

Operatori per il Parallelismo in CSP

Composizione Parallela di Processi Sequenziali (1)

- Le notazioni introdotte permettono di descrivere agevolmente tutti i processi che si presentano solo sotto forma sequenziale
- Componendo **in parallelo** 2 o più processi sequenziali la complessità e l'interesse aumentano notevolmente

Composizione Parallela di Processi Sequenziali (2)

- Con la composizione parallela lo stato dell'intero sistema dipende dallo stato dei processi componenti
 - Il numero di stati aumenta esponenzialmente con l'aumentare dei processi
 - La difficoltà nel gestire tali sistemi è elevatissima

Composizione Parallela di Processi Sequenziali (3)

- CSP considera la combinazione parallela **come un altro processo** a cui applicare gli operatori già definiti
- Nell'esecuzione parallela i processi si influenzano l'un l'altro attraverso le rispettive comunicazioni

Obiettivo

- Evitare
 - Deadlock
 - Livelock
 - sequenza infinita di comunicazioni interne tra componenti, con un'apparenza esterna analoga al deadlock
 - Comportamenti non deterministici

Sincronizzazione di azioni visibili (1)

- L'operatore \parallel è quello più semplice e permette di sincronizzare le azioni visibili svolte dai due processi operandi
- $P \parallel Q$ esegue l'azione $a \in \Sigma$ sse sia P che Q possono eseguire a
- Quindi:

$$(\exists x: A \rightarrow P(x)) \parallel (\exists x: B \rightarrow Q(x)) = \exists x: A \cap B \rightarrow (P(x) \parallel Q(x))$$

Sincronizzazione di azioni visibili (2)

- La legge precedente riflette l'idea che **ogni processo parallelo è equivalente ad uno sequenziale**:
 - I due prefissi posti in parallelo vengono trasformati in un singolo processo esterno al parallelismo
- Di conseguenza, CSP permette di governare agevolmente la complessità del sistema risultante

Esempio (1)

- Consideriamo un processo che comunica il simbolo **a**,
 - sempre tranne quando il numero di **a** comunicate è divisibile per N ,
 - in tal caso comunica il simbolo **b**
- Tale processo può essere espresso come

$$\text{Mult}(N, m) = \begin{cases} a \rightarrow \text{Mult}(N, N) \square b \rightarrow \text{Mult}(N, m) & \text{se } m=0 \\ a \rightarrow \text{Mult}(N, m-1) & \text{altrimenti} \end{cases}$$

Esempio (2)

- Combinando in parallelo $\text{Mult}(N, 0)$ e $\text{Mult}(M, 0)$ si ottiene $\text{Mult}(N, 0) \parallel \text{Mult}(M, 0) = \text{Mult}(\text{mcm}(M, N), 0)$ dove $\text{mcm}(M, N)$ è il minimo comune multiplo tra M e N
- E' un handshacking: b si verifica solo se entrambi i comunicanti sono in accordo
 - non c'è direzione nella comunicazione

Esempio (3)

- Lo stesso meccanismo può avere l'effetto di un output di un processo verso un altro $(c!x \rightarrow P) \parallel (c?y \rightarrow Q(y)) = c!x \rightarrow (P \parallel Q(x))$

y è input sul channel c: Q evolve in funzione di y

x è output sul channel c: P evolve indipendentemente da x

Interfaccia

- Per specificare che una coppia di processi deve sincronizzarsi rispetto a (deve effettuare un handshaking su) un determinato evento è definito l'operatore di parallelismo attraverso l'interfaccia X
- $P \parallel_X Q$
 - forza i processi P e Q a sincronizzarsi rispetto a tutti gli eventi in X
 - ma permette comunque di effettuare eventi esterni a X

Modello delle Tracce

- È un modello per la descrizione del comportamento di un processo
 - ogni processo è rappresentato dal suo insieme di tracce
- Ad es.
 - $\text{traces}(\text{Stop}) = \{\langle \rangle\}$, con $\langle \rangle$ sequenza vuota
 - $\text{traces}(\mu P.a \rightarrow P \square b \rightarrow \text{Skip}) = \{\langle a \rangle^n, \langle a \rangle^n \wedge \langle b \rangle, \langle a \rangle^n \wedge \langle b \rangle, \sqrt{\rangle} \mid n \in \mathbb{N}\}$, dove
 - s^t è la concatenazione di t a s
 - s^n è la concatenazione di s con sé stessa n volte
 - $\sqrt{\rangle}$ è il simbolo di terminazione

Proprietà Modello Tracce (1)

- Ogni processo è rappresentato dal suo insieme di tracce
- Per il set delle tracce valgono sempre le seguenti proprietà
 - P1: È non vuoto
 - include almeno la empty trace
 - P2: È chiuso rispetto all'operazione di prefix
- Le tracce assumono valori in un insieme di sequenze finite di simboli di Σ , con eventualmente un \surd finale

Proprietà Modello Tracce(2)

- Sia X un insieme di simboli e X^* l'insieme di sequenza finita di membri di X (inclusa la sequenza vuota), allora
$$\text{traces}(P) \subseteq \Sigma^* \cup \{s \wedge \surd \mid s \in \Sigma^*\}$$
- Il modello delle tracce T è l'insieme di tutti i sottoinsiemi di $\Sigma^* \cup \{s \wedge \surd \mid s \in \Sigma^*\}$ che soddisfano P1 e P2

Modello Tracce per CSP (1)

- $\text{traces}(\text{stop}) = \{\langle \rangle\}$
- $\text{traces}(a \rightarrow P) = \{\langle \rangle\} \cup \{\langle a \rangle \wedge s \mid s \in \text{traces}(P)\}$
 - Processo vuoto oppure processo in cui l'evento iniziale a è seguito da una traccia di P

Modello Tracce per CSP (2)

- $\text{traces}(\text{?}x: A \rightarrow P(x)) = \{\langle \rangle\} \cup \{\langle a \rangle \wedge s \mid a \in A \wedge s \in \text{traces}(P[a/x])\}$
 - analogo al precedente, con la differenza che l'evento iniziale è scelto nell'insieme A e da questo dipende l'evoluzione successiva;
 - $P[a/x]$ indica che a sostituisce le occorrenze di x

Modello Tracce per CSP (3)

- $\text{traces}(c?x: A \rightarrow P(x)) = \{\langle \rangle\} \cup \{\langle c.a \rangle^s \mid a \in A \wedge s \in \text{traces}(P[a/x])\}$
 - analogo al precedente
- $\text{traces}(P \square Q) = \text{traces}(P) \cup \text{traces}(Q)$
- $\text{traces}(P \sqcap Q) = \text{traces}(P) \cup \text{traces}(Q)$
- $\text{traces}(P \parallel Q) = \text{traces}(P) \cap \text{traces}(Q)$

Modello Tracce per la ricorsione (1)

- $P = F(P)$ significa che P ha lo stesso comportamento (e quindi la stessa traccia) del processo $F(P)$
- In termini di tracce, la $F()$ rappresenta sempre un mapping monotono da T a se stessa,
 - cioè, se $\text{traces}(Q_1) \subseteq \text{traces}(Q_2)$ allora $\text{traces}(F(Q_1)) \subseteq \text{traces}(F(Q_2))$

Modello Tracce per la ricorsione (2)

- Quindi $\text{traces}(F(\text{Stop})) \subseteq \text{traces}(P)$
- Applicando $F()$ e la monotonia si ottiene $\text{traces}(F^n(\text{Stop})) \subseteq \text{traces}(P)$ per ogni n , dove $F^n(\text{Stop})$ è ciò che si ottiene applicando n volte a Stop
- Quindi
 $\text{traces}(P) = \bigcup \{\text{traces}(F^n(\text{Stop})) \mid n \in \mathbb{N}\}$