Rappresentazione della Conoscenza

Lezione 9

Sommario

- sussunzione strutturale 2.3.1
- tableau 2.3.2
- cenni alla analisi di complessità

Tecniche di ragionamento

Il ragionamento su concetti si riconduce alla sussunzione

Siano C, D concetti:

1. C è insoddisfacibile sse C è sussunto da $\bot$;
2. C e D sono equivalenti sse C è sussunto da D e D è sussunto da C;
3. C e D sono disgiunti sse $C \sqcap D$ è sussunto da $\bot$.

Per il linguaggio $\mathcal{ALN}$ si sfrutta questa proprietà e il ragionamento si implementa tramite la **sussunzione strutturale**

Riduzione alla soddisfacibilità

Il ragionamento su concetti si riconduce alla soddisfacibilità (se il linguaggio ammette **congiunzione** e **negazione**)

1. C è sussunto da D sse $C \sqcap \neg D$ è insoddisfacibile;
2. C and D sono equivalenti sse $(C \sqcap \neg D)$ e $(\neg C \sqcap D)$ sono insoddisfacibili;
3. C e D sono disgiunti sse $C \sqcap D$ è insoddisfacibile.

Ragionamento sulla Abox

♢ Ragionamento su ABox si riconduce alla consistenza della ABox

- $\mathcal{A} \models C(a)$ iff $\mathcal{A} \cup \{\neg C(a)\}$ è inconsistente.

♢ La consistenza di concetti si riconduce alla consistenza della ABox

- C è soddisfacibile iff $\{C(a)\}$ è consistente,

Per linguaggi più espressivi ($\mathcal{ALCNR}$) il ragionamento sulla ABox si riconduce alla consistenza della ABox con il **metodo dei tableau**.

Sussunzione strutturale

Le tecniche di ragionamento su reti semantiche e frame si basano sulla costruzione di grafi che corrispondono ai concetti ed alla verifica di alcune proprietà dei grafi.

Sussunzione strutturale: verifica tramite *Confronto strutturale* che un grafo (il subsumee) si può mappare sull'altro (il subsumer).

La sussunzione strutturale è (in genere) *corretta* ma non sempre *completa*:

La risposta "yes" è corretta,
ma la risposta "no" talvolta non lo è.

Sussunzione strutturale

Consideriamo inizialmente il linguaggio semplificato $\mathcal{FL}_0$:

- $C \sqcap D$
- $\forall R.C$

Qunidi aggiungeremo:

- $\bot$
- $\neg A$, (atomica)

$(\leq n\, R)$ e $(\geq n\, R)$

Il linguaggio completo risulta essere $\mathcal{ALN}$.

Forma normale

$$A_1 \sqcap \cdots \sqcap A_m \sqcap \forall R_1.C_1 \sqcap \cdots \sqcap \forall R_n.C_n,$$

dove $A_1, \ldots, A_m$ sono nomi di concetti, $R_1, \ldots, R_n$ nomi di ruoli e $C_1, \ldots, C_n$ sono concetti di $\mathcal{FL}_0$ in forma normale.

La trasformazione di basa sulle proprietà di associatività, commutatività e idempotenza di $\sqcap$ e sull'equivalenza di

$\forall R.(C \sqcap D)$ e $(\forall R.C) \sqcap (\forall R.D)$

Algoritmo di sussunzione strutturale

$$fn(C) = A_1 \sqcap \cdots \sqcap A_m \sqcap \forall R_1.C_1 \sqcap \cdots \sqcap \forall R_n.C_n,$$

$$fn(D) = B_1 \sqcap \cdots \sqcap B_k \sqcap \forall S_1.D_1 \sqcap \cdots \sqcap \forall S_l.D_l,$$

$C \sqsubseteq D$ sse:

1. per ogni $i, 1 \leq i \leq k$, esiste $j, 1 \leq j \leq m$ tale che $B_i = A_j$.
2. per ogni $i, 1 \leq i \leq l$, esiste $j, 1 \leq j \leq n$ tale che $S_i = R_j$ and $C_j \sqsubseteq D_i$.

Proprietà della sussunzione strutturale

La sussunzione può essere verificata tramite un algoritmo ricorsivo:

- ♢ corretto
- ♢ completo
- ♢ di complessità polinomiale

Estensioni di $\mathcal{FL}_0$

Insoddisfacibilità di un concetto ha due conseguenze:

cambia la forma normale
un concetto insoddisfacibile è sussunto da qualsiasi concetto

Consideriamo $\mathcal{FL}_\bot = \mathcal{FL}_0 + \bot$.

Un concetto-$\mathcal{FL}_\bot$ è in **forma normale** sse è $\bot$ o:

$$A_1 \sqcap \cdots \sqcap A_m \sqcap \forall R_1.C_1 \sqcap \cdots \sqcap \forall R_n.C_n,$$

Forma normale

Si calcola la $\mathcal{FL}_0$ forma -normale:
$B_1 \sqcap \cdots \sqcap B_k \sqcap \forall R_1.D_1 \sqcap \cdots \sqcap \forall R_n.D_n$.

se uno dei B_i è $\bot$, l'intera espressione diventa $\bot$

Esempio: $\forall R.\forall R.B \sqcap A \sqcap \forall R.(A \sqcap \forall R.\bot)$

$$A \sqcap \forall R.(A \sqcap \forall R.(B \sqcap \bot)),$$

che diventa

$$A \sqcap \forall R.(A \sqcap \forall R.\bot).$$

Sussunzione

La sussunzione per $\mathcal{FL}_\bot$ è come per $\mathcal{FL}_0$, ma $\bot$ è sussunto da qualsiasi concetto.

Esempio:
$\forall R.\forall R.B \sqcap A \sqcap \forall R.(A \sqcap \forall R.\bot) \sqsubseteq \forall R.\forall R.A \sqcap A \sqcap \forall R.A$

Il confronto delle forme normali:
$A \sqcap \forall R.(A \sqcap \forall R.\bot)$ e $A \sqcap \forall R.(A \sqcap \forall R.A)$
porta al confronto tra $\bot$ e A.

Negazione su concetti primitivi

I concetti negati vengono trattati come primitivi.

Quando A e $\neg A$ si trovano allo stesso livello, vengono sostituiti da $\bot$

Esempio:
$\forall R.\neg A \sqcap A \sqcap \forall R.(A \sqcap \forall R.B)$
$A \sqcap \forall R.(A \sqcap \neg A \sqcap \forall R.B)$
$A \sqcap \forall R.(\bot \sqcap A \sqcap \neg A \sqcap \forall R.B)$
$A \sqcap \forall R.\bot$

Restrizioni numeriche

In $\mathcal{ALN}$, che comprende anche le restrizioni numeriche occorre considerare nuovi tipi di conflitto

- ♢ sui vincoli delle restrizioni: es. $(\geq 2\, R)$ and $(\leq 1\, R)$)
- ♢ sull'esistenziale: es. $(\geq n\, R)$ per $n \geq 1$ e $\forall R.\bot$

Si procede come in precedenza trattando questi nuovi vincoli.

La sussunzione strutturale non è completa per linguaggi più espressivi (negazione e quantificazione esistenziale qualificata)

Algoritmi basati su Tableau

$C \sqsubseteq D$ sse $C \sqcap \neg D$ è insoddisfacibile.

Si considera il linguaggio $\mathcal{ALCN}$ (in realtà anche $\mathcal{ALCNR}$).

Esempio: $(\exists R.A) \sqcap (\exists R.B) \sqsubseteq \exists R.(A \sqcap B)$.

$$C = (\exists R.A) \sqcap (\exists R.B) \sqcap \neg(\exists R.(A \sqcap B))$$

Esempio (continua)

$$C_0 = (\exists R.A) \sqcap (\exists R.B) \sqcap \forall R.(\neg A \sqcup \neg B),$$

che risulta in **forma normale negativa**

Idea di base: costruisco $\mathcal{I}$ tale che $C_0^{\mathcal{I}} \neq \{\}$

Se riesco a trovarla il concetto è soddisfacibile e la relazione di sussunzione non vale, altrimenti vale.

Esempio 2

$(\exists R.A) \sqcap (\exists R.B) \sqcap (\leq 1\, R) \sqsubseteq \exists R.(A \sqcap B)$

In questo caso il vincolo $(\leq 1\, R)$ impone di identificare i due individui che soddisfano $\exists R.A$ e $\exists R.B$.

Con questo vincolo non si riesce a trovare un modello per il concetto e quindi vale la sussunzione.

Non vale quindi l'assunzione di nome unico, ma si possono avere **disuguaglianze**: $x \neq y$

Costruzione del tableau

Sia C_0 un concetto-$\mathcal{ALCN}$ in forma normale negativa.

Per verificare la soddisfacibilità di C_0,

- ♢ si considera la ABox $\mathcal{A}_0 = \{C_0(x_0)\}$
- ♢ si applicano **regole di trasformazione** finché possibile

Se la ABOX "completata" non contiene una contraddizione (clash), allora $\mathcal{A}_0$ è consistente, i.e. C_0 soddisfacibile, e insoddsfacibile altrimenti.

Regole di espansione (trasformazioni)

Trasformazioni **deterministiche** $ABOX \rightarrow ABOX$

Trasformazioni **non-deterministiche** (disgiunzione e almeno)
$ABox \rightarrow$ insieme finito di $ABoxes$ $\mathcal{S} = \{\mathcal{A}_1, \ldots, \mathcal{A}_k\}$

Un insieme di ABOX è **consistente** sse c'è un i, $1 \leq i \leq k$, tale che $\mathcal{A}_i$ è consistente.

Contraddizione

1. $\{\bot(x)\} \subseteq \mathcal{A}$ per un individuo x;
2. $\{A(x), \neg A(x)\} \subseteq \mathcal{A}$ per un individuo x ed un concetto primitivo A;
3. $\{((\leq n\, R))(x)\} \cup \{R(x, y_i) \mid 1 \leq i \leq n+1\} \cup \{y_i \neq y_j \mid 1 \leq i < j \leq n+1\} \subseteq \mathcal{A}$ per individui $x, y_1, \ldots, y_{n+1}$, intero non negativo n, e un ruolo primitivo R.

Regole di trasformazione

$\rightarrow_{\sqcap}$-rule

Condizione: $(C_1 \sqcap C_2)(x) \in \mathcal{A}$, ma $C_1(x)$ e $C_2(x)$ non sono entrambi $\in \mathcal{A}$.

Azione: $\mathcal{A}' = \mathcal{A} \cup \{C_1(x), C_2(x)\}$.

$\rightarrow_{\sqcup}$-rule

Condizione: $(C_1 \sqcup C_2)(x) \in \mathcal{A}$, ma $C_1(x) \notin \mathcal{A}$ e $C_2(x) \notin \mathcal{A}$.

Azione: $\mathcal{A}' = \mathcal{A} \cup \{C_1(x)\}$, $\mathcal{A}'' = \mathcal{A} \cup \{C_2(x)\}$.

Regole di trasformazione 2

$\rightarrow_\exists$-rule

Condizione: $(\exists R.C)(x) \in \mathcal{A}$, ma non c'è uno z tale che $C(z)$ e $R(x,z) \in \mathcal{A}$.

Azione: $\mathcal{A}' = \mathcal{A} \cup \{C(y), R(x,y)\}$ dove y è un individuo $\notin \mathcal{A}$.

$\rightarrow_\forall$-rule

Condizione: $(\forall R.C)(x)$ e $R(x,y) \in \mathcal{A}$, ma $C(y) \notin \mathcal{A}$.

Azione: $\mathcal{A}' = \mathcal{A} \cup \{C(y)\}$.

Regole di trasformazione 3

$\rightarrow_{\geq}$-rule

Condizione: $((\geq n\, R))(x) \in \mathcal{A}$, e non ci sono individui $z_1, \ldots, z_n$ tali che $R(x, z_i)$, $(1 \leq i \leq n)$ e $z_i \neq z_j$ $(1 \leq i < j \leq n) \in \mathcal{A}$.

Azione: $\mathcal{A}' = \mathcal{A} \cup \{R(x, y_i) \mid 1 \leq i \leq n\} \cup \{y_i \neq y_j \mid 1 \leq i < j \leq n\}$, dove $y_1, \ldots, y_n$ sono individui distinti $\notin \mathcal{A}$.

$\rightarrow_{\leq}$-rule

Condizione: (distinti) $y_1, \ldots, y_{n+1} \in \mathcal{A}$ tali che $((\leq n\, R))(x)$ e $R(x, y_1), \ldots, R(x, y_{n+1}) \in \mathcal{A}$, e $y_i \neq y_j \notin \mathcal{A}$ per un $i \neq j$.

Azione: Per ogni coppia y_i, y_j tale che $i > j$ e $y_i \neq y_j \notin \mathcal{A}$, viene generata la ABox $\mathcal{A}_{i,j} = [y_i/y_j]\mathcal{A}$ da $\mathcal{A}$ sostituendo y_i con y_j.

Proprietà dell'algoritmo

- **Correttezza** Se $\mathcal{S}'$ è ottenuto da $\mathcal{S}$ con una regola di trasformazione. $\mathcal{S}$ è consistente sse $\mathcal{S}'$ è consistente
- **Terminazione** La sequenza $\{\{C_0(x_0)\}\} \rightarrow \mathcal{S}_1 \rightarrow \mathcal{S}_2 \rightarrow \cdots$. è finita
- **Completezza** Ogni ABox non contraddittoria ha un modello
- **Complessità** La verifica di soddisfacibilità di concetti $\mathcal{ALCN}$ è PSpace -completa

Analisi di complessità

Brachman e Levesque mostrano che esiste una stretta correlazione tra l'espressività del linguaggio e la complessità computazionale del ragionamento

La sussunzione in $\mathcal{FL}^-$ è polinomiale, ma l'aggiunta al linguaggio di un costrutto sui ruoli (chiamato role restriction) fa diventare la sussunzione coNP-hard.

1. "l'efficienza del ragionamento" può essere confrontata usando l'analisi di complessità;

2. l'insieme di costrutti ammessi nel linguaggio determina le proprietà computazionali del ragionamento.

Proprietà computazionali del ragionamento

1. $\mathcal{FL}$ è intrattabile

2. $\mathcal{ALC}$ is PSPACE-completo

3. $\exists R.C$ è una sorgente di complessità

4. linguaggi "massimamente polinomiali"

$\mathcal{AL}$ +	P	NP-comp			Co-NP comp		PSPACE-complete							
$C \sqcup D$					x	x	x	x		x	x	x		x
$\exists R.C$		x		x			x			x	x		x	x
$(\geq n\, R)$ $(\leq n\, R)$	x					x			x	x		x	x	x
$R \sqcap R'$			x	x				x	x		x	x	x	x

Risultati dell'analisi di complessità

- algoritmi/tecniche di ragionamento corretti e completi
- individuazione dei "casi critici" /interazione dei costrutti
- corrispondenze con altri formalismi

Ma quali sono le implicazioni dei risultati di intrattabilità?

In pratica il **caso peggiore** non capita nelle basi di conoscenza realizzate e questo consente l'uso di linguaggi più espressivi.

Dalla trattabilità alla decidibilità

- trade-off expressivity/tractability
- trade-off expressivity/decidability

"very expressive" Description Logics: ammettono un insieme di costrutti sui concetti e sui ruoli più ricco ed il ragionamento (corretto e completo) più difficile.