Consiglio Nazionale delle Ricerche

ISTITUTO DI ELABORAZIONE DELLA INFORMAZIONE

PISA

Logiche Modali e Temporali per Sistemi Concorrenti

S. Gnesi

Nota Interna IEI B4 - 56

Dicembre 1990

INTRODUZIONE

E' stato provato che le logiche modali e temporali sono appropriate per descrivere interessanti proprietà temporali di sistemi di processi comunicanti e concorrenti [Pnu 81, Man 81, Lam 83] che non possono essere espresse direttamente per mezzo di logiche tradizionali (calcolo proposizionale, logica del primo ordine). Tra queste proprietà si possono citare le classi delle proprietà di "safety" e quelle di "liveness". Una proprietà di safety esprime il fatto che "niente di negativo accadrà" in un sistema, mentre una proprietà di liveness esprime il fatto che "qualcosa di buono accadrà". Proprietà rilevanti del primo tipo sono l'assenza di deadlock, correttezza parziale, etc., mentre del secondo sono la terminazione, la fairness, etc.

Un'altra classe di proprietà esprimibili con la logica temporale include quelle relative a comportamenti ciclici cioè quelli che si hanno in un sistema che deve compiere ciclicamente una sequenza di azioni.

Se caratteriziamo un sistema di processi con l'insieme delle proprietà che esso soddisfa, si può definire un concetto di equivalenze tra diversi sistemi:

i) Due sistemi sono equivalenti se e solo se soddisfano lo stesso insieme di proprietà

oppure:

ii) Due sistemi non sono equivalenti se esiste una proprietà che è soddisfatta da uno ma non dall'altro.

Finora non abbiamo dato una precisa definizione di cosa è un sistema di processi comunicanti e concorrenti; per farlo utilizzeremo una tecnica algebrica che consiste di in un linguaggio per la descrizione di tali sistemi, di un insieme di regole per dare un significato (semantica) ai termini del linguaggio e di alcuni metodi per l'analisi dei sistemi descritti nel linguaggi. La tecnica di cui stiamo parlando è il "Calculus of Communicating Systems" (CCS) sviluppato da R. Milner agli inizi degli anni 80 [Mil 80]. I termini del linguaggio (agenti) sono costruiti attraverso un insieme basilare di operatori e ad ogni termine sintattico è dato un significato operazionale. Diverse nozioni di equivalenza tra agenti possono essere definite, basate su diversi criteri di osservazioni.
Lo scopo di questa nota è quello di mettere in relazione logiche modali e temporali e CCS, definendo logiche adeguate a descrivere il comportamento e le proprietà di sistemi espressi in CCS modulo una certa nozione di equivalenza.

Nel cap. 1 diamo la definizione della sintassi e della semantica del CCS e di alcune equivalenze definite sui termini del linguaggio.

CAPITOLO 1

Il CCS è un modello di computazioni distribuite per trattare la comunicazione in modo matematico. l CCS è un linguaggio semplice ma sufficientemente espressivo infatti con esso si riesce a modellare una vasta classe di sistemi distribuiti. La semplicità del linguaggio deriva dalle due nozioni primitive su cui è basato il CCS: comunicazione e processo. Ogni sistema distribuito si costituito da una collezione processi indipendenti comunicanti tra loro e in CCS l' unico tipo di processo è il processo comunicante.

Il modello CCS è costituito da un linguaggio e da una teoria. Il *linguaggio* consente di descrivere sistemi distribuiti ed il loro comportamento ed è costituito da un insieme di azioni elementari, da un processo basico Nil e da un insieme di operatori che consentono di costruire nuovi processi a partire da quelli esistenti. La *teoria* consente di provare teoremi, analizzare il comportamento e predire proprietà dei sistemi descritti dal linguaggio e consta della definizione di relazioni di equivalenza tra termini del linguaggio, relazioni che vengono rafforzate in modo da ottenere delle congruenze, e di sistemi di assiomi, soddisfatti da tali congruenze, per la corretta riscrittura dei termini.

1.1 Sintassi del CCS

Sia A un insieme dato di *nomi* (etichette di input) α, β. Sia $\neg A = \{ \neg\alpha \mid \alpha \in A\}$ un insieme di *conomi* (etichette di output), disgiunto da A e con la seguente proprietà: $\alpha \in A \Rightarrow \neg\alpha \in \neg A$.

Def. 1.1
Sia $\mathcal{L} = A \cup \neg A$. Una etichetta, denotata con λ, è un elemento di $\mathcal{L}$.

Le etichette rappresentano le azioni osservabili che un processo può fare, dove l'osservazione va riferita ad un osservatore esterno, che sarà un altro processo.

Def. 1.2
Sia $Act = \mathcal{L} \cup \{\tau\}$ dove τ denota un azione non osservabile (silente) mentre $\lambda \in \mathcal{L}$ sono azioni visibili.

Diamo ora la sintassi per la definizione di agenti del CCS :

$P ::= Nil \mid \mu.P \mid P\backslash\alpha \mid P[S] \mid P+P \mid P \mid P \mid x \mid rec.x\,P$ dove $\mu \in Act$, $\alpha \in \mathcal{L}$

Espressioni di questo linguaggio permettono di descrivere sistemi di processi comunicanti.

Diamo ora una descrizione informale degli operatori del CCS:

1. Nil: Rappresenta il processo che non può eseguire alcuna azione.

2.ACTION PREFIXING: Data una etichetta $\mu \in$ Act e un agente P possiamo formare μ.P, che rappresenta un processo che può inizialmente eseguire l'azione μ e poi comportarsi come il processo descritto da P.

3.SOMMA: Siano P e Q agenti su etichette Act_1 e Act_2, P+Q rappresenta un agente su etichette $Act_1 \cup Act_2$ le cui azioni sono la congiunzione di quelle di P e Q

4. COMPOSIZIONE PARALLELA: Siano P e Q agenti su etichette Act_1 e Act_2, P | Q, agente su etichette $Act_1 \cup Act_2$, rappresenta la composizione parallela di P e Q in modo tale che sia P che Q, possono procedere indipendentemente l'uno dall'altro, ma anche possano comunicare tra loro mediante etichette complementari.

5) RESTRIZIONE: Dato P definito su etichette Act e $\alpha \in \mathcal{L}$, $P\backslash\alpha$ è definito su $\mathcal{L}$- { α,$\neg\alpha$ } e rappresenta un agente che si comporta come P tranne che non gli è possibile comunicare attraverso le etichette α,$\neg\alpha$.

6) RELABELLING: Sia P definito su etichette Act e S:Act ->Act' una bigezione che rispetta il complemento, $S(\neg\alpha) = \neg S(\alpha)$; allora P[S] è un agente su Act' che rappresenta il processo stesso con le etichette rinominate secondo S.

7) RICORSIONE: Sia P definito su etichette Act e contenente una variabile di processo x, allora recx.P rappresenta il processo che si comporta come P, dove si sia sostituito alle occorrenze di x il processo recx.P stesso.

1.2 Semantica Operazionale del CCS

La semantica operazionale del CCS è data mediante assiomi e regole di inferenza sui termini dell'insieme degli agenti o processi $\mathcal{P}$ definibili in CCS. Queste regole definiscono il comportamento di un processo P in funzione dei processi che lo costituiscono. Ciò è realizzato definendo su $\mathcal{P}$ una relazione binaria -μ-> per ogni azione atomica μ.
Per dare la semantica al nostro linguaggio, useremo la nozione di sistema delle transizioni etichettato:

Def. 1.3.
Un sistema di transizioni etichettato è una tripla ($\mathcal{P}$, Act, {-μ->:μ∈ Act}), dove $\mathcal{P}$ è un insieme di agenti, Act è l'alfabeto delle azioni e -μ-> ⊆ P×P è una relazione di transizione per ogni μ∈ Act.

Il significato intuitivo della relazione -μ-> è il seguente: Dati gli agenti P_1, $P_2 \in \mathcal{P}$ e un'azione μ e $P_1 = \mu.P_2$ allora:

P_1 -μ-> P_2 significa che P_1 compie un'azione μ e si transforma in P_2.

Diamo ora le regole della semantica operazionale per i termini del CCS:

1) **Nil**: Non c'è alcuna regola. Corrisponde alla nozione intuitiva che Nil rappresenta il processo terminante.

2) **Action-prefixing**: μ.P -μ-> P. L'assioma afferma che il processo può fare μ e poi comportarsi come P.

3) **Somma**: P+Q. Il + rappresenta l'operatore di scelta. Le operazioni possibili di P+Q sono quelle di P o quelle di Q e quindi P+Q può comportarsi come P o come Q. La scelta può essere non deterministica oppure dettata dall'ambiente.

$$(1)\quad \frac{P \xrightarrow{\mu} P'}{P+Q \xrightarrow{\mu} P'} \qquad (2)\quad \frac{Q \xrightarrow{\mu} Q'}{P+Q \xrightarrow{\mu} Q'}$$

4)**Composizione parallela**: P | Q. L'operatore | rappresenta la combinazione di due processi P e Q che possono avanzare concorrentemente oppure possono comunicare tra loro.

$$(1)\quad \frac{P \xrightarrow{\mu} P'}{P \mid Q \xrightarrow{\mu} P' \mid Q} \qquad (2)\quad \frac{Q \xrightarrow{\mu} Q'}{P \mid Q \xrightarrow{\mu} P \mid Q'}$$

$$(3)\quad \frac{P \xrightarrow{\lambda} P' \qquad Q \xrightarrow{\neg\lambda} Q'}{P \mid Q \xrightarrow{\tau} P' \mid Q'}$$

Le regole (1) e (2) esprimono il fatto che P | Q ha tutte le azioni che hanno P e Q quado sono considerati indipendentemente

La regola (3) esprime le possibilità per P e Q di comunicare quando un'azione λ di P e una $\neg\lambda$ di Q si complementano.
Quando λ e $\neg\lambda$ si verificano simultaneamente i due processi P e Q si sincronizzano e il risultato è una singola azione non osservabile dall'ambiente esterno al processo P | Q.

5) **Restrizione**: $P\backslash\lambda$ $(\lambda \in \mathcal{L})$.

$$(1)\quad \frac{P \xrightarrow{\mu} P'}{P\backslash\lambda \xrightarrow{\mu} P'\backslash\lambda} \qquad \mu \notin \{\lambda \text{ e } \neg\lambda\}$$

La regola afferma che $P\backslash\lambda$ ha tutte le azioni di P tranne λ e $\neg\lambda$

es.: $(\alpha.P)\backslash\alpha$ e $(\neg\alpha.Q)\backslash\alpha$ non hanno azioni

6) **Relabelling**: P[S].

$$\frac{P \xrightarrow{\mu} P'}{P[S] \xrightarrow{S(\mu)} P'[S]} \qquad S(\tau) = \tau$$

Ogni azione μ che fa evolvere P in P' diventa un'azione $S(\mu)$ di P[S] che evolve in P'[S]

7) **Ricorsione**: $P = A$ e A sia definito in funzione di P

$$\frac{A \xrightarrow{\mu} P'}{P \xrightarrow{\mu} P'}$$

Esempio 1.2.1

$P_1 = \alpha.\neg\beta.P_1$ $P_2 = \beta.P_2$

Si consideri la composizione parallela P_1 e P_2; una possibile sequenza di azioni del processo $P = P_1 | P_2$ sarà:

$P = P_1 | P_2 \xrightarrow{\alpha} \neg\beta.P_1 | \beta.P_2 \xrightarrow{\tau} P_1 | P_2 \ldots$

Un processo esterno a P può osservare l'azione α eseguita da P.
Due processi che comunicano tramite l'esecuzione di azioni complementari si osservano l'un l'altro dando luogo ad un'azione inosservabile τ.

Esempio 1.2.2

$P = \alpha.P_1 + \beta.P_2 + \beta.P_3$

$P_4 = \alpha.P_4$

$P \mid P_4 = (\alpha.P_1 + \beta.P_2 + \beta.P_3) \mid \neg\alpha.P_4$

Una possibile sequenza di esecuzione sarà la comunicazione tra P_1 e P_4.

$P \mid P_4 \text{ -}\tau\text{-> } P_1 \mid P_4$

1.3 Alberi di derivazione

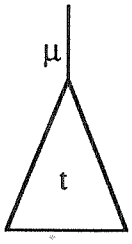
Un modo semplice di rappresentare il comportamento di un processo è costituito da un albero, *albero di derivazione*, il quale visualizza tutte le possibili sequenze di esecuzione.

Def. 1.4
Un *albero di derivazione* su $\mathcal{L}$ è un albero radicato, non ordinato, con un numero finito di rami uscenti da ogni nodo, di cui ogni arco è etichettato da $\mathcal{L} \cup \{\tau\}$

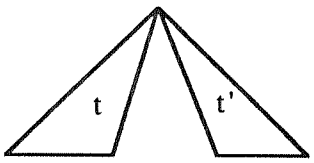
Tra alberi e comportamenti di processiè definibile una corrispondenza biunivoca.
Indichiamo con t, t', t_1... gli alberi di derivazione ed usiamo la seguente notazione: rappresentiamo un albero t in funzione dei sottoalberi componenti t_i collegati ciascuno alla radice di t tramite un arco etichettato con l'azione μ_i.

Nil: l'albero di derivazione per Nil è dato dalla sola radice.

Action prefix: L'albero di derivazione di $\mu.t$, è dato da:

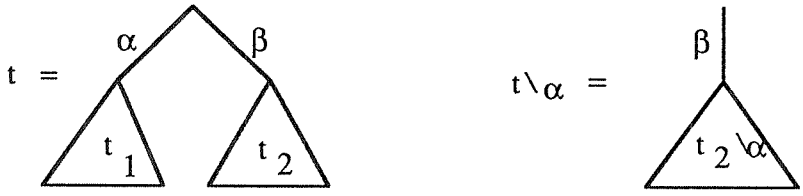


Somma: Dati t_1 e t_2 l'albero di derivazione di t_1+t_2 è ottenuto identificando le radici di t_1 e t_2:

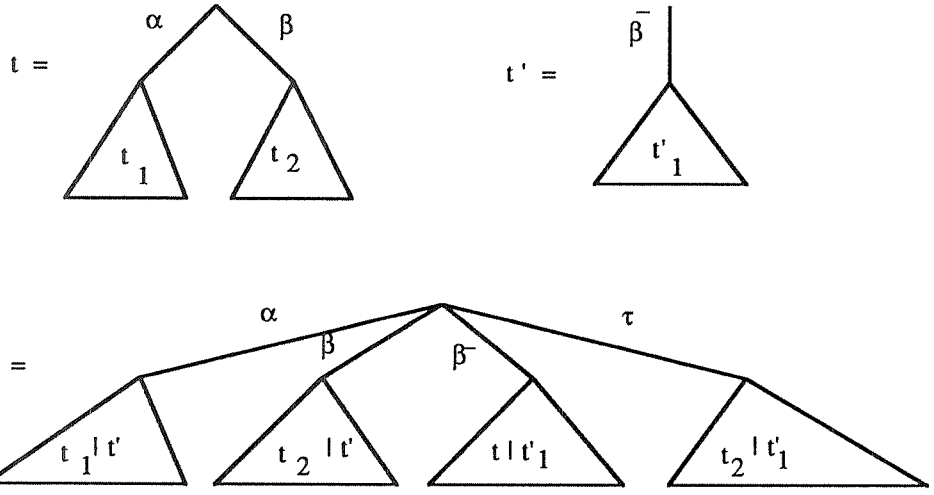


Relabelling: Dato t su etichette $\mathcal{L}$, l'albero di derivazione di t[S] è dato da quello di t con gli archi ridenominati secondo S.

Restrizione: $P\backslash\lambda$ è l'abero ottenuto eliminando i rami etichettati con con $\lambda,\neg\lambda$



Composizione parallela: L'albero di t | t'deve descrivere tutte le azioni di t e t' e tutte le possibili sincronizzazioni, in qualsiasi ordine queste azioni possano verificarsi.



Osservazioni

Ogni cammino su un action tree descrive il possibile ordine in cui le azioni possono verificarsi ma in ciascun momento può svolgersi una sola azione.

L'unico caso in cui si svolgono due azioni contemporaneamente è il caso di una sincronizzazione, ma le due azioni complementari danno luogo sempre ad un'unica azione non osservabile.

CCS modella la concorrenza in termini di non determinismo, ma non esiste concorrenza reale.

1.4 Relazioni di equivalenze definibili su agenti CCS

Sugli agenti del CCS possono essere definite diverse nozioni di equivalenza. Esse corrisponderanno ai possibili modi in cui un agente esterno può osservare un agente P.
Si può definire una nozione di equivalenza tra due agenti P e Q che si basa sugli insiemi delle possibili esecuzioni di P e Q. A seconda che vengano considerate sia esecuzioni parziali che totali, oppure solo totali, verranno indotte due diverse nozioni di equivalenza. Queste saranno, rispettivamente: equivalenza a *tracce* e equivalenza a *tracce massimali.*

Def. 1.5
Una sequenza $\sigma \in Act^*$ è una computazione parziale di un processo P se $\sigma \in Act^*$ ed esistono azioni $\alpha_1, \alpha_2,...,\alpha_n \in Act$ e processi $P_1, P_2,...,P_n \in \mathcal{P}$ tali che $\sigma = \alpha_1\ \alpha_2\ \ \alpha_n$ e:

$$P\text{-}\alpha_1\text{->}\ P_1\ \text{-}\alpha_2\text{->}\ ....\text{-}\alpha_n\text{->}\ P_n$$

Due processi P e Q sono *equivalenti a tracce* se essi hanno esattamente lo stesso insieme di computazioni parziali e lo indicheremo con: $P \sim Q$.

Se consideriamo invece l' insieme delle computazioni totali (quelle infinite oppure quelle per cui o non esiste un successore nell' albero delle derivazioni) di un processo, questo può essere identificato con il linguaggio generato dal processo quindi si può dire:

Def. 1.6
Due processi P, Q sono equivalenti a *tracce massimali.* se essi denotano lo stesso linguaggio. e lo indicheremo con: $P \simeq Q$.

Def. 1.7
Una relazione binaria $\mathcal{R} \subseteq PxP$ su agenti è una *bisimulazione forte* se $P\mathcal{R}q$ implica, per ogni $\alpha \in Act$

i) se P' è tale che $P\text{-}\alpha\text{->}\ P'$ allora esiste un Q' tale che $Q\text{-}\alpha\text{->}Q'$ e $P'\mathcal{R}Q'$
ii) se Q' è tale che $Q\text{-}\alpha\text{->}Q'$ allora esiste un P' tale che $P\text{-}\alpha\text{->}P'$ e $P'\mathcal{R}Q'$

Due processi P e Q sono *fortemente bisimili* (strong) se e solo se esiste una bisimulazione $\mathcal{R}$ con P$\mathcal{R}$Q, e lo indicheremo con: P≋Q.

La relazione $\mathcal{R}$ si basa sul concetto di osservazione: due processi sono equivalenti se un osservatore esterno non rileva alcuna differenza nel comportamento dei due processi. $\mathcal{R}$ è una relazione di equivalenza per cui gode delle proprietà: riflessiva, simmetrica e transitiva:

P≋P
se P≋Q allora Q≋P
se P≋Q. e Q≋R allora P≋Q

In realtà questa relazione di equivalenza distingue processi che dal punto di vista di un osservatore esterno sono in realtà indistinguibili. Introduciamo quindi una nuova relazione di transizione tra processi:

Def. 1.8
Siano P e Q due processi e α un'azione: diciamo che
$P \Rightarrow_{\alpha} Q$ se $P \to^{\tau^m} \to^{\alpha} \to^{\tau^n} Q$

Def. 1.9
Una relazione binaria $\mathcal{R} \subseteq P \times P$ è una *bisimulazione debole* (weak) se P$\mathcal{R}$'Q implica, per ogni $\alpha \in$ Act:

1) Se P' è tale che $P \xrightarrow{\alpha} P'$ allora o
   i) esiste Q' tale che $Q \Rightarrow_{\alpha} Q'$ e P'$\mathcal{R}$Q', oppure
   ii) $\alpha = \tau$...e P'$\mathcal{R}$Q.

2) Se Q' e tale che $Q \xrightarrow{\alpha} Q'$ allora o
   i) esiste P' tale che $P \Rightarrow_{\alpha} P'$ e P'$\mathcal{R}$Q' oppure
   ii) $\alpha = \tau$. e P$\mathcal{R}$Q'.

Due processi P e Q sono *osservazionalmente equivalenti* se e solo se esiste una bisimulazione debole $\mathcal{R}$ con P$\mathcal{R}$Q, e lo indicheremo con: P≈Q.

Utilizzando la relazione $\Rightarrow_{\alpha}$ si possono definire le versioni deboli della equivalenza a tracce e di quella a tracce massimali.

Esempio 1.4.1
$P_1 = \alpha.(\beta.Nil+\gamma.Nil)$ $P_2 = \alpha.\beta.Nil+\alpha.\gamma.Nil$

P_1 e P_2 sono equivalenti a tracce ma non sono bisimili; infatti l' insieme di computazioni parziali e totali di questi coincide e è $\{\alpha.\beta, \alpha.\gamma\}$, ma applicando Def.1.7 si ha che:

$P_1\text{-}\alpha\text{->} (\beta.Nil+\gamma.Nil)$ ======> $P_2\text{-}\alpha\text{->} \beta.Nil$ oppure
$P_2\text{-}\alpha\text{->} \gamma.Nil$

Dobbiamo quindi verificare se: $(\beta.Nil+\gamma.Nil)=\beta.Nil$ oppure se $(\beta.Nil+\gamma.Nil)=\gamma.Nil$.
Applicando di nuovo la def.1.7 a $(\beta.Nil+\gamma.Nil)$ e a $\gamma.Nil$ si vede che : $(\beta.Nil+\gamma.Nil)\text{-}\beta\text{->} Nil$ ma $\gamma.Nil$ non può compiere transizioni etichettate con β.

Esempio 1.4.2

$P_1= \alpha.Nil+\alpha.Nil$ $P_2= \alpha.Nil+\tau.\alpha.Nil$
P_1 e P_2 sono osservazionalmente equivalenti, infatti:

$P_1\text{-}\alpha\text{->} Nil$ ======> $P_2\text{-}\alpha\text{->}Nil$ e viceversa
$P_2\text{-}\tau\text{->}\alpha.Nil\text{-}\alpha\text{->} Nil$ ======> $P_1\text{-}\alpha\text{->} Nil\text{-}\alpha\text{->} Nil$.

Esempio 1.4.2
$P_1= \alpha.Nil+\beta.Nil$ $P_2= \alpha.Nil+\tau.\beta.Nil$
P_1 e P_2 non sono osservazionalmente equivalenti, infatti:

$P_1\text{-}\alpha\text{->} Nil$ ======> $P_2\text{-}\alpha\text{->}Nil$
$P_1\text{-}\beta\text{->} Nil$ ======> $P_2 =\beta\text{=>}Nil$ viceversa si ha:

$P_2\text{-}\tau\text{->}\beta.\ Nil$ ======> $P_1 \neq \beta.Nil$.