権限集合の類似度を用いたアクセスルールのクラスタリングに基づくロール抽出方式

古川　諒†　　中江　政行†　　小川　隆一†

†NEC 共通基盤ソフトウェア研究所
211-8666 神奈川県川崎市中原区下沼部 1753

{r-furukawa@cb, m-nakae@bp, r-ogawa@bq}.jp.nec.com

あらまし　近年，組織の内部統制強化のために Role-based Access Control(RBAC) モデルに基づいたアクセス制御ポリシの徹底が求められている．RBAC の利用に際しては適切なロール設計が重要であるが，管理者によるトップダウンなロール設計は負荷が高く，アクセス設定からロールを抽出するボトムアップな設計による支援が必要である．しかし，既存のロール抽出方式は抽出されるロール数が多くなるという問題があり，ロール設計効率化の観点から望ましくない．本稿では，ユーザごとの権限集合の類似度を用いて，頻出する権限パターンを発見することで各ロールにユーザと権限の両者を多く割り当てることができるロール抽出方式を提案する．ルータのフィルタリング設定を対象とした実験により，提案方式は既存方式と比べ少ないロール数でよりアクセス設定の再現率が高いロール集合を抽出できることを確認した．

A Role Mining Method Based on Access Rule Clustering with the Similarity of Permission Sets

Ryo Furukawa†　　Masayuki Nakae†　　Ryuichi Ogawa†

†Common Platform Software Res. Lab., NEC Corporation
1753 Shimonumabe Nakahara-Cho Kawasaki-Shi Kanagawa 211-8666 Japan

{r-furukawa@cb, m-nakae@bp, r-ogawa@bq}.jp.nec.com

Abstract　As access control management to enterprise IT systems is crucial, an RBAC-based access control management system in needed. To design roles in such a system, because the top-down role designing by an administrator is consume high workload, the bottom-up approach that extracts roles from access control lists is needed. In this paper, we propose a role mining method that uses the similarity of permission sets to discover permission pattern to extract roles that assigned much user and permission. We evaluate the proposed method using network filtering rules as input data. As a result, the proposed method can extract the less number of roles than an existing method, and renders more reproduction rate. This result implicates that the proposed method is more effective from the viewpoint of decreasing role designing load.

1　はじめに

近年，J-SOX 法施行などの法制強化により，企業内での内部統制強化の意識が高まっており，不正を防止できる IT システムの再構築や，不正使用や改ざんのない IT 基盤の管理が必要となっている [1]．そのため，企業内の IT システムに対する，設定不備のないアクセス制御ポリシの徹底が重要であり，職務に基づく権限の分離 (職務分掌) が求められている．

こうした目的でよく使用される方法として RBAC(Role-based Access Control)[2] が知られている．RBAC では職務を表すロール (役割) に必要な権限及び所属するユーザを割当てることで，アクセス制御を実現する．

RBAC を用いて適切に職務分掌を行うためにはロールをどのように設計するかが非常に重要である．通常，ロールは中央の管理者が人事情報，組織構造などからトップダウンに設計するが，このような設計方式では各ロールの権限の把握などに多くの手間がかかる．一方で，近年，

アクセス設定を基にボトムアップにロールを抽出し，見える化するロール抽出方式が研究されている [3][4][5]．ボトムアップな方式とトップダウンな方式は相補的に利用できるため，優れたロールマイニング方式の開発は，ロール設計の効率化に有用である．

ロール抽出方式は，ロール設計効率化のために，少数のロールでより多くのアクセス設定を見える化する必要があると考えられる．しかし，文献 [3][5] などの既存のロール抽出方式は，アクセス設定全体を見える化ために必要なロール数の最小化を考えており，上記のような観点に基づいて設計されていない．

そこで，本稿ではより少数のロールでより多くのアクセス設定情報の見える化を目的としたロール抽出方式を提案する．提案方式はユーザごとの権限集合の類似度を用いてアクセスルールをクラスタリングすることでロール抽出を行う．本方式は各ロールにユーザ数，権限がなるべく多く割り当てられるようにロールを抽出する．そのため少数のロールで，入力アクセス設定をよく再現するロール集合を生成でき，ロール管理コストを低く，より多くのアクセス設定情報を見える化できることを実験により示す．

2　ロール抽出問題とは

2.1　RBAC とその課題

従来，アクセス制御はアクセス制御リスト (Access Control List: ACL) と呼ばれる，アクセスルールが複数記述された形式により設定されてきた．アクセスルールはあるユーザが，アクセスの対象となるオブジェクトに対してどのような操作が許可・不許可されているかを表すアクションの組で表される．

これに対して，RBAC[2] は職務や職位を表すロールを用いてアクセス制御を行う．ロールにはそのロールが持つべきパーミッション（リソースとアクションの組）とそのロールに所属するユーザが対応付けられており，この対応関係を用いてユーザとパーミッションをマッピングすることでアクセス設定できる．ロールとパーミッション，ユーザとの対応関係を設計することをロール設計と呼ぶ．

一般的に RBAC に基づいたアクセス管理システムでは，中央の管理者が役割定義書や人事組織情報を参照してトップダウンにロールを設計する．しかし，この設計方法は多くの時間，労力がかかり，システムの導入やメンテナンスに支障をきたす原因となっている．

このため，実際には代表的なロールのみを設計し，それ以外は必要になるたびに個別にアクセス設定を追加･変更するというアドホックなカスタマイズがされている．こうしたアドホックな手法には，管理者が現状のアクセス設定を把握できずに統制が困難になるという問題がある．

一方，業務サーバ等における既存のアクセス設定からユーザ，パーミッションをまとめることで，実状に即したロールを抽出することができる．このようなボトムアップなロール抽出方式を用いたロールの見える化により，図 1 のようにアドホックに追加されたアクセス設定のロール化や既存のロールの修正が可能となる．これをトップダウンな設計と相補的に利用することで上記のような問題を解決できると考えられる．

ロール設計効率化のためには，ロール抽出方式によって抽出されたロールの管理負荷を低減する必要がある．これには，抽出ロール数をおさえなければならない．しかし，既存のロール抽出方式 [3][5] は，全アクセス設定の見える化を前提にしているため，ロール数が非常に多くなってしまう．より効果的なロール抽出にはロール数減少と見える化する情報量の増加のトレードオフを考慮して，両者のバランスをとる必要があると考えられる．

そこで，本稿では少数のロールでより多くのアクセス設定を見える化できるロール抽出方式の開発を目的とする．

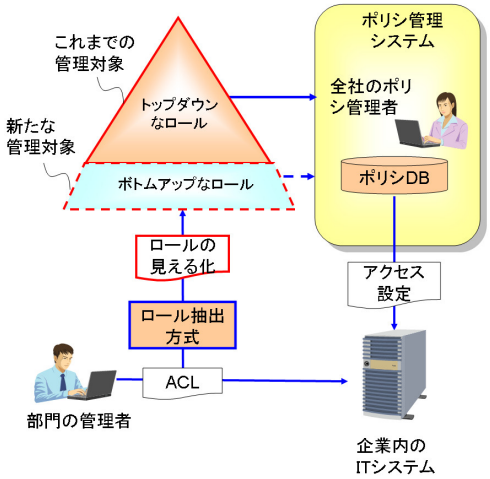

図 1: ロール抽出方式の役割

2.2　問題の定式化

本節ではロール抽出問題の定式化を行う．まず，ACL および RBAC を形式的に説明する．

ACL は，ユーザを u，オブジェクトを o，アクションを a，パーミッション $p = (o, a)$ と表記すると，$ACL = \{(u, p)\}$ と記述できる．

RBAC はロール r の集合 $R = \{r\}$，ロールへのユーザ割当 $UA = \{(r, u)\}$，ロールへのパーミッション割当 $PA = \{(r, p)\}$ を要素として持つ．そして，$\exists r \quad \in R, (r, u) \in UA, (r, p) \in PA$ を満たすとき，アクセスルール (u, p) に対応するアクセス設定がされていることになる．また，あるロール r_i に割り当てられたユーザの集合を

$U_i = \{\forall u, (r_i, u) \in UA\}$，パーミッションの集合を $P_i = \{\forall p, (r_i, p) \in PA\}$ としたとき，r_i と (U_i, P_i) は一対一対応の関係にある．そのため，一般性を失わずに $r_i = (U_i, P_i)$ と記述することができる．

ロール抽出問題は，$ACL = \{(u, p)\}$ を入力とし，ロール $r = (U, P)$ の集合 $R = \{r\}$ を抽出することを目的とする．ここで，RBAC の性質から，あるロール r をアクセスルール集合へ変換する操作を $Rule(r)$ と書くと，式（1）が成り立つ．

$$Rule(r) = U \otimes P \tag{1}$$

したがって，ロール抽出問題は ACL を式 (1) の形式のアクセスルール集合に分類する問題である．

また，出力されるロール集合の性質を測る指標として，ロール数 N_r 及び再現率 R_r をそれぞれ式 (2)，(3) で定義する．

$$N_r = |R| \tag{2}$$

$$R_r = |\bigcup_{\forall r_i \in R} Role(r_i))|/|ACL| \tag{3}$$

ロール数はロールの管理コストに関連する指標であり，ロール数が少なければ見える化されたロールの管理を行いやすいという性質を持つ．再現率は抽出されたロール集合を変換したアクセスルール集合の，基の ACL に対する包含率を示している．これはロール抽出によりどれだけアクセス設定情報を見える化できたかを表す指標であり，高いほうが優れたロール集合であるという性質を持っている．

したがって，本稿で扱うロール抽出問題は，式 (2) の最小化, 式 (3) の最大化を目的とし，ACL を式 (1) で表される形式のアクセスルール集合へ分類問題であると定式化できる．

3 提案方式

ロール抽出問題の望ましい解を得るためには，

- ロールに割り当てられたユーザを多くする
- ロールに割り当てられたパーミッションを多くする

必要がある．これには，多くのユーザが持つ，なるべく大きなパーミッションのパターンを発見してロールを抽出しなければならない．

パーミッションのパターン発見のためにはパーミッションの類似するユーザをまとめる必要がある．そのため，はじめにアクセスルール集合をユーザの軸で分類し，その後パーミッションの軸で分類することが妥当であると判断した．

そこで，パーミッションのパターンの大きさを予測する尺度としてパーミッション集合の類似度を定義し，上記設計に基づいて類似度を用いたアクセスルールの分類に基づくロール抽出方式を提案する．

3.1 パーミッション集合の類似度

パーミッション集合の類似度を以下のように定義する．

[定義１：パーミッション集合の類似度]
２つのパーミッション集合 P_1, P_2 の類似度は，以下の式 (4) で定義することができる．

$$Sim(P_1, P_2) = 2|P_1 \cap P_2|/(|P_1| + |P_2|) \tag{4}$$

この類似度は２つのパーミッション集合間で共通するパーミッションの割合が多きければ類似度が高く，小さければ類似度が低くなるように設計されている．そのため，類似度が高いパーミッション集合を分類することで大きなパーミッションのパターンを発見することができる．

3.2 アルゴリズム

本節では，パーミッション集合の類似度を用いたロール抽出方式を提案する．提案方式の処理を以下に示す．

[提案方式]

1. ACL を入力し，ロール集合 $R = \emptyset$ とする
2. ACL からパーミッション集合の類似度を用いてユーザグループ集合 $Sg = \{Gu\}$ を生成
3. Sg 内の各ユーザグループ Gu_i ごとに，ACL をアクセスルール集合 $ARu_i = \{(u_m, *, *), u_m \in Gu_i\}$ に分類
4. 各 ARu_i に対して，ARu_i 内の全ユーザに共通のパーミッションをもつアクセスルールだけを取り出し $ARup_i$ とする
5. 各 $ARup_i$ に含まれるユーザの集合 U_i，パーミッションの集合 P_i を用いたロール $r_i = (U_i, P_i)$ を R に追加
6. ロール集合 R を出力

提案方式はステップ２におけるパーミッション集合の類似度を用いたユーザの分類を主要な処理としており，パーミッションの軸での分類は共通パーミッションを取り出すだけの簡単な処理となっている．以降ではステップ２の処理をユーザ分類方式と呼び，詳細に述べる．

3.2.1 ユーザ分類方式

提案方式におけるユーザ分類方式は，パーミッション集合の類似度に基づいて，類似度の高いパーミッション集合を持つユーザ同士が一つのグループとなるように分類する．

これを達成するために，各分類内のユーザの持つパーミッション集合の類似度が定められた閾値以上かつ，ユーザ数が多くなる分類方式を提案する．閾値を操作することで，ユーザ数，類似度を調節が可能になり，適切な値に設定することで両者のバランスの取れた分類を行うことができる．

提案するユーザ分類方式はユーザを葉ノードとした二分木を生成する分類木生成方式と，分類木からユーザのグループを抽出するグループ抽出方式により構成されている．以降，分類木生成方式，グループ抽出方式それぞれについて詳細に述べる．

(1) 分類木生成方式

分類木生成方式は，ユーザの分類に利用できる木構造である，ユーザ分類木を生成する方式である．

ユーザ分類木は，ユーザ数と同じ数の葉ノードを持ち，ユーザと葉ノードが一対一対応する二分木である．また，パーミッション集合の類似度に基づいて分類を可能とするために，葉ノード l_i に対応するユーザ $u(l_i)$ に設定されたパーミッション集合を $P(u(l_i))$ と表したときに，その類似度 $Sim(P(u(l_i)), P(u(l_j)))$ が高いほど葉ノード間の枝の本数が少なくなる (パス長が短い) 性質を持つ．

分類木生成方式は，各葉ノードに対応するユーザのパーミッション集合間の類似度を基に，ノード間の類似度を計算し，ボトムアップにユーザ分類木を構築する．ノード間類似度は以下のように定義される．このノード間類似度は，各ノード以下の部分木の葉ノードに対応するユーザグループ間の最悪パーミッション類似度である．

[定義２：ノード間類似度]

２つのノード間の類似度は，あるノード n_i を根ノードとする部分木に含まれる葉ノードの集合を $L(n_i)$ としたとき，式５で定義する．

$$Sim_n(n_i, n_j) = \min_{l_k \in L(n_i), l_m \in L(n_j)} Sim(P(u(l_k)), P(u(l_m))) \quad (5)$$

ノード間類似度を用いた分類木生成方式を以下に示す．ここで，ある中間ノード n_i の２つの子ノードを $c_1(n_i), c_2(n_i)$，親ノードを $p(n_i)$ と表す．この手法は階層的クラスタリング方式 [6] に基づいた方式となっている．

[ユーザ分類木生成方式]

1. 各ユーザに対応する葉ノードを生成し，ノード集合 N に格納する
2. $Sim_n(n_i, n_j), \forall n_i, n_j \in N$ を計算し，ノード間類似度テーブル Ts に登録する
3. Ts から類似度が最大のノードの組 (n_i, n_j) を選択する
4. 新しいノード n_p を生成し，$p(n_i) = n_p$, $p(n_j) = n_p, c_1(n_p) = n_i, c_2(n_p) = n_j$ とする
5. $N = (N \cap \{n_p\}) \backslash \{n_i, n_j\}$ とする
6. Ts に $Sim_n(n_p, n_m), \forall n_m \in N$ を追加し，$Sim_n(n_i, n_m), Sim_n(n_j, n_m), \forall n_m \in N$ を消去する
7. $|N| = 1$ の場合，$n \in N$ を分類木の根ノードとして出力する．そうでない場合ステップ３へ戻る

この方式は，パーミッション集合の類似度が高い関係にある２ユーザを取り出したとき，そのユーザに対応する葉ノード同士間のパス長が短い関係にある分類木を生成できる．したがって，類似度の高いユーザで大きな部分木を構成されている．

(2) グループ抽出方式

次に，ユーザ分類木からユーザグループを抽出するグループ抽出方式について述べる．ここでは，ユーザグループ内のユーザのパーミッション集合の類似度が必ず定められた閾値以上になるようにユーザグループを生成する．

グループ抽出方式は，ユーザ分類木から部分木を抽出し，各部分木に含まれる葉ノードに対応するユーザをグループとする．

以下にユーザグループ抽出方式を，図２にステップ３における動作を示す．ここで，与えられるユーザ分類木の根ノードを n_r，ユーザ分類木内のあるノード n_i を根ノードとした部分木の葉ノード集合を $L(n_i)$ と表す．

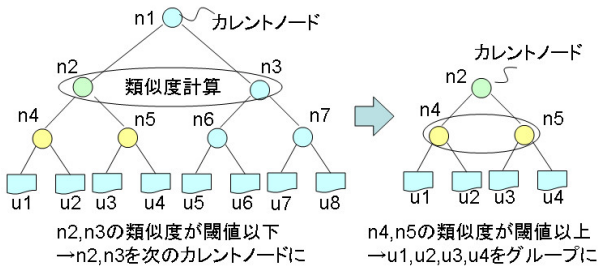

図 2: グループ抽出方式の動作

[グループ抽出方式]

1. 閾値 θ を設定，ユーザグループ集合 $Sg = \emptyset$ とする．
2. カレントノード $n_c = n_r$ と設定する．
3. 以下を再帰的に実行する
   (a) n_c が葉ノードならば何も行わない
   (b) $Sim(c_1(n_c), c_2(n_c)) > \theta$ である場合，ユーザグループ $Gu_i = \{u(l_m)\}$, $\forall l_m \in L(n_c)$ を Sg に追加
   (c) それ以外の場合，$c_1(n_c), c_2(n_c)$ それぞれをカレントノード n_c として，順番にステップ 3 を再帰的に実行する
4. 最後にユーザグループ集合 Sg を出力

提案方式で利用するユーザ分類方式は図 2 から分かるように，ユーザ分類木の根ノードから深さ優先探索を行い，類似度が閾値以上になるユーザグループを抽出する方式である．

また，ステップ 3(a) でどのユーザとも類似度が閾値以上にならないユーザはグループとして取り出さないようにしており，これにより個人グループが生成されないようになっている．

このようにユーザ分類方式は，類似度の高いパーミッション集合を持つユーザ同士で大きな部分木を構成するようにまとめ，そこから，類似度が閾値以上となるように部分木ベースでユーザのグループを抽出する．このため，パーミッション集合の類似度が閾値が高く，ユーザ数の多いグループを生成できる．

このようなユーザ分類方式を用いることで，提案方式はユーザ，パーミッションがともに多く割り当てられたロールを抽出することができ，ロール数，再現率に優れたロール集合の抽出を行うことができる．

4 実験と考察

4.1 実験方法

本実験では，社内で利用されていたルータのフィルタリング設定を用いて提案方式を評価する．このフィルタリング設定はユーザとしてアクセス元の IP アドレス，リソースとしてアクセス先の IP アドレス・ポート番号，アクションとして通過を許可するプロトコルが記述されており，ルールが 1410 行，ユーザが 492 アドレスの規模である．

提案方式の評価には 2.2 節で述べたロール数 Nr と再現率 R_r 及び，ロール数あたりの再現率 R_r/N_r を用いる．ロール数が少なく，再現率が高い方式の評価が高いため，ロール数あたりの再現率が高い方式が最も評価が高いものとする．

4.2 実験結果と考察

4.2.1 閾値による性能変化実験

まず，提案方式の閾値による性能の変化を評価する．閾値 $\theta = 0.9, 0.7, 0.5, 0, 3$ と変化させ，上記の実験方法を用いた評価を行った結果を表 1 に示す．

表 1: 予備実験結果

閾値	ロール数 N_r	再現率 R_r	R_r/N_r
0.9	45	0.36	0.0080
0.7	50	0.58	0.0116
0.5	82	0.60	0.0073
0.3	77	0.58	0.0075

表 1 から分かるように $\theta = 0.7$ の場合が R_r/N_r が最も高くなっており，最適値である．その他の値では，性能は大きく劣化するため，提案方式の性能は閾値に敏感であり，適切に閾値を選ぶ必要があるといえる．ただし，閾値と性能の関係は単峰性を有しているように見えるため，山登り的手法により比較的容易に最適値を決定できると予想される．

4.2.2 既存方式との比較実験

本節では，文献 [3] で提案されている Greedy 法との比較実験により提案方式を評価する．Greedy 法は再現率 1.0 にすることを前提としているが，正当に比較するためにユーザ数が 1 のロール (個人ロール) を除いて評価する．このようにすることの妥当性については 4.2.3 節に述べる．また，提案方式の閾値は前節で最適値であった 0.7 を用いている．

提案方式と既存方式によって抽出されたロール集合のロール数と再現率を表 2 に示す．表から分かるように，提案方式は既存方式に比べロール数が少なく再現率は大幅に上回っており，ロール数あたりの再現率が優れていることが分かる．

これは提案方式が，パーミッション集合の類似度に基づいてユーザを分類することにより，ユーザ間で多少パーミッションが異なる場合でも個人的パーミッションを無視してロールとしてまとめるため，ユーザ，パーミッション両者が多く割り当てられたロールを抽出できるためだと考えられる．

表 2: 実験結果

	ロール数 N_r	再現率 R_r	R_r/N_r
提案方式	50	0.58	0.0116
既存方式	67	0.48	0.0071

このように提案方式によって生成されたロール集合は管理負荷が低く，かつ，より多くのアクセス設定情報を見える化することができる．このロール集合を中央の管理者にフィードバッ

クすることで，ロール設計の効率化を行うことができると考えられる．

4.2.3 個人ロールを除くことの妥当性

提案方式は個人ロールを抽出せず，4.2.2 節では既存方式もこれにあわせて個人ロールを含めず評価を行っている．こうすることの妥当性について考察を行う．

まず，個人ロールの多くは一時的に設定された，または特権的なアクセス設定をロール化したものであり，ロールとして中央で管理するべきものではないと考えられる．

また，個人ロールを抽出することによりロール数の爆発し，管理負荷が増大してしまう．個人ロールを含め，再現率を 1.0 とした場合の各方式のロール数を表 3 に示す．なお，提案方式は再現率を 1.0 にするために，ロール抽出時の非共通パーミッションを持つユーザも個人ロールとしている．

表 3: 個人ロールを含めたロール数

	提案方式	既存方式
ロール数	199	183

表 3 から分かるように，どちらの方式でも表 2 の結果に比べ，ロール数が大幅に増加していることが分かる．このように，個人ロールは，ロール数が大幅な増加を招き，ロールのメンテナンス性を低下させるものである．

これらのことから個人ロールは，その性質から管理対象とすべきでなく，また量が多いため管理を困難にするものである．このため，個人ロールは見える化しないべきであり，提案方式にてこれを抽出しないこと及び 4.2.2 節における評価方法は妥当であると考えられる．

5 関連研究

アクセス制御リストからロールを抽出する方式の研究はさまざまに取り組まれている．

文献 [3][5] は本稿で定義したロール抽出問題を再現率 1 に固定してロール数最小化問題として扱い，グラフ分割問題や組合せ最適化問題へ帰着して解いている．これらの方式はロールの最小化のみを考慮しているため，特権を持つユーザが多い場合に多くのパーミッションが割り当てられた個人ロールが抽出されやすく，管理負荷の観点からは好ましくない．

また，文献 [4] は本稿と同様に分類木を生成する方式を用いてロール階層の抽出を目的としている．ロールは階層構造を持つ場合が多いため，このようなアプローチは有用であるが，この方法ではロール階層を二分木で表現するため，そのまま利用することは難しい．

文献 [7][8] は抽出したロールへの意味づけを考慮しており，文献 [7] はユーザ，パーミッションの属性からロールの意味を考慮したロール抽出を，文献 [8] は既存のロール集合との差異をなるべく小さく，かつロール数が減少するようなロール抽出を行っている．このようなアプローチは，抽出したロールの理解を助けるものであり，今後取り組むべき課題であると考えている．

6 おわりに

本稿では，RBAC で用いるロールをアクセス制御リストから抽出する方式を提案した．本方式は，ユーザに設定されたパーミッション集合の類似度を基に，アクセスルールの分類を行う．これにより，既存方式と比較して，少ないロール数で再現率の高いロール集合を抽出できる．そのため管理負荷が低く，より効率的なロールの見える化を行うことが可能である．

本方式により見える化されたロールを用いることでトップダウンなロール設計方式を補助し，より効率的にロール設計を行うことができる．

ただし，より効率的にボトムアップなロールをロール設計に利用するためには，各抽出ロールへ名前付けを行う必要がある．本研究の今後の課題は，名前の付けやすさを考慮したロール生成，自動的な名前付け技術などを確立し，統合的なロール設計方式を確立することである．

参考文献

[1] 企業会計審査議会，財務報告に係る内部統制の評価及び監査の基準並びに財務報告に係る内部統制の評価及び監査に関する実施基準の設定について（意見書），金融庁, Feb 2007, http://www.fsa.go.jp/singi/singi_kigyou/tosin/20070215

[2] R. S. Sandhu, E.J. Coyne, H.L. Feinstein, C.E. Youman (1996), "Role-Based Access Control Models", IEEE Computer 29(2): 38-47, IEEE Press, 1996

[3] Alina Ene et al, "Fast Exact and Heuristic Methods for Role Minimization Problems", SACMAT'08, pp.1-10

[4] J. Schlegelmilch, "Role Mining with ORCA", SACMAT'05, pp.168-176

[5] J. Vaidya et al, "The Role Mining Problem: Finding a Minimal Descriptive Set of Roles", SACMAT'07, pp.175-184

[6] Sung Young Jung, Taek-Soo Kim, "An Agglomerative Hierarchical Clustering using Partial Maximum Array and Incremental Similarity Computation Method", IEEE International Conference on Data Mining 2001

[7] I. Molloy, "Mining Roles with Semantic Meanings", SACMAT'08, pp.21-30

[8] J. Vaidya et al, "Migrating to Optimal RBAC with Minimal Perturbation", SACMAT'08, pp.11-20