表の更新時間を示す．

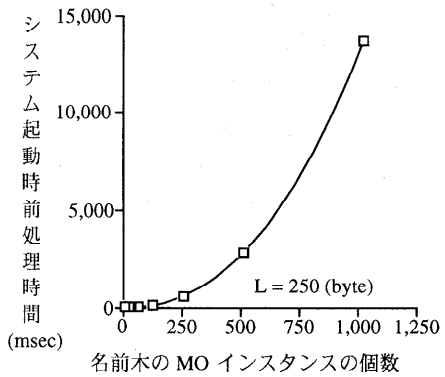


図 10 システム起動時前処理時間
Fig. 10 Processing time for system start-up phase.

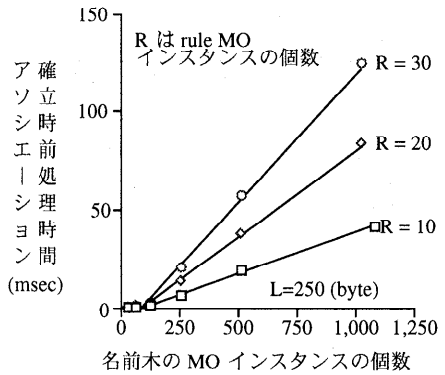


図 11 アソシエーション確立時前処理時間
Fig. 11 Processing time for association establishment phase.

図 10 から 図 14 まで，いずれにおいても，表 4 で示した結果と一致している．図 12 より，名前木の MO インスタンスの個数が 1,023 個で，BO インスタンスの識別名の長さが 250 byte の場合，アクセス拒否と許可を決定するための処理時間は 8 msec 程度である．これはアクセス制御を含まない管理操作の処理時間が数 100 msec であることを考えると，提案アルゴリズムのアクセス制御によるオーバヘッドは小さい．

また，4.2 節で示した 2 つの前処理を行わず，管理操作が発行されるたびに，管理操作対象の MO イン

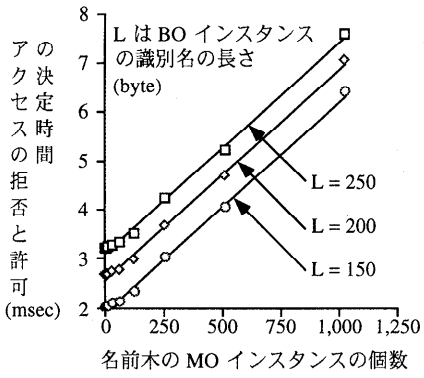


図12 アクセスの拒否と許可の決定時間
Fig. 12 Processing time for access decision.

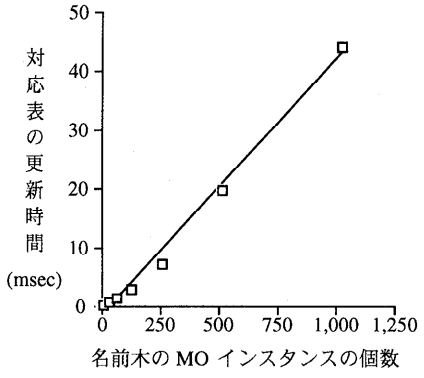


図13 MO インスタンスの生成による対応表の更新時間
Fig. 13 Processing time for updating mapping table in creating MO instance.

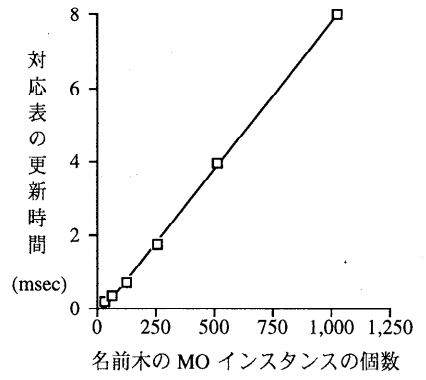


図14 MO インスタンスの削除による対応表の更新時間
Fig. 14 Processing time for updating mapping table in deleting MO instance.

スタンスを特定し，これら個々の MO インスタンスが保護対象または開放対象に含まれるか否かをしらみつぶしに決定する単純なアルゴリズムを使って，MO インスタンス単位のアクセス制御を行う場合と比較して，前処理の効果を示す．MO インスタンスの個数が 1,023 個ならびに BO インスタンスの識別名の長さが 250 byte のとき，提案アルゴリズムのアクセスの拒否と許可の決定時間が 8 msec 程度であるのに対し，単純なアルゴリズムの決定時間は 10 sec 程度と非常に大きい．

以上，単純なアルゴリズムと比較して，前処理の導入により，アクセスの拒否と許可の決定の処理時間を大幅に削減する提案アルゴリズムは有効である．

5.3 対応表の記憶容量

対応表に必要な記憶領域の大きさは，1 つの整数型変数を 4 byte で格納する一般的なワークステーションの場合，12.6 Mbyte 程度であるため，主記憶上に常駐させることが可能である．

6. おわりに

本論文では国際標準で規定される管理オブジェクトを使う MIB のアクセス制御のアルゴリズムを提案し，その計算量や処理時間等の評価結果を示した．

粒度が管理操作単位の場合，アクセスが許可されているにもかかわらず，実際にはアクセスできない管理オブジェクトインスタンス（MO インスタンス）が生じる．このため，複数の MO インスタンスを指定するスコープを使わず，個々の MO インスタンスへ管理操作を発行する必要があり，結果として，管理操作の発行回数の増加を招き，ネットワーク管理システム全体の処理の効率が低下するという問題がある．

提案アルゴリズムはこの問題を解決するため，効果的な前処理を導入することにより，粒度をより細かい MO インスタンス単位としながら，管理操作を受信した際に実行する処理の計算量を $O(n)$ に抑え，従来アルゴリズムと同程度の計算量で MIB のアクセス制御を行う．また，提案アルゴリズムを実装し，管理操作の処理時間に対するアクセス制御のオーバヘッドは小さいことを示した．これにより，ネットワーク管理システム全体の処理の効率を低下させることなく，セキュリティ品質の向上が図れる．

謝辞 日頃ご指導いただく国際電信電話（株）研究所 村上仁己取締役所長に感謝します．また，ご討論いただいた鈴木健二副所長に感謝します．

参考文献

1) CCITT Recommendation X.701 | *ISO/IEC* 10040, *Systems management overview* (1992).
2) O'Mahony, D.: Security Considerations in a Network Management Environment, *IEEE Network*, Vol.8, No.3, pp.12–17 (1994).

3) Bhatti, S., Graham, K., Gurle, D. and Rodier, P.: Secure remotc management, *Proc. 4th International Symposium on Integrated Network Management*, pp.156–169 (1995).
4) Rudiger, G. and Thomas, H.: Security policies in OSI-management experiences from the DeTeBerkom project BMSec, *Computer Nètworks and ISDN Systems*, Vol.28, pp.499–511 (1996).
5) ITU-T Recommendation X.160, Architecturc for Customer Network Management Service for Public Data Networks (1994).
6) ITU-T Recommendation X.161, Definition of Customer Network Management Service for Public Data Networks (1994).
7) ITU-T Recommendation X.741 | ISO/IEC 10164-9, System Management: Objects and attributes for access control (1995).
8) CCITT Recommendation X.710 | ISO/IEC 9595, Common management information service definition (1991).
9) 大野陽介，依田育生，藤井伸朗：通信網オペレーションシステムにおけるアクセス制御適用法，電子情報通信学会通信方式研究会技術報告，Vol.94, No.39, pp.19–24 (1994).
10) Graham, K., Saleem, B. and Luca, D.: Secure Remote Management in thc ESPRIT MIDAS Project, *IFIP Trans. ULPAA*, pp.77–86 (1994).
11) 吉原貴仁，堀内浩規，杉山敬三，小花貞夫：ネットワーク管理における管理情報ベース（MIB）のアクセス制御アルゴリズムの提案，電子情報通信学会情報ネットワーク研究会技術研究報告，Vol.97, No.22, pp.17–22 (1997).
12) Stallings, W.: *SNMP, SNMPv2, and CMIP: the practical guide to network management standards*, Addison-Wesley Publishing Company (1994).
13) 吉原貴仁，堀内浩規，杉山敬三，小花貞夫：OSI管理操作の最適なスコープの組合せを求める近似アルゴリズムの提案，電子情報通信学会情報ネットワーク研究会技術研究報告，Vol.96, No.84, pp.35–40 (1996).
14) 大野陽介，依田育生：網管理システムへのアクセス制御機能の実装，電子情報通信学会通信方式研究会技術報告，Vol.96, No.143, pp.19–24 (1997).

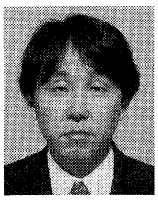
(平成 9 年 5 月 12 日受付)
(平成 10 年 1 月 16 日採録)

吉原 貴仁（正会員）

昭和 45 年生．平成 5 年東京工業大学情報工学科卒業．平成 7 年同大大学院理工学研究科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所属．この間，ネットワーク管理の研究に従事．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大会優秀賞を各受賞．電子情報通信学会会員．



杉山 敬三（正会員）

昭和 37 年生．昭和 60 年京都大学工学部情報工学科卒業．昭和 62 年同大大学院工学研究科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査．この間，OSI プロトコル実装方式，ネットワークアーキテクチャ，分散処理，ネットワーク管理，EDI の研究に従事．平成 6 年度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．

小花 貞夫（正会員）

昭和 28 年生．昭和 51 年慶応義塾大学工学部電気工学科卒業．昭和 53 年同大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．