UNIVERSITÀ DEGLI STUDI ROMA TRE
DIPARTIMENTO DI MATEMATICA E FISICA
CORSO DI LAUREA IN MATEMATICA

Sintesi della Tesi di Laurea Magistrale in Matematica

Uso della varianza per la valutazione dei vincoli nel controllo degli accessi

Maria Eleuteri

Anno accademico 2013/2014

Relatore
Prof. Roberto Di Pietro

Correlatore
Prof. Alessandro Colantonio

Indice

Sommario

Il *controllo degli accessi* è l'insieme dei processi utilizzati al fine di mediare le richieste di accesso ai dati. Per dati si intendono tutte le informazioni memorizzate su sistemi informatici e tutte le risorse necessarie per la loro gestione, all'interno di una azienda o di una qualunque organizzazione.

Nel panorama del controllo degli accessi si pone l'argomento dei *vincoli*: essi consistono in regole che devono essere rispettate nel gestire l'accesso ai dati, al fine di far valere le politiche di sicurezza dell'azienda. I vincoli vengono appunto costruiti seguendo queste politiche: rispettando i vincoli si rispettano le norme di sicurezza dell'organizzazione. Grazie ai vincoli è possibile decidere se una richiesta di accesso ai dati, che risulta nuova rispetto alla configurazione stabilita, debba essere concessa o negata.

Non sempre però risulta semplice definire i possibili vincoli, per questo è strato introdotto il *constraints mining*, cioè la possibilità di estrarre vincoli "de facto" a partire dalla configurazione di permessi esistente. Tra tutti i vincoli estratti è necessario scegliere quelli che corrispondono maggiormente a reali regole di sicurezza. Il problema è che non esiste ancora in letteratura un metodo di stima dei vincoli.

Il contributo principale di questa tesi sarà l'introduzione di una metrica in grado di valutare quanto un vincolo sia rilevante per l'organizzazione. A tal fine introdurremo un particolare indice di varianza, il quale ci permetterà di definire la nuova metrica per la valutazione delle regole. L'obiettivo sarà quello di ottenere, tramite un processo automatizzato, regole che siano corrette e che rispecchino il più possibile le reali norme di sicurezza vigenti all'interno dell'organizzazione.

Il particolare contesto in cui applicheremo il nostro studio sarà quello di RBAC (*Role Based Access Control*), un modello di controllo degli accessi basato sui ruoli, i quali consistono essenzialmente in un insieme di permessi. I vincoli in RBAC possono essere utilizzati durante il processo di estrazione dei ruoli (*role mining*) oppure nella gestione di nuove richieste, per poter mantenere i ruoli in regola con le norme di sicurezza.

Capitolo 1

Introduzione

Nell'ambito della sicurezza informatica il *controllo degli accessi* [1] consiste nell'insieme dei processi impiegati al fine di mediare le richieste di accesso ai dati. Al giorno d'oggi le imprese ripongono la maggior parte dei loro dati e delle informazioni su sistemi informatici, perciò nel tempo è aumentata la necessità di "tenere al sicuro" questi dati: essi devono essere protetti per diverse motivazioni, come privacy o prevenzione di frodi. Si pensi ai sistemi informatici delle banche o ai permessi che ogni membro di una organizzazione possiede, in base al proprio compito all'interno di essa. Di fondamentale importanza è la gestione dell'accesso a questi dati: un'organizzazione deve conoscere le identità degli utenti che richiedono l'accesso e saper stabilire se tali richieste debbano essere soddisfatte o negate.

Nozione chiave in questo contesto è quella dei *vincoli*: essi consistono in regole di amministrazione che devono essere rispettate nel gestire l'accesso ai dati. Si decide di concedere o negare un accesso all'utente che lo richiede, secondo le regole che i vincoli impongono, in quanto essi vengono stabiliti seguendo le politiche di sicurezza dell'organizzazione. Potrebbero sussistere, ad esempio, vincoli di separazione dei compiti che vietano l'assegnazione di determinate combinazioni di permessi ad uno stesso utente.

Un contesto specifico in cui si ritrova l'applicazione dei vincoli è quello del modello di controllo degli accessi RBAC (*Role Based Access Control* [12]). Il controllo degli accessi può essere effettuato seguendo diversi modelli ed RBAC risulta quello ad oggi più diffuso, per la semplicità ed i benefici che comporta. RBAC introduce il concetto di *ruolo*: insieme di permessi frequentemente assegnati ad un gruppo di utenti. L'utilizzo dei ruoli riduce il numero di relazioni da gestire, minimizzando gli sforzi di amministrazione, e favorisce il rispetto dei principi della *sicurezza*: "separazione dei compiti" e "least privilege".

Non sempre però l'adozione di RBAC comporta i benefici desiderati, poiché

spesso non si pone la giusta attenzione nella definizione dei ruoli. Nasce quindi la questione di come "disegnare" i ruoli (*role engineering*). Solitamente essi vengono creati tramite processi automatici che raggruppano i permessi più utilizzati: si parla di *role mining*, in quanto possono essere utilizzate tecniche di *data mining* per poter estrarre, da una grande quantità di dati, informazioni utili per la definizione dei ruoli [2]. Anche nell'attività di role engineering è importante introdurre il concetto di *vincolo*. In questo caso i vincoli possono essere imposti nel role mining, cioè durante il processo di estrazione dei ruoli, per far sì che essi vengano costituiti con maggior criterio, oppure al momento dell'assegnazione dei ruoli agli utenti o dei permessi ai ruoli, al fine di amministrare gli accessi in modo sicuro.

Purtroppo non sempre i vari vincoli sugli accessi sono noti o ben definiti: una possibile soluzione a questo problema potrebbe essere quella di dedurli dalla configurazione dei permessi in essere. Si parla quindi di *constraints mining*, ossia estrazione dei vincoli. L'idea è quella di partire dalle assegnazioni di permessi presenti all'interno dell'organizzazione e da esse ricavare le possibili regole, proprio come il *role mining* cerca di dedurre i ruoli.

Il problema principale che emerge in questo tipo di studi è che, dato l'insieme di vincoli estratti con tecniche di constraints mining, è necessario stabilire quali di essi siano più vicini alle politiche di sicurezza dell'azienda. Nasce quindi la necessità di stimare le candidate regole, poiché non sempre una tecnica automatizzata come un algoritmo riesce a produrre risultati che abbiano un significato riscontrabile nella realtà. La nostra osservazione è che non esiste ad oggi in letteratura un metodo per la stima dei vincoli, che possa quindi garantire la loro l'applicabilità.

Il contributo principale di questo lavoro di tesi sarà perciò l'introduzione di una metrica per la valutazione dei vincoli, in modo che sia possibile stabilire quanto un vincolo sia "buono", cioè utile dal punto di vista pratico. Questo si verifica quando i vincoli estratti possiedono un "significato di business", ossia rispecchiano il reale assegnamento di permessi agli utenti ed hanno una corrispondenza con le norme di sicurezza vigenti all'interno dell'organizzazione.

Con lo scopo della stima dei vincoli, introdurremo un particolare indice di varianza che ci permetterà di operare la valutazione delle regole estratte fornendone un "ranking", al fine di ottenere risultati significativi dal punto di vista applicativo. L'introduzione dell'indice di varianza è stata avviata grazie alla collaborazione con il gruppo di ricerca CRISES durante un periodo di studio presso l'Università *Rovira i Virgili* di Tarragona.

Nel corso della nostra ricerca abbiamo iniziato con lo studio dei recenti lavori sui vincoli, concentrandoci in seguito su una particolare tecnica di *constraints mining* presente in uno di essi: l'algoritmo analizzato genera possibili regole ma, come

precedentemente argomentato, non ne stima la *bontà*. Nell'intento di apportare un miglioramento al metodo di estrazione di regole descritto, abbiamo utilizzato il particolare concetto di "varianza semantica", ideato dal gruppo di ricerca citato e costituente l'indice di varianza da cui siamo partiti. Questa nozione di varianza, che tiene in considerazione le informazioni semantiche degli elementi su cui viene calcolata, è stata sviluppata in ambito di sicurezza informatica ma in un contesto applicativo diverso rispetto a quello di questa tesi: è stato perciò necessario ridefinire alcune componenti dell'indice di varianza, operandone una estensione, al fine di renderlo applicabile al nostro contesto. In seguito, nel metodo di valutazione proposto, abbiamo riunito il concetto di confidenza, facente già parte della tecnica di constraints mining analizzata, e la nostra estensione di quello di varianza, il quale costituisce invece un contributo innovativo proprio di questa tesi.

Il lavoro seguente è stato così suddiviso: nel Capitolo 2 viene descritto il tema del controllo degli accessi, nell'ambito della sicurezza informatica; viene approfondito il modello RBAC, descrivendo i vantaggi che derivano dalla sua adozione e le questioni legate al problema del role mining.

Nel Capitolo 3 viene introdotto il concetto di vincoli nel controllo degli accessi ed in particolare nel contesto del modello RBAC, spiegando cosa siano i vincoli, le motivazioni legate al loro impiego ed il centrale problema del constraints mining.

Nel Capitolo 4 vengono approfonditi due particolari lavori che propongono algoritmi di constraints mining.

Il Capitolo 5 contestualizza ed introduce il citato indice di varianza, descrivendo il contesto applicativo in cui questo particolare indice è stato sviluppato; viene inoltre illustrato un tipico esempio della sua applicazione.

Nel Capitolo 6 svolgiamo tutta la parte innovativa del lavoro: inizialmente rivisitiamo le modalità di calcolo di alcune componenti dell'indice di varianza, con conseguente modifica dello stesso, al fine di renderne possibile l'applicazione al nostro contesto di studio. Successivamente ci dedichiamo alla definizione della nuova metrica di valutazione e ne illustriamo funzionamento e motivazioni. Un esempio mostra l'utilità pratica di applicazione di tale metrica all'algoritmo di constraints mining.

Il Capitolo 7 trae conclusioni sul lavoro svolto e riporta possibili lavori futuri.

Capitolo 2

Controllo degli accessi

Al giorno d'oggi il crescente uso dei servizi informatici e della rete hanno portato alla necessità di proteggere dati e informazioni posseduti dalle aziende e dalle organizzazioni: di questo si occupa la *sicurezza informatica*. Uno degli aspetti della protezione dei dati è il *controllo degli accessi*, ossia la gestione delle possibilità di accesso di ogni utente, alle risorse disponibili, in base allo specifico compito che egli svolge all'interno dell'organizzazione.

Per regolare il controllo possono essere utilizzati diversi modelli, ossia rappresentazioni formali di modalità di controllo degli accessi. Un semplice modello sono le *matrici di controllo degli accessi* (ACM)[3], in cui sono elencati utenti e risorse, con la descrizione dei permessi che tutti gli utenti dell'organizzazione hanno su ogni risorsa presente, si veda l'esempio di Figura 2.1.

	Oggetto 1	Oggetto 2	...	Oggetto n
Soggetto 1	r, w	r		r
Soggetto 2	w	d		r, w
⋮				
Soggeto m	d	r		r, w, d

r = read
w = write
d = delete

Figura 2.1: Matrice di controllo degli accessi (ACM)

Questo e altri modelli però non risultano impiegabili in casi di organizzazioni costituite da un elevato numero di utenti. Di più facile applicabilità è il Role Based Access Control, modello di sostanziale importanza.

2.1 Role Based Access Control (RBAC)

Il *Role Based Access Control* [12] è un modello di controllo degli accessi basato sui *ruoli*. Al momento RBAC è il più comune modello impiegato [7] grazie alla funzionalità e ai benefici che derivano dalla sua applicazione.

L'introduzione del concetto di *ruolo* semplifica la gestione degli accessi: un ruolo identifica in modo unico un insieme di permessi e ad ogni utente vengono assegnati uno o più ruoli appropriati in base alle sue responsabilità all'interno dell'organizzazione. Il meccanismo dei ruoli riduce il lavoro per la gestione delle relazioni sui permessi e comporta una facilità di gestione anche da parte di personale non esperto del settore informatico.

Inoltre RBAC mette in atto alcuni principi tra quelli identificati per la sicurezza informatica: la "separazione dei compiti" (Separation of Duty, SoD) ed il "least privilege" [1]. Il principio della *separazione dei compiti* riguarda gli accessi che non dovrebbero essere concessi contemporaneamente ad un singolo individuo, mentre il *least privilege* stabilisce che gli utenti debbano possedere solo i permessi strettamente necessari per eseguire le proprie mansioni.

Classicamente l'assegnazione dei permessi avveniva in modo diretto, ossia per ogni utente si decideva quali permessi egli potesse avere; con l'introduzione di RBAC invece l'assegnazione dei permessi non è più diretta ma avviene in modo indiretto attraverso i ruoli. Quindi al posto della diretta relazione *utente-permesso* si creano altri due tipi di relazione: *utente-ruolo* e *ruolo-permesso* (Figura 2.2).

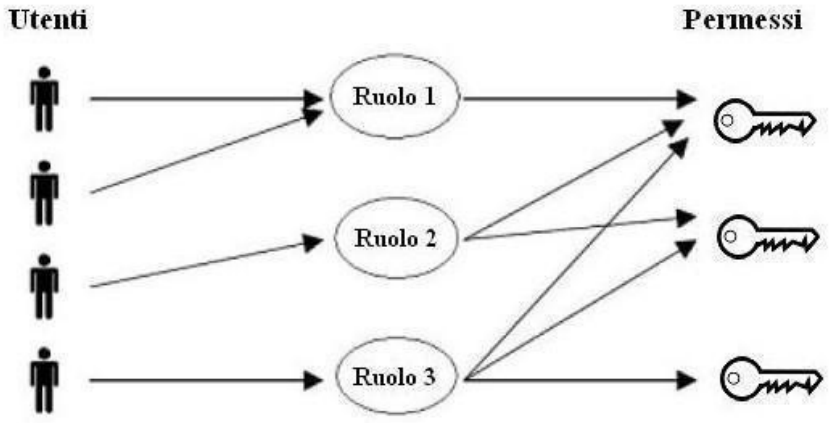

Figura 2.2: Relazioni utente-ruolo e ruolo-permesso

Veniamo alla rappresentazione matematica del modello.

Definizione 1. (Modello RBAC)
Il modello RBAC è costituito dai seguenti componenti: Siano $U = \{u_1, ..., u_m\}$

l'insieme degli utenti, $P = \{p_1, \ldots, p_n\}$ *l'insieme dei permessi e* $R = \{r_1, \ldots, r_k\}$ *l'insieme dei ruoli. Possiamo definire:*

- $UPA \subseteq U \times P$ *relazioni utente-permesso*
- $URA \subseteq U \times R$ *relazioni utente-ruolo*
- $RPA \subseteq R \times P$ *relazioni ruolo-permesso*

Essendo i ruoli il fulcro di questo modello, la procedura più difficoltosa durante la sua implementazione è proprio la corretta definizione dell'insieme di ruoli (*role engineering*).

2.2 Role Engineering

Il processo di *role engineering* consiste nella corretta identificazione dei ruoli secondo la struttura organizzativa dell'azienda [4]. Il modo più utilizzato per "disegnare" i ruoli parte dalle esistenti assegnazioni di permessi che gli utenti hanno già prima dell'adozione di RBAC. I ruoli vengono creati tramite l'aggregazione di questi permessi: il processo può essere per la maggior parte automatizzato, ed è detto *role mining*, in quanto possono essere sfruttare tecniche di *data mining* [2] (insieme di processi per l'estrazione di informazioni precedentemente sconosciute da una grande quantità di dati).

Molti algoritmi di role mining cercano di identificare un insieme di ruoli che copra tutte le preesistenti relazioni utente-permesso, nel tentativo di risolvere il *Role Mining Problem* [4].

2.3 Role Mining Problem

Il processo di trovare un completo e corretto insieme di ruoli è ritenuto uno dei lavori più importanti e impegnativi nell'implementazione di RBAC. La formalizzazione di questo problema è rappresentata dal *Role Mining Problem* (RMP): esso consiste nello scoprire un insieme di ruoli con cardinalità minima che copra tutti i permessi esistenti. Diamo delle definizioni preliminari per poi fornire la rappresentazione formale del problema.

Definizione 2. (Moltiplicazione tra matrici booleane)
La moltiplicazione tra due matrici booleane $A \in \{0,1\}^{m \times k}$ *e* $B \in \{0,1\}^{k \times n}$ *è il*

prodotto $A \otimes B = C$ *dove* $C \in \{0,1\}^{m \times n}$ *e l'operatore* $\otimes$ *è tale che per* $1 \leq i \leq m$ *e* $1 \leq j \leq n$ *si abbia*

$$c_{ij} = \bigvee_{h=1}^{k} (a_{ih} \wedge b_{hj})$$

Ricordiamo che nella definizione del modello RBAC, data nella sezione dedicata (Definizione 1)

- UPA rappresenta le relazioni utente-permesso
- URA rappresenta le relazioni utente-ruolo
- RPA rappresenta le relazioni ruolo-permesso

Definizione 3. (Matrici delle relazioni)
Date UPA *relazioni utente-permesso, definiamo* M(UPA) *la rappresentazione matriciale di* UPA*, ossia la matrice delle assegnazioni utente-permesso, che soddisfa*

$$\mathrm{M(UPA)}[i][j] = 1 \Leftrightarrow (u_i, p_j) \in \mathrm{UPA}$$

ossia se all'utente u_i *è assegnato il permesso* p_j*.*
Similmente definiamo M(URA) *e* M(RPA)*.*

Definizione 4. (Role Mining Problem)
Data la matrice booleana $m \times n$ *delle assegnazioni utente-permesso* M(UPA)*, il* Role Mining Problem *consiste nel trovare una matrice booleana* $m \times k$*,* M(URA)*, e una matrice booleana* $k \times n$*,* M(RPA)*, tale che il numero* k *sia minimizzato e*

$$\mathrm{M(UPA)} = \mathrm{M(URA)} \otimes \mathrm{M(RPA)} \tag{2.1}$$

Grazie a questa formalizzazione il problema del role engineering può essere facilmente mutato da un problema teorico di ricerca dei ruoli ad un problema matematico di fattorizzazione della matrice booleana M(UPA), soddisfacendo l'equazione (2.1). Osserviamo però che le varie risoluzioni di questo problema non porteranno sempre alla realizzazione di ruoli facilmente riconducibili a reali mansioni aziendali, ma potrebbero essere casuali aggregazioni di permessi. Uno dei modi possibili per attenuare questo problema è l'utilizzo dei vincoli, che tratteremo nel capitolo successivo.

Capitolo 3

Vincoli nel controllo degli accessi

In questo capitolo viene trattato il concetto di vincolo nel controllo degli accessi ed in particolare nel modello RBAC.

3.1 Perché utilizzare i vincoli

I vincoli consistono in regole che devono essere rispettate al fine di gestire correttamente gli accessi. In RBAC supportano una appropriata assegnazione di ruoli agli utenti, in base ai compiti che essi rivestono all'interno dell'organizzazione. Se un utente richiede un ruolo o un permesso, diversi rispetto a quelli già stabiliti nella configurazione del sistema, l'amministratore deciderà se concedere o meno tali privilegi secondo le regole che i vincoli impongono. Poiché essi vengono generati seguendo le politiche di sicurezza vigenti all'interno dell'organizzazione, rispettare i vincoli vuol dire far valere le politiche di sicurezza. Per esempio se un utente possiede i permessi necessari per la funzione "emettere ordine", nel caso in cui richieda i permessi necessari per svolgere la funzione "approvare ordine", che per motivi di sicurezza non possiede, dovrà esistere un vincolo che impedisca la concessione di questo ulteriore privilegio, al fine di evitare frodi.

I vincoli possono essere impiegati anche durante il processo di estrazione dei ruoli, integrandoli nel role mining. Spesso accade che i sistemi RBAC implementati non portino i benefici aspettati, a causa della non corretta definizione dei ruoli: sia quando si stabiliscono che quando si assegnano i ruoli, è necessario rispettare determinati principi di sicurezza [5] e sicuramente i vincoli aiutano in questa direzione.

3.2 Classificazione dei vincoli

I vincoli variano a seconda delle esigenze della singola organizzazione e possono essere applicati ai ruoli, ai permessi e agli utenti.

I più comuni sono i vincoli di separazioni dei compiti, usati per le politiche di conflitto di interesse [10], i quali portano all'impossibilità di assegnare contemporaneamente ruoli o permessi definiti *mutuamente esclusivi*, poiché permetterebbero agli utenti di compiere una operazione "pericolosa" [1]. Perciò potrebbe essere necessario evitare particolari combinazioni di assegnazioni di ruoli o permessi allo stesso utente o di permessi allo stesso ruolo, come anche la concessione di ruoli o permessi durante una stessa sessione di un utente. Per esempio un utente può avere la possibilità sia di richiedere che di approvare un ordine ma non sarà in grado di operare entrambe i compiti sullo stesso ordine.

Vi sono poi i vincoli di *prerequisito*: essi richiedono che un ruolo possa essere assegnato ad un utente solo se all'utente sono già stati assegnati i prerequisiti di tale ruolo; equivalentemente per i permessi.

Un altro tipo di vincoli sono i vincoli di *cardinalità*: essi possono essere assegnati per limitare per esempio il numero di utenti che possiedono uno stesso ruolo, il numero di ruoli che un utente può possedere, il numero di permessi in un ruolo o il numero di sessioni contemporanee in cui un permesso viene assegnato [11].

Le *eccezioni* potrebbero rappresentare un altro tipo di vincolo: esse sono delle varianti nelle assegnazioni dei permessi e vengono stabilite quando si ha la necessità di assegnare ad un utente un determinato ruolo ad eccezione di uno o più permessi contenuti in esso.

3.3 Estrazione dei vincoli

Come spiegato, i vincoli sono di fondamentale importanza per il controllo degli accessi. Nel modello RBAC, grazie ad essi, possiamo ottenere ruoli migliori o fare assegnazioni rimanendo in regola con le politiche di sicurezza dall'azienda. Non sempre però appare chiaro quali siano i vincoli sul controllo degli accessi e come sia possibile utilizzarli. Cercando di risolvere questo problema sono stati proposti algoritmi per l'estrazione dei vincoli, i quali partono dalle correnti assegnazioni dei permessi: ogni azienda, prima dell'implementazione del modello RBAC, già possiede assegnazioni di permessi regolate secondo altre strutture. Quindi proprio come il role mining cerca di estrarre i ruoli, in questi algoritmi si cerca di estrarre vincoli: si parla quindi di *constraints mining*. Nel capitolo successivo approfondiremo due lavori che trattano appunto l'estrazione dei vincoli.

Capitolo 4

Algoritmi di Constraints Mining

In questo capitolo descriveremo i lavori e i due algoritmi di constraints mining proposti rispettivamente nei testi [6] e [7].

4.1 Constraint-aware Role Mining via Extended Boolean Matrix Decomposition

In questo testo viene descritta la possibilità di dedurre vincoli dal processo di role mining. Viene ripresa la "Boolean Matrix Decomposition" (BMD [8]), che è alla base del *Role Mining Problem*, e viene estesa tramite "Extended Boolean Matrix Decomposition" (EBMD), una decomposizione che permette assegnazioni negative, meccanismo alla base del *Constraint-aware Role Mining Problem*: un permesso o un ruolo assegnato negativamente ad un utente, prevengono l'utente dal possedere quel permesso o i permessi di quel ruolo.

Vincoli

Grazie alle assegnazioni negative di permessi o di ruoli, indicate con elementi -1, è possibile scoprire vincoli sottostanti. I vincoli considerati in [6] sono le "eccezioni" e la "separazione dei compiti".

Consideriamo l'esempio della matrice M(UPA) di Figura 4.1. Una soluzione ottimale del convenzionale *Role Mining Problem*, minimizzando il numero di ruoli richiesti, è mostrata in Figura 4.2. Ammettendo invece l'uso di assegnazioni negative, la soluzione ottimale diventa quella illustrata in Figura 4.3. Notiamo come con le assegnazioni negative di permessi, abbiamo bisogno solamente di due ruoli per ricostruire la stessa matrice di partenza. Inoltre è possibile ricavare ulteriori informazioni: vediamo che se il ruolo $r_2 : (0, 1, -1, 1)$ è assegnato ad un utente,

	p_1	p_2	p_3	p_4
u_1	1	0	1	1
u_2	1	0	1	1
u_3	1	1	0	1
u_4	0	1	0	1

Figura 4.1: Matrice M(UPA) di esempio

	r_1	r_2	r_3
u_1	1	0	1
u_2	1	0	1
u_3	1	1	0
u_4	0	1	0

M(URA)

	p_1	p_2	p_3	p_4
r_1	1	0	0	1
r_2	0	1	0	1
r_3	0	0	1	0

M(RPA)

Figura 4.2: Scomposizione nel role mining convenzionale

	r_1	r_2
u_1	1	0
u_2	1	0
u_3	1	1
u_4	0	1

M(URA)

	p_1	p_2	p_3	p_4
r_1	1	0	1	1
r_2	0	1	-1	1

M(RPA)

Figura 4.3: Scomposizione nel Constraint-aware Role Mining, con assegnazioni negative

egli non potrà mai avere il permesso p_3. Questo implica che p_3 potrebbe essere esclusivo da p_2 e p_4. Da $r_1 : (1, 0, 1, 1)$ deduciamo che p_3 e p_4 possono coesistere in un ruolo, quindi l'unica spiegazione è che c'è un vincolo di separazione dei compiti su p_2 e p_3. Se invece avessimo elementi -1 nella prima matrice di scomposizione, potremmo ricavare vincoli di eccezione.

Per poter effettuare questo tipo di scomposizione che ammette assegnazioni negative, è necessario introdurre un nuovo approccio, l'Extended Boolean Matrix Decomposition (EBMD).

Extended Boolean Matrix Decomposition

La EBMD consiste nello scomporre una matrice booleana in due matrici, una booleana e una matrice in $\{-1, 0, 1\}$. Essa fornisce due vantaggi: la riduzione del numero di ruoli necessari e la possibilità di dedurre i vincoli dalla configurazione finale di assegnazioni che l'algoritmo genera. Dal punto di vista matematico, il *Constraint-aware Role Mining Problem* consiste semplicemente nel tro-

vare una buona scomposizione EBMD della matrice booleana corrispondente alle assegnazioni utente-permesso.

Ricordando la definizione di moltiplicazione tra matrici booleane, introdotta nella sezione sul Role Mining Problem, definiamo la BMD e la EBMD.

Definizione 5. (Scomposizione di matrici booleane, BMD)
Se $A = B \otimes C$, dove A, B e C sono matrici booleane, $B \otimes C$ è detta scomposizione di A.

Definizione 6. (EBMD)
Data la matrice booleana $A \in \{0,1\}^{m \times n}$, una soluzione EBMD è una scomposizione denotata da $A = B \odot C$ con $B \in \{-1,0,1\}^{m \times k}$ e $C \in \{0,1\}^{k \times n}$, dove $A_i = \cup_{b_{ij}=1} C_j \setminus \cup_{b_{ij}=-1} C_j$, essendo A_i l'i-esima riga di A e C_j la j-esima riga di C.

Dove l'operatore $\odot$ è tale che:

Definizione 7. (Operatore $\odot$)
L'operatore $\odot$ opera su una matrice $B \in \{-1,0,1\}^{m \times k}$ e una matrice $C \in \{0,1\}^{k \times n}$. Se $A = B \odot C$, si ha:

$$\begin{cases} a_{ij} = 1 & se (\exists t_1)(c_{i,t_1} = 1 \wedge b_{t_1,j} = 1) \wedge (\nexists t_2)(c_{i,t_2} = 1 \wedge b_{t_2,j} = -1) \\ a_{ij} = 0 & se (\nexists t_1)(c_{i,t_1} = 1 \wedge b_{t_1,j} = 1) \vee (\exists t_2)(c_{i,t_2} = 1 \wedge b_{t_2,j} = -1) \end{cases}$$

dove $1 \leq i \leq m$ e $1 \leq j \leq n$

Notiamo che gli operatori $\odot$ e $\otimes$ sono equivalenti quando tutte le entrate di B sono in $\{0,1\}$.

L'operatore $\odot$ possiede la proprietà commutativa, descritta come segue:

Proprietà 1. (Commutatività)

$$(B \odot C)^T = C^T \odot B^T$$

La commutatività implica che se $A = C \odot B$, dove C è in $\{-1,0,1\}$ e B è in $\{0,1\}$ vale anche che $A^T = B^T \odot C^T$, cioè l'ordine delle due matrici non ha importanza: la matrice di scomposizione in $\{-1,0,1\}$ può essere sia quella di destra (se si desiderano assegnazioni negative di permessi per vincoli SoD) che quella di sinistra (se si desiderano assegnazioni negative di ruoli per vincoli di eccezione).

Problemi

Analizziamo i problemi che l'algoritmo e le sue varianti, dovranno risolvere. Diremo "ruoli ricchi" o "assegnazioni di ruoli ricche" per indicare i casi in cui sono presenti assegnazioni negative.

Problema 1. (Constraint-aware Role Mining, CRM)
Data la matrice delle assegnazioni utente-permesso $M(UPA)_{m\times n}$ e un numero positivo k, scoprire un insieme di ruoli ricchi (o di ruoli regolari) ed identificare le corrispondenti assegnazioni di ruoli regolari (o le assegnazioni di ruoli ricche), tali che $M(URA)_{m\times k}$ e $M(RPA)_{k\times n}$ riducono al minimo gli errori di ricostruzione.

Il CRM può essere descritto dal seguente problema di ottimizzazione:

Problema 2. (Optimization CRM)
Data $M(UPA)_{m\times n}$, trovare $M(URA)_{m\times k}$ e $M(RPA)_{k\times n}$ tali che si minimizzi

$$\|M(UPA)_{m\times n} - M(URA)_{m\times k} \odot M(RPA)_{k\times n}\|_1$$

dove $\|\cdot\|_1$ è usata per misurare la dissimilarità tra due matrici booleane o in $\{-1, 0, 1\}$:

$$\|X_{m\times n}\|_1 = \sum_{i=1}^{m}\sum_{j=1}^{n}|x_{ij}|$$

Può accadere che, nel ricostruire poi la matrice di partenza, compaiano errori di ricostruzione, i quali possono essere di due tipi: sovra-assegnazioni e sotto-assegnazioni. Nel caso di sotto-assegnazioni, l'utente può sempre chiedere all'amministratore di correggere gli errori, mentre le sovra-assegnazioni potrebbero causare seri problemi di sicurezza. Per evitare le sovra-assegnazioni viene proposta una variante del problema presentato.

Problema 3. (Conservative CRM)
Data $M(UPA)_{m\times n}$, trovare $M(URA)_{m\times k}$ e $M(RPA)_{k\times n}$ tali che si minimizzi

$$\|M(UPA)_{m\times n} - M(URA)_{m\times k} \odot M(RPA)_{k\times n}\|_1$$

con

$$\Big(M(URA) \odot M(RPA)\Big)_{ij} = 0 \; se \; M(UPA)_{ij} = 0$$

Notiamo che sia il CRM che la sua versione "conservativa" devono determinare le due matrici M(URA) e M(RPA): sicuramente è difficile determinare due matrici che variano contemporaneamente, mentre è relativamente più semplice determinare una delle due quando l'altra è stata fissata. Viene quindi proposto un algoritmo di *minimizzazione alternata* per il problema *CRM*, descritto nell'Algoritmo 1.

Ogni iterazione dell'algoritmo è una procedura di due fasi: il problema CRM viene quindi scomposto in due sotto-problemi Partial CRM I e Partial CRM II, con le relative versioni "conservative". Tutti questi problemi vengono risolti con

Algoritmo 1 MINIMIZZAZIONE ALTERNATA PER IL CRM

Input: $A \in \{0,1\}^{m \times n}$
Output: $B \in \{-1,0,1\}^{m \times k}$ e $C \in \{0,1\}^{k \times n}$
1: Definire un valore iniziale di $\{B^0, C^0\}$
2: $B^{curr} = B^0,\ C^{curr} = C^0$
3: $B^{next} = \arg\min_B \|A - B \odot C^{curr}\|_1$
4: $C^{next} = \arg\min_C \|A - B^{curr} \odot C\|_1$
5: **fintanto che** $(B^{curr} \neq B^{next})\ ||(C^{curr} \neq C^{next})$ **ripeti**
6: $B^{curr} = B^{next},\ C^{curr} = C^{next}$
7: $B^{next} = \arg\min_B \|A - B \odot C^{curr}\|_1$
8: $C^{next} = \arg\min_C \|A - B^{curr} \odot C\|_1$
9: **fine-ciclo**

algoritmi euristici di tipo *greedy* che consistono in varianti degli algoritmi risolutivi del problema RBSC (*Red-Blue Set Cover Problem*).

Considerazioni

Notiamo che nel lavoro approfondito in questa sezione [6], l'estrazione di vincoli non viene eseguita completamente: con l'estrazione dei ruoli, si genera una configurazione da cui è possibile dedurre alcuni vincoli, ma è necessaria una ulteriore analisi per la loro definitiva generazione. La deduzione dei vincoli parte dagli elementi -1 che l'algoritmo pone nelle matrici di scomposizione. Manca però una tecnica automatizzata che, a partire da questi elementi, deduca le regole vere e proprie.

Nel secondo articolo che abbiamo analizzato invece, l'algoritmo proposto porta effettivamente all'estrazione di regole complete.

4.2 Mining Constraints in RBAC

In questo lavoro vengono inizialmente definiti una serie di vincoli, tra cui ritroviamo le regole di anti-associazione, e successivamente viene proposto un algoritmo che si occupa di estrarre queste regole.

Vincoli

I vincoli considerati in [7] sono ruoli mutuamente esclusivi, permessi mutuamente esclusivi, cardinalità sui ruoli, sugli utenti e sui permessi, anti-associazione tra permessi e tra ruoli. Ci concentreremo sulle regole di anti-associazione tra permessi.

Una regola di questo tipo sarà valida se possiederà *supporto* e *confidenza* minimi (rispettivamente *minsup* e *minconf*): Per regole del tipo $X \Rightarrow \overline{Y}$ si denota

- La confidenza come il rapporto tra $numUsers(X\overline{Y})$ e $numUsers(X)$ (dove $numUsers(X\overline{Y})$ è il numero di utenti che possiedono l'insieme di permessi X e non possiedono l'insieme di permessi Y; $numUsers(X)$ è il numero di utenti che possiedono l'insieme di permessi X)

- Il supporto come il rapporto tra $numUsers(X\overline{Y})$ e $numUsers(All)$ (dove $numUsers(All)$ è il numero totale di utenti)

Similmente per $\overline{X} \Rightarrow Y$ e $\overline{X} \Rightarrow \overline{Y}$.

Algoritmo

L'algoritmo proposto genera regole di anti-associazione (vincoli di mutua esclusione): esso è diviso in tre fasi e l'Algoritmo 2 ne fornisce lo pseudo-codice.

Fase 1: Generazione degli insiemi di permessi 1-frequenti
Dalla la scansione della matrice delle assegnazioni utente-permesso, M, si ricavano i permessi 1-frequenti p_i; dalla la scansione della matrice $\overline{M}$, costruita come opposta di M, si ricavano i permessi 1-frequenti $\overline{p}_i$. Si denota con F_1 l'insieme dei permessi 1-frequenti p_i e con $\overline{F}_1$ l'insieme dei permessi 1-frequenti $\overline{p}_i$.

Fase 2: Generazione degli insiemi di permessi k-frequenti
Si itera sull'insieme di tutti gli insiemi di permessi (k-1)-frequenti, F_{k-1}, per generare l'insieme di permessi k-frequente F_k, e sull'insieme di tutti gli insiemi di permessi (k-1)-frequenti, $\overline{F}_{k-1}$, per generare l'insieme di permessi k-frequente $\overline{F}_k$.

Fase 3: Generazione delle regole di anti-associazione
Usando F ed $\overline{F}$ si generano regole di anti-associazione: se per esempio $\{p_1, p_2\}$ e $\{\overline{p}_3, \overline{p}_4\}$ sono insiemi di permessi 2-frequenti, la regola di anti-associazione $p_1p_2 \Rightarrow \overline{p}_3\overline{p}_4$ vale se il rapporto $\mathrm{r} = \frac{\mathrm{supporto(p_1p_2\overline{p}_3\overline{p}_4)}}{\mathrm{supporto(p_1p_2)}} > minconf$.

Considerazioni

Questo approccio mira specificamente all'estrazione dei vincoli, tuttavia notiamo che la bontà di una regola è stimata solamente in base alla confidenza, rischiando la generazione di regole che sono in realtà casuali. Vogliamo quindi proporre un nuovo metodo di stima delle regole estratte: a tal fine utilizzeremo il concetto di "varianza" che verrà argomentato nei capitoli seguenti.

Algoritmo 2

Input: m, n, $M_{m\times n}$, minsup, minconf, F, $\overline{F}$
Output: Regole di anti-associazione tra permessi

1: {*Generazione degli insiemi di permessi 1-frequenti*}
2: **per** $(i = 1; i < n, i++)$ **ripeti**
3: $\mathrm{sf}(\mathrm{p_i}) = \mathrm{numUsers}(\mathrm{p_i})/\mathrm{m}, \mathrm{sf}(\overline{\mathrm{p}}_\mathrm{i}) = \mathrm{numUsers}(\overline{\mathrm{p}}_\mathrm{i})/\mathrm{m}$
4: inserire $\big(\mathrm{p_i};\ \mathrm{sf}(\mathrm{p_i});\ \mathrm{perm\_users}(\mathrm{p_i})\big)$ in F_1 se $\mathrm{sf}(\mathrm{p_i}) \geq \mathrm{minsup}$
5: inserire $\big(\overline{\mathrm{p}}_\mathrm{i};\ \mathrm{sf}(\overline{\mathrm{p}}_\mathrm{i});\ \mathrm{perm\_users}(\overline{\mathrm{p}}_\mathrm{i})\big)$ in $\overline{\mathrm{F}}_1$ se $\mathrm{sf}(\overline{\mathrm{p}}_\mathrm{i}) \geq \mathrm{minsup}$
6: **fine-ciclo**
7: {*Generazione degli insiemi di permessi k-frequenti*}
8: **per** $(k = 2; F_k \neq 0; k++)$ **ripeti**
9: $\mathrm{F_k} = \mathrm{FrequentPermissionGen}(\mathrm{F_{k-1}}, \mathrm{minsup})$
10: **fine-ciclo**
11: **per** $(k = 2; \overline{F}_k \neq 0; k++)$ **ripeti**
12: $\overline{\mathrm{F}}_\mathrm{k} = \mathrm{FrequentPermissionGen}(\overline{\mathrm{F}}_\mathrm{k-1}, \mathrm{minsup})$
13: **fine-ciclo**
14: $\mathrm{F} = \cup_\mathrm{k}\mathrm{F_k}, \overline{\mathrm{F}} = \cup_\mathrm{k}\overline{\mathrm{F}}_\mathrm{k}$
15: {*FrequentPermissionGen* $(\mathrm{T_{k-1}}, \mathrm{minsup})$}
16: **per** (ogni X e ogni Y in $\mathrm{T_{k-1}}$) **ripeti**
17: **se** i primi k-2 permessi di X e Y sono gli stessi **allora**
18: $\mathrm{perm\_users}(\mathrm{X}\cup\mathrm{Y}) = \mathrm{perm\_users}(\mathrm{X}) \cap \mathrm{perm\_users}(\mathrm{Y})$
19: $\mathrm{sf}(\mathrm{X}\cup\mathrm{Y}) = \mathrm{numUsers}(\mathrm{perm\_users}(\mathrm{X}\cup\mathrm{Y}))/\mathrm{m}$
20: inserire $\big(\mathrm{X}\cup\mathrm{Y};\ \mathrm{sf}(\mathrm{X}\cup\mathrm{Y});\ \mathrm{perm\_users}(\mathrm{X}\cup\mathrm{Y})\big)$ in $\mathrm{T_k}$ se $\mathrm{sf}(\mathrm{X}\cup\mathrm{Y}) \geq \mathrm{minsup}$
21: **fine-condizione**
22: **fine-ciclo**
23: Restituire $\mathrm{T_k}$
24: {*Generazione delle regole di anti-associazione*}
25: **per** (ogni Y in $\overline{\mathrm{F}}$) **ripeti**
26: Genera le regole usando lo stesso metodo di Apriori
27: **fine-ciclo**
28: **per** (ogni X in F e ogni Y in $\overline{\mathrm{F}}$) **ripeti**
29: Genera le regole usando lo stesso metodo di Apriori
30: **fine-ciclo**

Capitolo 5

Indice di varianza

In questo capitolo introdurremo uno specifico "indice di varianza", studiato e sviluppato negli anni dagli autori di [13], [14] e [15], lavorando nel campo della sicurezza informatica e del *data-privacy*. Questo strumento sarà per noi fondamentale per l'applicazione sul constraints mining, e l'introduzione della nuova metrica di stima dei vincoli.

5.1 Data-privacy e varianza

Il data-privacy consiste nella protezione dei dati al fine di preservare informazioni sensibili degli individui a cui essi appartengono. La protezione è richiesta soprattutto nel caso in cui è necessaria la loro pubblicazione, per esempio in campo medico le informazioni sui pazienti sono fondamentali per lo sviluppo delle ricerche future.

Il problema è che spesso si ha a che fare con dati che non sono sotto forma di numeri, bensì di parole (*attributi nominali*): l'anonimizzazione di attributi nominali è continuo oggetto di ricerca. Lo scopo è quello di trasformare nomi in numeri, per poter applicare note tecniche di anonimizzazione già sviluppate sui numeri, cercando sempre di preservare il significato incorporato nei dati. A tal scopo in [15] viene proposto un metodo, per "mappare" gli attributi nominali in attributi numerici, che tiene in considerazione la semantica sottostante. Questo metodo porta alla definizione e al calcolo sugli attributi nominali di una *varianza* semanticamente significativa, detta "varianza basata sulla marginalità": la funzione che permette la mappatura sopra descritta è il calcolo della marginalità, e su di essa si basa anche il calcolo dell'indice di varianza.

Questo concetto di varianza può essere utilizzato per migliorare il criterio di estrazione di regole di anti-associazione: calcolando l'indice sulle categorie a cui gli

utenti appartengono si ottengono informazioni sull'omogeneità dei gruppi di utenti che sono coinvolti nella regola estratta.

5.2 Marginalità e distanza

Per poter descrivere la funzione numerica che mappa valori nominali, è necessario definire alcuni concetti. Nel seguito utilizzeremo il termine *tassonomia* nel senso matematico di struttura ad albero di categorie appartenenti ad un dato gruppo di concetti. Le gerarchie possono essere rappresentate tramite grafi in cui le relazioni gerarchiche sono i collegamenti tra i nodi, ossia gli spigoli, i quali corrispondono a concetti: si veda la Figura 5.1. In [15] viene proposta una *distanza semantica*

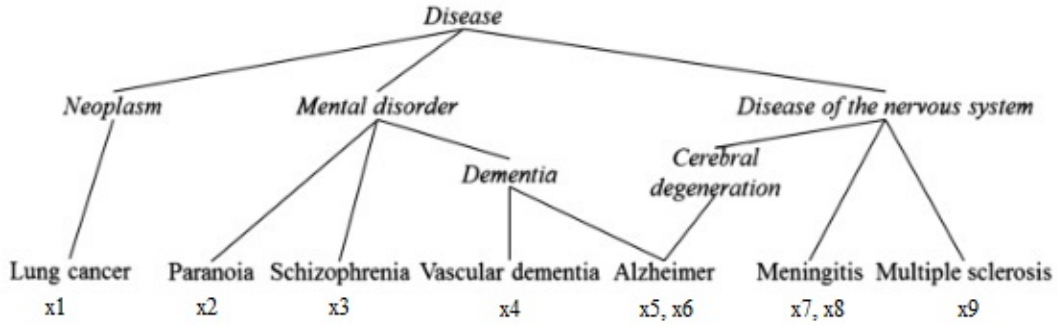

Figura 5.1: Esempio di tassonomia per un campione di attributo "Diagnosi"

che cattura il significato dei valori nominali, basandosi sulla struttura gerarchica a cui essi appartengono. La distanza tra due elementi x_1 e x_2 della gerarchia viene misurata tramite una funzione del numero dei loro antenati non comuni diviso (per normalizzare) dal numero totale di antenati.

Definizione 8. (Distanza semantica)
Dati due elementi della gerarchia x_1 e x_2, la distanza tra loro è definita come:

$$d(x_1, x_2) = \log_2 \left(1 + \frac{|T(x_1) \cup T(x_2)| - |T(x_1) \cap T(x_2)|}{|T(x_1) \cup T(x_2)|} \right)$$

dove $T(x_i)$ è l'insieme degli antenati di x_i, incluso se stesso, e $|\cdot|$ indica la cardinalità *dell'insieme.*

Notiamo che

$$d(x_1, x_2) = \log_2 \left(2 - \frac{|T(x_1) \cap T(x_2)|}{|T(x_1) \cup T(x_2)|} \right)$$

dove il rapporto

$$\frac{|T(x_1) \cap T(x_2)|}{|T(x_1) \cup T(x_2)|}$$

è conosciuto come "Indice di similarità di Jaccard" e

$$d(x_1, x_2) \in \big[\log_2(1), \log_2(2)\big) = [0, 1)$$

in cui 0 corrisponde al caso in cui x_1 e x_2 hanno tutti gli antenati in comune -quando x_1 e x_2 sono due elementi dello stesso attributo- e 1 corrisponde al caso in cui tutti gli antenati sono distinti -non raggiunto in quanto almeno la radice è un antenato sempre in comune-.

Questa distanza restituisce valori normalizzati in $[0, 1]$ che possono essere confrontati in modo coerente con le distanze ottenute su altre gerarchie; inoltre è stato dimostrato [16] che essa soddisfa non-negatività, riflessività, simmetria e sub-additività, ossia è una distanza in senso matematico. Ricordiamo infatti che la distanza è definita nel modo seguente:

Definizione 9. (Distanza)
Una distanza su un insieme X è una qualsiasi funzione

$$d : X \times X \longrightarrow \mathbb{R}$$

che soddisfa le seguenti proprietà per ogni scelta di x, y, z in X:

1. $d(x, y) \geq 0$ *(non-negatività)*
2. $d(x, y) = 0 \iff x = y$ *(riflessività)*
3. $d(x, y) = d(y, x)$ *(simmetria)*
4. $d(x, y) \leq d(x, z) + d(z, y)$ *(sub-additività)*

L'obiettivo della funzione di mappatura è quello di associare, ad ogni valore nominale, un numero detto *marginalità*, il quale ingloba le caratteristiche sia semantiche che distributive del valore. Consideriamo un attributo nominale X i cui valori sono classificati in una tassonomia. Sia T_X un campione (inteso come gruppo di dati rilevati) di valori di X.

Definizione 10. (Marginalità)
La marginalità $m(\cdot)$ di ogni valore x_j in T_X è calcolata come

$$m(x_j) = \sum_{x_i \in T_X \setminus \{x_j\}} d(x_j, x_i)$$

dove $d(\cdot,\cdot)$ è la distanza sopra definita.

La marginalità può essere intesa come una misura del valore di centralità all'interno di una gerarchia/tassonomia, cioè ha la funzione di determinare quale sia il *centro* della gerarchia e quanto ogni valore nominale si trovi distante da quel centro. Maggiore è il valore di $m(x_j)$ e più marginale (ossia meno centrale) è x_j. La marginalità tiene in considerazione la posizione, all'interno della gerarchia, della categoria di cui il valore fa parte ed anche la frequenza con cui esso compare: un valore che appartiene ad una categoria esterna (cioè distante dal centro) diventa sempre più centrale se la sua frequenza cresce.

Vediamo ora un esempio.

Esempio 1. Assumiamo di avere un attributo nominale "Diagnosi" per cui è disponibile un campione i cui valori sono classificati come nella Figura 5.1. Il campione è costituito da un elemento per ogni categoria di diagnosi, eccetto per la categoria "Alzhheimer" e "Meningitis", per ognuna delle quali vi sono due elementi. Di seguito riportiamo la matrice delle distanze tra gli elementi (Figura 5.2), in cui l'entrata (j, i) rappresenta la distanza semantica $d(x_j, x_i)$ come precedentemente definita.

	x_1	x2	x3	x4	x5	x6	x7	x8	x9
x_1	0	0,85	0,85	0,87	0,91	0,91	0,85	0,85	0,85
x2	0,85	0	0,58	0,68	0,78	0,78	0,85	0,85	0,85
x3	0,85	0,58	0	0,68	0,78	0,78	0,85	0,85	0,85
x4	0,87	0,68	0,68	0	0,65	0,65	0,87	0,87	0,87
x5	0,91	0,78	0,78	0,65	0	0	0,65	0,65	0,65
x6	0,91	0,78	0,78	0,65	0	0	0,65	0,65	0,65
x7	0,85	0,85	0,85	0,87	0,65	0,65	0	0	0,58
x8	0,85	0,85	0,85	0,87	0,65	0,65	0	0	0,58
x9	0,85	0,85	0,85	0,87	0,65	0,65	0,58	0,58	0

Figura 5.2: Matrice delle distanze per la gerarchia della Figura 5.1

La marginalità $m(x_j)$ dell'elemento x_j può essere ottenuta sommando tutte le distanze della *j-esima* riga (o colonna) di questa matrice. Nella Figura 5.3 sono state riportate le marginalità per tutti gli elementi. Possiamo notare che gli elementi x_5 e x_6 sono gli elementi meno marginali (per posizione e quantità), quindi costituiscono il *centro* della gerarchia, mentre l'elemento x_1 è l'elemento più marginale, ossia quello più distante dal centro: entrambe queste considerazioni numeriche sono coerenti con quanto si nota visivamente dalla tassonomia della Figura 5.1.

x_j	$m(x_j)$
x_1	0 + 0,85 + 0,85 + 0,87 + 0,91 + 0,91 + 0,85 + 0,85 + 0,85 = 6,94
x_2	0,85 + 0 + 0,58 + 0,68 + 0,78 + 0,78 + 0,85 + 0,85 + 0,85 = 6,22
x_3	0,85 + 0,58 + 0 + 0,68 + 0,78 + 0,78 + 0,85 + 0,85 + 0,85 = 6,22
x_4	0,87 + 0,68 + 0,68 + 0 + 0,65 + 0,65 + 0,87 + 0,87 + 0,87 = 6,14
x_5	0,91 + 0,78 + 0,78 + 0,65 + 0 + 0 + 0,65 + 0,65 + 0,65 = 5,07
x_6	0,91 + 0,78 + 0,78 + 0,65 + 0 + 0 + 0,65 + 0,65 + 0,65 = 5,07
x_7	0,85 + 0,85 + 0,85 + 0,87 + 0,65 + 0,65 + 0 + 0 + 0,58 = 5,3
x_8	0,85 + 0,85 + 0,85 + 0,87 + 0,65 + 0,65 + 0 + 0 + 0,58 = 5,3
x_9	0,85 + 0,85 + 0,85 + 0,87 + 0,65 + 0,65 + 0,58 + 0,58 + 0 = 5,88

Figura 5.3: Marginalità degli elementi della gerarchia della Figura 5.1

5.3 Indice di varianza

Abbiamo ora gli strumenti per poter definire l'"indice di varianza". L'idea intuitiva dietro al concetto di *varianza* in una tassonomia è che un campione di valori nominali appartenente a categorie figlie della stessa categoria superiore hanno una una varianza più piccola rispetto ad un campione con figli di categorie superiori che sono tra loro diverse. La marginalità media di un campione si rivela essere proprio questo concetto di varianza.

Definizione 11. (Varianza semantica o "basata sulla marginalità")
Dato un campione T_X estratto da un attributo nominale X, la varianza del campione è definita come

$$Var(T_X) = \frac{\sum_{x_j \in T_X} m(x_j)}{|T_X|}$$

Sarebbe praticamente la media di quanto ogni elemento è distante dagli altri. Vediamo ora un esempio di calcolo della varianza.

Esempio 2. Dalla Figura 5.3 possiamo calcolare la varianza del campione della gerarchia di Figura 5.1 nel modo seguente:

$$\frac{6,94 + 6,22 + 6,22 + 6,14 + 5,07 + 5,07 + 5,3 + 5,3 + 5,88}{9} = 5,79$$

Questo è l'indice di varianza di base da cui è partito il nostro studio. Nel seguente capitolo analizzeremo meglio questo indice per poter poi operare opportune modifiche.

Capitolo 6

Proposta di applicazione della varianza per la valutazione dei vincoli

In questo capitolo vedremo come applicare la varianza per la valutazione delle regole di anti-associazione estratte dall'algoritmo di constraints mining presentato nel Capitolo 4 - Sezione 2. Illustreremo piccole modifiche dell'indice di *varianza* e di quello di *distanza* precedentemente definiti, al fine di adattarli maggiormente alle nostre esigenze. Introduciamo successivamente un indice da noi sviluppato, il quale unisce i concetti di confidenza e varianza, per poter stimare le regole in modo più completo. Presentiamo infine un esempio di applicazione di questo indice di "ranking", per mostrare l'utilità pratica della varianza.

6.1 Definizione di distanza fra utenti

In questa sezione faremo delle osservazioni riguardo la formula definita nel capitolo precedete per il calcolo della distanza (Definizione 8):

Osservazione 1. Abbiamo già osservato che, per il modo in cui è definita, tale formula di distanza non avrà mai valore 1 in quanto esiste sempre almeno un elemento in comune, la radice. Essendo la radice sempre presente, e quindi irrilevante nei calcoli, possiamo escluderla dal conteggio. In questo modo la distanza potrà raggiungere il valore 1.

Osservazione 2. Sempre riguardo la formula della distanza, abbiamo precedentemente osservato, nel Capitolo 5 - Sezione 2, che essa rappresenta una applicazione dell'*indice di Jaccard* [17]. In questo capitolo proponiamo la sua sostituzione con

una versione "pesata". Se x_1 e x_2 sono due elementi del campione e se consideriamo $X = T(x_1)$ e $Y = T(x_2)$, dove $T(x_i)$ è linsieme degli antenati di x_i, possiamo scrivere l'indice di Jaccard come

$$J(X,Y) = \frac{|X \cap Y|}{|X \cup Y|} \tag{6.1}$$

Vogliamo ora definire la sua versione pesata. Intanto esprimiamo il precedente rapporto in un'altra forma:

$$J(X,Y) = \frac{\sum_{i=1}^{n} \min(X_i, Y_i)}{\sum_{i=1}^{n} \max(X_i, Y_i)} \tag{6.2}$$

in cui supponiamo che X e Y siano due vettori di lunghezza n (dove n è il numero totale dei nodi dell'albero inclusi nei percorsi dalla radice a x_1 e x_2) e X_i ed Y_i corrispondono ad elementi 0 o 1 a seconda che nel percorso rispettivamente fino ad x_1 e x_2 sia presente o meno il nodo i-esimo. La formula 6.1 corrisponde esattamente alla formula 6.2.

Se ora al posto degli elementi 1 utilizziamo valori in $[1, +\infty)$ come pesi, otteniamo l'*indice di Jaccard pesato*. Per i pesi utilizziamo numeri naturali, partendo da 1 per i nodi al livello più profondo e aumentando di uno da un livello al suo superiore, mano a mano che si risale l'albero. Nell'esempio sotto illustreremo questo meccanismo. Quindi avremo la seguente nuova formula per la distanza:

$$d(x_1, x_2) = \log_2 \left(2 - \frac{\sum_{i=1}^{n} \min(X_i, Y_i)}{\sum_{i=1}^{n} \max(X_i, Y_i)} \right)$$

dove X_i ed Y_i corrispondono ai pesi associati ai nodi nel percorso rispettivamente fino ad x_1 e x_2, o a 0 se il nodo non è presente nel percorso. La scelta di associare pesi maggiori ai livelli più alti dell'albero comporta il dare più importanza al fatto che una coppia di elementi abbia un determinato numero di nodi in comune e meno al fatto che abbia anche una quantità di nodi distinti, che per noi è meno importante. Vediamo meglio questo concetto: in un caso di due gerarchie in cui, nei rispettivi percorsi dalla radice a x_1 e x_2, la seconda gerarchia ha un maggior numero di nodi distinti, rispetto alla prima, ma anche un maggior numero di nodi in comune, possiamo dire che nella seconda la "separazione dei percorsi" avviene ad un livello più basso: in questa situazione vogliamo far pesare maggiormente il fatto che nella seconda ci siano più nodi in comune rispetto al fatto che ci siano anche più nodi distinti. Pesi più bassi a livelli più bassi influiscono meno in termini di "peso della distanza", e viceversa pesi più alti verso la radice.

Chiariamo con un esempio: facciamo riferimento alla Figura 6.1, in cui J indica l'indice di Jaccard e J_p indica l'indice di Jaccard pesato. La radice, R, è esclusa dal conteggio. In entrambi i casi, l'indice di Jaccard classico risulta essere $\frac{1}{3}$, ma nel secondo caso la "separazione dei percorsi" avviene ad un livello più basso, quindi ritroviamo un maggior numero di nodi in comune rispetto al primo caso, per questo vogliamo un indice di Jaccard maggiore, nonostante ci sia anche un maggior numero di nodi distinti. Utilizzando pesi maggiori a livelli più alti otteniamo proprio questo: si può notare come l'indice J_p nel secondo caso aumenta rispetto al primo.

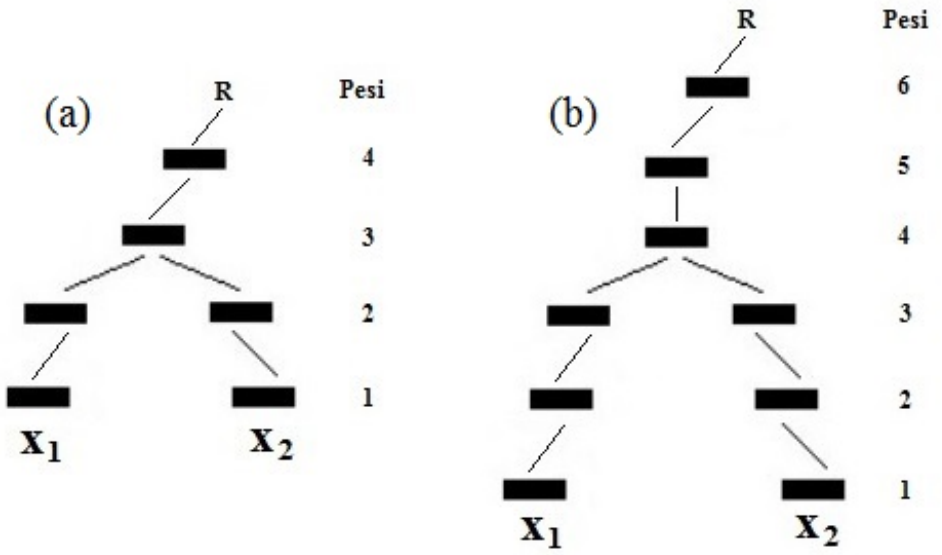

Figura 6.1: (a) $J = \frac{1}{3}$ $J_p \approx 0,5$ (b) $J = \frac{1}{3}$ $J_p \approx 0,6$

6.2 Una nuova definizione di varianza

La formula della varianza definita in [15] viene utilizzata dagli autori su insiemi di elementi tutti della stessa cardinalità. Paragonando invece indici di varianza corrispondenti a due campioni di diversa cardinalità, i valori non risultano confrontabili.

Chiariamo con un esempio: vediamo le Figure 6.2 e 6.3, che raffigurano campioni di cardinalità rispettivamente 4 e 9. Nei rispettivi Casi (b) la varianza è giustamente più bassa rispetto ai Casi (a) perché il campione è più "concentrato" ossia più omogeneo, in quanto distribuito su attributi con stesso padre, mentre nei Casi (a) è distribuito su attributi non tutti figli dello stesso padre. Fin qui è tutto coerente. Se però confrontiamo i due esempi tra loro (si vedano per esempio il Caso (a) della Figura 6.2 e Caso (b) della Figura 6.3), intuitivamente la varianza

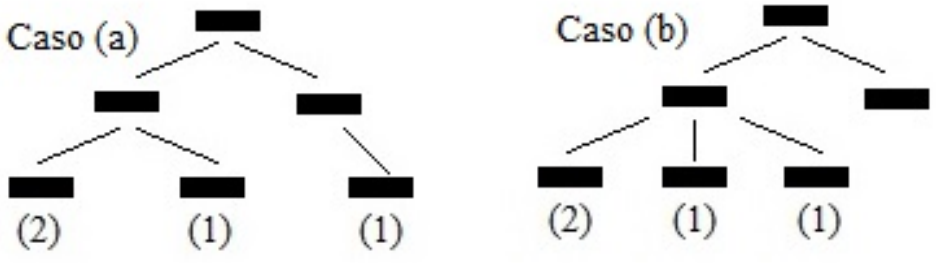

Figura 6.2: Caso (a): $Var(T_X) = 1,86$; Caso (b): $Var(T_X) = 1,45$

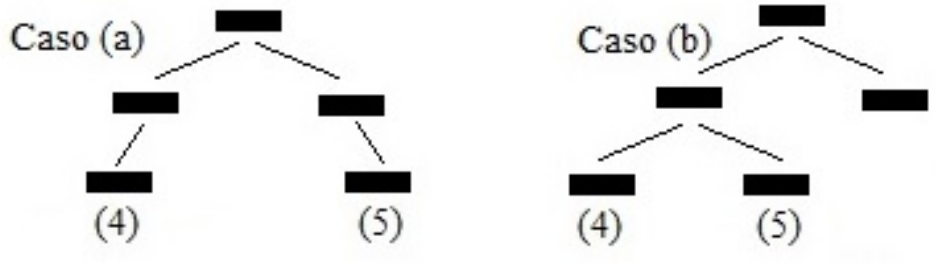

Figura 6.3: Caso (a): $Var(T_X) = 3,78$; Caso (b): $Var(T_X) = 2,58$

dovrebbe essere minore nel secondo caso in quanto il campione è distribuito su due attributi con lo stesso padre, mentre nell'altro è distribuito su tre attributi non tutti figli dello stesso padre; eppure il valore della varianza è maggiore nel secondo caso. Ciò avviene perché le cardinalità dei due campioni sono distinte e la varianza aumenta se aumenta la cardinalità del campione. Questo "problema" ovviamente non si nota se l'indice viene calcolato su campioni di stessa cardinalità; nel nostro caso però abbiamo quasi sempre insiemi di cardinalità diversa: se una regola estratta da un algoritmo coinvolge determinati permessi, non è assolutamente detto che i gruppi di utenti a cui tali permessi appartengono siano della stessa cardinalità.

La varianza aumenta con la cardinalità, $|T_X|$, del campione perché aumenta la marginalità di ogni elemento, la quale dipende dalla cardinalità del campione. Per renderla indipendente, possiamo operare una normalizzazione, quindi nella pratica la dividiamo per $|T_X| - 1$, essendo questo il numero di addendi nella formula della marginalità. Ridefiniamo quindi la formula della varianza:

Definizione 12. (Varianza)
Dato un campione T_X estratto da un attributo nominale X, calcoliamo la varianza del campione nel modo seguente

$$Var(T_X) = \frac{\frac{\sum_{x_j \in T_X} m(x_j)}{|T_X|}}{|T_X| - 1} = \frac{\sum_{x_j \in T_X} m(x_j)}{|T_X|\big(|T_X| - 1\big)} = \frac{2\sum_{\substack{x_i, x_j \in T_X \\ i<j}} d(x_i, x_j)}{|T_X|\big(|T_X| - 1\big)} = \frac{\sum_{\substack{x_i, x_j \in T_X \\ i<j}} d(x_i, x_j)}{\binom{|T_X|}{2}}$$

In questo modo anche la varianza sarà indipendente dalla cardinalità del campione. La modifica fatta corrisponde esattamente ad aver normalizzato la marginalità (una volta introdotta la modifica sulla distanza, citata nell'Osservazione 2 della sezione precedente, permettendole di arrivare ad 1). In questo modo i risultati saranno sicuramente confrontabili perché indipendenti dalla cardinalità del campione. Notiamo che, come la Definizione 11 della "Varianza basata sulla marginalità" rappresenta media delle marginalità del campione, questa definizione di Varianza rappresenta invece la media delle distanze di un campione, infatti la formula finale è esattamente la media aritmetica delle distanze tra tutte le possibili coppie di elementi. Leggere la formula della varianza in questo modo, porta a dedurre facilmente alcune proprietà, come il fatto che $Var(T_X) \in [0, 1]$ in quanto media di elementi in $[0, 1]$.

6.3 Indice di valutazione dei vincoli

Abbiamo detto che utilizzeremo la varianza per stimare le regole di anti-associazione estratte dall'algoritmo di constraints mining. In questa sezione introduciamo un indice di valutazione che riunisce confidenza (unico criterio utilizzato in [7]) e più calcoli di varianza. Tale indice di "ranking" verrà utilizzato per poter operare una valutazione più completa delle regole, rispetto alla semplice stima tramite confidenza: maggiore è l'indice, maggiore sarà la bontà della regola.

Prima di definire questo indice, abbiamo bisogno di fornire una definizione preliminare.

Definizione 13. (Varianza tra gruppi)
Dato un campione T_X estratto da un attributo nominale X, dati $T_1, T_2 \subseteq T_X$ e dato l'insieme delle coppie

$$K = \{\langle x_i, x_j \rangle \in T_1 \times T_2 \mid x_i < x_j\}$$

definiamo la varianza tra gruppi, come:

$$Var(T_1, T_2) = \frac{1}{|K|} \sum_{\langle x_i, x_j \rangle \in K} d(x_i, x_j)$$

dove $|K|$ è la cardinalità dell'insieme delle coppie K.

In questo tipo di varianza, stiamo calcolando praticamente la "distanza tra due gruppi". A differenza della Definizione 12, la sommatoria non è operata su tutte

le possibili coppie di elementi del campione ma solo sulle coppie $\langle x_i, x_j \rangle \in T_1 \times T_2$, ossia su quelle in cui x_i appartiene al gruppo T_1 e x_j appartiene al gruppo T_2. Questo corrisponde a definire una nuova marginalità per gli elementi dei gruppi T_1 e T_2: per gli elementi del gruppo T_1 si sta calcolando la marginalità sommando solamente le distanze con gli elementi del gruppo T_2, senza considerare le distante con gli altri elementi del campione, e viceversa per gli elementi del gruppo T_2.

Il motivo per cui abbiamo bisogno di introdurre la "varianza tra gruppi" è fondamentalmente perché saremo interessati a calcolare la differenza tra due gruppi di elementi. Vediamo meglio questo concetto: consideriamo A, B permessi, T_A l'insieme degli utenti che possiedono il permesso A e T_B l'insieme degli utenti che possiedono il permesso B. Al posto della varianza tra gruppi, si potrebbe pensare di calcolare semplicemente $Var(T_A \cup T_B)$ ma in questo modo non valuteremmo correttamente la "distanza" tra i due gruppi di utenti. Se le varianze di T_A e T_B sono rispettivamente basse (come vogliamo che siano), cioè T_A è tutta una categoria e T_B un'altra, allora $Var(T_A \cup T_B)$ sarà anch'essa bassa, diversamente da ciò che cerchiamo. In particolare, se le cardinalità di T_A e T_B sono alte, il calcolo della varianza sul campione $T_A \cup T_B$ avrà molti addendi della sommatoria uguali a 0, dati dalle tante distanze nulle tra elementi dello stesso sottoinsieme (T_A o T_B).

Quindi non ha senso confrontare elementi delle stesso sottogruppo per misurare la differenza con l'altro sottogruppo, per questo $Var(T_A \cup T_B)$ non è adatta per valutare la "distanza" tra i due gruppi di utenti. Perfettamente adatta è invece la misura della "varianza tra gruppi": se le cardinalità di T_A e T_B sono maggiori, si accentuano le differenze nel calcolo della varianza tra gruppi, e giustamente essa aumenta. Inoltre notiamo che se gli utenti di T_A sono tutti di una solo categoria e gli utenti di T_B sono tutti di un'altra categoria, quindi tutti gli utenti di T_A sono a distanza 1 da tutti gli utenti di T_B, allora la varianza tra gruppi sarà 1, che corrisponde esattamente a massima differenza tra i due gruppi; viceversa se gli utenti di T_A e T_B sono tutti della stessa categoria, la varianza tra gruppi è 0; quindi più i gruppi sono simili e più la varianza tra gruppi si abbassa.

Riportiamo ora la definizione dell'indice di valutazione.

Definizione 14. (Indice di valutazione)
Siano A, B permessi, T_A l'insieme degli utenti che possiedono il permesso A e T_B l'insieme degli utenti che possiedono il permesso B. Allora definiamo l'indice come:

$$I = conf(\mathrm{A} \Longrightarrow \overline{\mathrm{B}}) \times \left(1 - \frac{Var(T_A) + Var(T_B)}{2}\right) \times Var(T_A, T_B)$$

dove conf($\mathrm{A} \Longrightarrow \overline{\mathrm{B}}$) è la confidenza della regola $\mathrm{A} \Longrightarrow \overline{\mathrm{B}}$, $Var(T_A)$ è la varianza

dell'insieme degli utenti che possiedono il permesso A, equivalentemente $Var(T_B)$*, e* $Var(T_A, T_B)$ *è la varianza tra gruppi di* T_A *e* T_B*.*

Osservazione 3. Data una candidata regola, prodotta dall'algoritmo di constraints mining, andremo a considerare gli insiemi di utenti che possiedono i permessi coinvolti nella regola. Tali insiemi di utenti costituiranno i nostri campioni e le varianze verranno calcolate rispetto alle categorie di appartenenza degli utenti: ogni organizzazione infatti sarà sempre strutturata in modo tale che gli utenti siano classificati in una gerarchia di categorie. Partendo dall'algoritmo in [7], dati A e B permessi, ci limiteremo all'estrazione di regole del tipo A $\Longrightarrow \overline{\text{B}}$ ossia "chi ha il permesso A, non ha il permesso B". I nostri campioni saranno T_A, insieme degli utenti che possiedono il permesso A, e T_B, insieme degli utenti che possiedono il permesso B.

Osservazione 4. Per quanto riguarda l'indice notiamo che I $\in$ [0,1]: nei due casi estremi, se l'indice assume valore 1, la regola è ottima, se assume valore 0 il caso è da escludere. Questa formula per la valutazione include il concetto di confidenza, la varianza di T_A, la varianza di T_B e la varianza tra gruppi di T_A e T_B.

La regola A $\Longrightarrow \overline{\text{B}}$ è buona se che gli utenti di A sono il più possibile di una categoria, gli utenti di B di un'altra e se le due categorie sono ben distinte, di modo che sia ancora più giustificato (e quindi meno casuale) il fatto che chi possiede il permesso A, non debba possedere il permesso B.

Noi ricerchiamo quindi:

- confidenza alta della regola: vuol dire che nella pratica la maggior parte delle persone che possiedono A, non possiedono B; meno eccezioni di questo fatto esistono, ossia più la confidenza è alta, e più è probabile che la regola sia corretta

- varianza di T_A bassa: vogliamo che sia semplice dire che coloro i quali possiedono il permesso A siano utenti con "qualcosa in comune", ossia che appartengano tutti o quasi tutti alla medesima categoria; in termini matematici questo si traduce in varianza rispettivamente 0 o comunque bassa

- varianza di T_B bassa: stesso discorso di T_A, per poter nel complesso sostenere che chi possiede A oppure B lo possiede per un motivo chiaro, ossia perché svolge un determinato tipo di lavoro

- varianza tra gruppi di T_A, T_B alta: per essere giusto che chi possiede A non debba possedere B, gli utenti di A e quelli di B devono essere gruppi di

persone distinti; in termini matematici ciò si traduce in alta varianza tra questi due gruppi.

Utilizzando solo la confidenza come stima, si potrebbero ottenere anche regole che risultano casuali, ossia non hanno un vero significato. Trattando questo caso con l'indice di valutazione che include anche le varianze, nonostante la confidenza sia alta, l'indice risulterà basso per il contributo delle varianze: nell'esempio proposto nella sezione successiva, è presente anche questo caso, che porterà all'esclusione della regola. Il caso migliore invece è quello in cui gli utenti di A e B sono di due categorie completamente distinte, quindi $\mathrm{Var}(T_A)$ e $\mathrm{Var}(T_B)$ minime e $\mathrm{Var}(T_A, T_B)$ massima, e ovviamente confidenza alta.

Esempio

Veniamo ora alla presentazione di un esempio in cui il calcolo dell'indice di valutazione ci permette di capire meglio se le particolari regole estratte siano buone o siano da escludere. Consideriamo la seguente situazione in cui sono presenti 9 utenti $(u_1, \ldots, u_9)$, suddivisi in tre categorie (C_1, C_2, C_3), e 5 permessi $(p_1, \ldots, p_5)$. Si veda la Figura 6.4.

	C_1	C_2	C_3	p_1	p_2	p_3	p_4	p_5
u_1	×			1	0	1	0	0
u_2	×			1	0	1	0	0
u_3	×			0	1	1	0	0
u_4	×			0	1	1	0	0
u_5		×		1	0	0	1	1
u_6	×			0	0	1	0	0
u_7			×	1	0	0	0	0
u_8	×			1	0	1	0	1
u_9		×		0	0	0	1	1

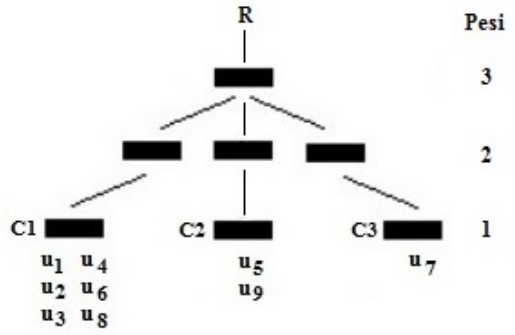

Figura 6.4: Permessi e gerarchia degli utenti

Avremo le seguenti candidate regole prodotte dall'algoritmo, con minsup=0.5 e minconf=0.6:

- $\mathrm{p}_1 \Longrightarrow \overline{\mathrm{p}_2}$ con $conf = 1$
- $\mathrm{p}_3 \Longrightarrow \overline{\mathrm{p}_4}$ con $conf = 1$
- $\mathrm{p}_3 \Longrightarrow \overline{\mathrm{p}_5}$ con $conf = 0.8$

Calcoliamo l'indice in questi casi:

1) - La confidenza della regola di anti-associazione è 1;

- Chiamiamo T_1 linsieme degli utenti che possiedono il permesso p_1 e T_2 linsieme degli utenti che possiedono il permesso p_2. La varianza di T_2 è 0, immediatamente calcolabile in quanto notiamo che gli utenti che possiedono p_2 (u_3, u_4) sono tutti della stessa categoria (C_1), mentre la varianza di T_1 viene calcolata nel modo seguente: gli utenti che possiedono p_1 $(u_1, u_2, u_5, u_7, u_8)$ sono di diverse categorie (C_1, C_2, C_3); calcoliamo la distanze tra gli utenti, tenendo conto che l'indice di Jaccard pesato, calcolato su utenti di categorie distinte, in questo caso risulta essere 0:

$$d(u_1, u_2) = 0 \qquad d(u_1, u_5) = \log_2(2-0) = 1$$
$$d(u_1, u_7) = \log_2(2-0) = 1 \qquad d(u_1, u_8) = 0$$
$$d(u_2, u_5) = \log_2(2-0) = 1 \qquad d(u_2, u_7) = \log_2(2-0) = 1$$
$$d(u_2, u_8) = 0 \qquad d(u_5, u_7) = \log_2(2-0) = 1$$
$$d(u_5, u_8) = \log_2(2-0) = 1 \qquad d(u_7, u_8) = \log_2(2-0) = 1$$

Calcoliamo ora la varianza di T_1 utilizzando la Definizione 12:

$$Var(T_1) = \frac{2 \times 7}{2 \times 4} = 0,7$$

- La varianza tra gruppi di T_1 e T_2, secondo la Definizione 13, sarà:

$$Var(T_1, T_2) = \frac{4}{10} = 0,4$$

essendo le distanze calcolate come precedentemente:

Quindi avremo $I = 1 \times (1 - \frac{0,7+0}{2}) \times 0,4 = \quad 0,26$

In modo analogo è possibile calcolare gli indici dei restanti due casi.

2) $I = 1 \times (1 - \frac{0+0}{2}) \times 1 = \quad 1$

3) $I = 0,83 \times (1 - \frac{0+0.67}{2}) \times 0,71 = \quad 0,39$

Osservazione 5. Controllando i valori dell'indice di valutazione, vediamo come il caso (1), nonostante l'alta confidenza, abbia un indice basso: questo ci porta a pensare che il valore alto della confidenza sia un fatto casuale perché poi effettivamente gli utenti che possiedono i permessi coinvolti nella regola (p_1 e p_2) appartengono tutti a diverse categorie (C_1, C_2, C_3); il contributo della varianza ci indica che la regola è da escludere.

Il caso (2) presenta alta confidenza ed inoltre il valore dell'indice è massimo, quindi la varianza ci conferma che la regola è corretta: controllando notiamo infatti che gli utenti dei due permessi della regola (p_3 e p_4) sono rispettivamente di due categorie separate (C_1 per i primi e C_2 per i secondi).

Il caso (3) presenta un indice non troppo alto, la regola sembrerebbe quindi non particolarmente interessante, ma neanche da escludere. Analizzando più nel dettaglio il calcolo dell'indice, si può notare che la confidenza non massima è data da una singola eccezione, costituita dal solo utente u_8, che nonostante possieda il permesso p_3, possiede anche il permesso p_5. L'eccezione potrebbe essere giustificata da particolari esigenze dell'azienda e magari, in questo caso specifico, essere ammessa dalle politiche di sicurezza vigenti; oppure potrebbe essere un errore di assegnazione che necessita correzione. Escludendo l'utente che fa eccezione, la regola avrebbe confidenza ed indice massimi, quindi in entrambe i casi ipotizzati, la regola può essere considerata valida.

6.4 Conclusioni

In questo capitolo abbiamo visto come è stato possibile applicare la varianza per la valutazione delle regole di anti-associazione. Abbiamo illustrato le modifiche apportate alle definizioni di *distanza* e *varianza* per adattarli maggiormente alle nostre esigenze. Abbiamo successivamente fornito il contributo principale di questo lavoro di tesi, consistente nell'introduzione dell'indice da noi sviluppato per la valutazione più completa dei vincoli. Abbiamo infine presentato un esempio di applicazione di questo indice di "ranking", per mostrare l'utilità pratica dell'utilizzo della varianza.

Capitolo 7

Conclusioni e lavori futuri

Il contributo fondamentale della tesi è stato il lavoro di studio e adattamento della nozione di "varianza semantica", con conseguente introduzione ed applicazione di un particolare indice di "ranking", da noi definito per la stima dei vincoli.

Per quanto riguarda le possibilità di lavoro futuro, illustreremo due diverse "strade". Per quanto riguarda la prima, il meccanismo di estrazione delle regole di anti-associazione genera permessi mutuamente esclusivi solo se essi riguardano una parte consistente degli utenti. Si potrebbe risolvere restringendo il campo di applicazione dell'algoritmo ad un singolo dipartimento dell'organizzazione, che si traduce nella selezione di un solo ramo della gerarchia di categorie, con conseguente riduzione del campione di utenti e permessi analizzato. Ulteriore possibilità è quella di lavorare all'estrazione di altri tipi di regole: ricordiamo che, nell'applicazione della varianza, ci siamo limitati alle sole regole del tipo $A \Longrightarrow \overline{B}$, ossia "chi ha il permesso A non ha il permesso B". Applicando la varianza in modo simile a quanto svolto, si può lavorare all'estrazione delle regole del tipo $\overline{A} \Longrightarrow B$ e $\overline{A} \Longrightarrow \overline{B}$.

Per quanto riguarda invece la seconda "strada", ci spostiamo sulla questione del role mining. Abbiamo spiegato che la generazione dei ruoli è il punto centrale per l'implementazione di un sistema RBAC: un processo automatico, quale un algoritmo di role mining, non riesce sempre ad estrarre ruoli che siano di reale significato. Un'interessante via di sviluppo è sicuramente quella di applicare la varianza per l'estrazione dei ruoli, in modo che essi possano essere corrispondenti a reali mansioni aziendali. È possibile impiegare la varianza per ottenere una stima dei ruoli che un possibile algoritmo di role mining propone: ogni probabile ruolo estratto coinvolgerà un determinato gruppo di utenti: se tramite la varianza risulta che questi utenti appartengono tutti alla stessa categoria, il ruolo risulterà sicuramente più interessante perché coinvolgendo utenti tutti dello stesso tipo sarà più probabilmente identificabile come reale mansione operabile all'interno dell'organizzazione.

Bibliografia

[1] A. Colantonio, R. Di Pietro, A. Ocello *Role Mining in Business - Taming Role-Based Access Control Administration* (2012), World Scientific Publishing Co. Inc

[2] M. Kuhlmann, D. Shohat, G. Schimpf *Role mining-revealing business roles for security administration using data mining technology* (2003), Proceedings of the 8th ACM symposium on Access control models and technologies, pp. 179-186

[3] B. W. Lampson *Protection* (1971), Proceedings of the 5th Princeton Conf. on Information Sciences and Systems, pp. 18-24

[4] J. Vaidya, V. Alturi, Q. Guo *The Role Mining Problem: Finding a Minimal Descriptive Set of Roles* (2007), Proceedings of the 12th ACM symposium on Access control models and technologies, pp. 175-184

[5] J.C. John, S. Sural, V. Atluri, J.S. Vaidya *Role mining under role-usage cardinality constraint* (2012), Information Security and Privacy Research
IFIP Advances in Information and Communication Technology - vol. 376, pp. 150-161

[6] H. Lu, J. Vaidya, V. Atluri, Y. Hong *Constraint-aware role mining via extended boolean matrix decomposition* (2012), IEEE Transactions on Dependable and Secure Computing, vol. 9, no. 5, pp. 655-669

[7] X. Ma, R. Li, Z. Lu, W. Wang *Mining constraints in Role Based Access Control* (2012), Mathematical and Computer Modelling 55.1, pp. 87-96

[8] H. Lu, J. Vaidya, V. Atluri *Optimal boolean matrix decomposition: Application to role engineering* (2008), IEEE 24th International Conference on Data Engineering, pp. 297-306

[9] R. Agrawal, R. Srikant *Fast algorithms for mining association rules* (1994), Proceedings of the 20th Int. Conf. Very Large Data Bases, VLDB, Vol. 1215, pp. 487-499

[10] I. Ray, N. Li, R. France, K. D. Kim *Using UML to visualize role-based access control constraints* (2004), Proceedings of the ninth ACM symposium on Access control models and technologies, pp. 115-124.

[11] R. S. Sandhu, E. J. Coyne, H.L. Feinstein, C.E. Youman *Role-based access control models* (1996), Computer vol.29, n.2, pp. 38-47

[12] D. F. Ferraiolo, D. R. Kuhn *Role-Based Access Controls* (1992), 15th National Computer Security Conference, pp. 554-563

[13] J. Domingo-Ferrer, A. Solanas *A measure of variance for hierarchical nominal attributes* (2008), Information Sciences 178.24, pp. 4644-4655

[14] J. Domingo-Ferrer, *Marginality: a numerical mapping for enhanced exploitation of taxonomic attributes* (2012), Modeling Decisions for Artificial Intelligence. Springer Berlin Heidelberg, pp. 367-381

[15] J. Domingo-Ferrer, D. Sánchez, G. Rufian-Torrell *Anonymization of nominal data based on semantic marginality* (2013), Information Sciences, 242, pp. 35-48

[16] D. Sánchez, M. Batet, D. Isern, A. Valls *Ontology-based semantic similarity: A new feature-based approach* (2012), Expert Systems with Applications 39.9 pp. 7718-7728

[17] F. Chierichetti, R. Kumar, S. Pandey, S. Vassilvitskii *Finding the jaccard median* (2010), Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms. Society for Industrial and Applied Mathematics, pp. 293-311