Metodi Formali

UniShare

Davide Cozzi
@dlcgold

Gabriele De Rosa
@derogab

Federica Di Lauro
@f_dila

Indice

Capitolo 1

Introduzione

Questi appunti sono presi a lezione. Per quanto sia stata fatta una revisione è altamente probabile (praticamente certo) che possano contenere errori, sia di stampa che di vero e proprio contenuto. Per eventuali proposte di correzione effettuare una pull request. Link: `https://github.com/dlcgold/Appunti`.
Grazie mille e buono studio!

1.1 Contenuti del Corso

Il corso tratta di metodi e tecniche formali per specificare, disegnare e analizzare sistemi complessi, in particolare sistemi concorrenti e distribuiti costituiti da componenti che operano in modo indipendente e che interagiscono tra loro.

Si usa un linguaggio logico che spiega il comportamento di tali sistemi e fa riferimento alla **logica temporale** di tali sistemi, in quanto le proprietà di tali sistemi sono tali per cui evolvono con il cambiamento di stato del sistema e quindi serve una logica che descriva le proprietà dell'evoluzione del comportamento.
Si parlerà delle **Reti di Petri**, ovvero uno strumento per modellare tali sistemi concorrenti e distribuiti. Questo modello ha intrinsechi dei teoremi matematici atti a studiare il comportamento di tali sistemi.
In laboratorio si studieranno algoritmi e strumenti software per la modellazione e l'analisi di tali sistemi.
Si introducono a che sistemi dinamici a tempi discreti, come gli **automi cellulari**.

Capitolo 2

Sviluppo di Modelli e Sistemi

Si hanno diverse fasi di sviluppo di *sistemi complessi* (nel nostro caso **concorrenti** e **distribuiti**). Si hanno 4 grandi fasi (che riprendono le generiche fasi dello sviluppo software), che non seguono una rigida sequenza cronologica tra di loro:

1. specifica del problema e delle proprietà della soluzione
2. modellazione della soluzione
3. implementazione
4. verifica, validazione e collaudo, sia sul modello che implementazione (con eventuali modifiche)

Queste fasi possono alternarsi a vicenda.
I metodi formali possono svolgere una parte rilevante in tutte queste 4 fasi e hanno la prerogativa di sviluppare questi sistemi in maniera corretta e persistente.
Ci si focalizza sulla modellazione e sulla specifica delle proprietà. Si studia inoltre la verifica delle proprietà sul modello costruito. *In questo corso si lascia un attimo da parte l'aspetto implementativo, che comunque seguirebbe alla verifica e alla validazione del metodo.*
Si hanno diversi modelli di sistemi concorrenti e distribuiti, presenti in letteratura:

- **Algebre di Processi**, ovvero una miriade di diversi linguaggi, studiate inizialmente da Milner, che introdusse il calcolo dei sistemi comunicanti, un calcolo algebrico utile alla semantica della concorrenza. Inoltre Hoare ha introdotto i **processi sequenziali comunicanti** come un nucleo di linguaggio di programmazione,

usato come linguaggio macchina per le prime macchine parallele. Queste algebre si basano sul paradigma di avere un forte aspetto della **composizionalità**, in quanto un sistema viene visto come costituito da diverse componenti autonome (sia hardware, che software, che umane) che interagiscono tra loro sincronizzandosi (in modo sincrono, *handshaking*, sfruttando un "canale di comunicazione" che viene modellato come un processo) e scambiandosi messaggi. Questo paradigma è anche alla base dello sviluppo di molti linguaggi di programmazione specificatamente dedicati alla concorrenza.

- **Automi a Stati Finiti**. Un modello concorrente e distribuito viene spesso rappresentato attraverso **sistemi di transizioni etichettati**, che sono una derivazione del modello degli automi a stati finiti, già usati in letteratura per modellare reti neurali, progettare circuiti asincroni, modellare macchine a stati finiti, riconoscere linguaggi regolari (il teorema di Kleene ci ricorda che *ad un automa a stati finiti è possibile associare un'espressione regolare*) e per la modellazione di protocolli di comunicazione.

- **Reti di Petri**, introdotte da Petri con la **teoria generale delle reti di Petri** nella sua tesi di dottorato. Questa teoria parte da una critica al modello a stati finiti dove il focus è su stati globali e trasformazione di stati globali. Petri cercava invece una teoria matematica (fondata sui principi della fisica moderna della relatività e della quantistica) che fosse una teoria dei sistemi in grado di descrivere sistemi complessi in cui mettere al centro il flusso di informazione e che potesse permettere di analizzare l'organizzazione dal punto di vista del flusso di informazione che passa da una componente all'altra. Non si ha il focus, quindi, su "macchine calcolatrici" ma come supporto alla comunicazione in organizzazioni complesse. Si hanno quindi diversi elementi chiave:

  - la comunicazione
  - la sincronizzazione tra componenti
  - il flusso di informazione che passa tra le varie componenti
  - la relazione di concorrenza e l'indipendenza causale tra i vari eventi che comportano i cambiamenti di stato. Ci si concentra su stati locali e non sulla visione di una sequenza di azioni e di uno stato globale

La teoria delle reti di Petri è stata poi sviluppata e ha avuto diverse applicazioni. Sono stati sviluppati diversi linguaggi, ovvero diverse **classi di reti di Petri** per descrivere un sistema complesso a livelli differenti di astrazione.
Sono state anche sviluppate tecniche formali di analisi e di verifica del modello (disegnato mediante reti di Petri), basate sulla teoria dei grafi e sull'algebra lineare.
Le reti di Petri hanno avuto un notevole utilizzo in diversi ambiti applicativi anche estranei all'informatica pura e allo studio della concorrenze, come la modellazione di sistemi biologici o la modellazione di reazioni chimiche. Mediante una classe di reti particolare, le **reti stocastiche** si può valutare le prestazioni di un determinato modello.

2.0.1 Sistemi di Transizioni Etichettati

Definizione 1. *I sistemi di transizione etichettati sono definiti come gli automi a stati finiti ma senza essere visti come riconoscitori di linguaggi infatti un sistema è formato da un insieme, solitamente finito, di stati globali S. Si ha poi un alfabeto delle possibili azioni che può eseguire il sistema. Si hanno anche delle relazioni di transizioni, ovvero delle transizioni che permettono di specificare come, attraverso un'azione, si passa da uno stato ad un altro. Le transizioni si rappresentano con archi etichettati tra i nodi, che rappresentano gli stati. Le etichette degli archi rappresentano le azioni necessarie alla trasformazione. L'insieme delle azioni viene chiamato E mentre $T \subseteq S \times E \times S$ è l'insieme degli archi etichettati. Può essere, opzionalmente, individuato uno stato iniziale s_0. Un sistema non è obbligato a "terminare", quindi non si ha obbligatoriamente uno stato finale.*
Riassumendo quindi un sistema di transizione etichettato è un quadrupla:

$$A = (S, E, T, s_0)$$

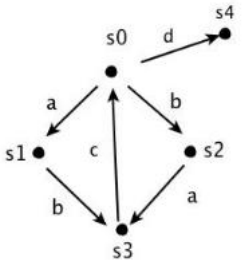

Figura 2.1: Esempio di sistema di transizione etichettato

La critica di Petri è che in un sistema distribuito non sia individuabile uno **stato globale**, che in un sistema distribuito le trasformazioni di stato siano **localizzate** e non globali, che non esista un sistema di riferimento temporale unico (si possono avere più assi temporali in un sistema distribuito). Quindi la simulazione sequenziale non deterministica (emantica a "interleaving") dei sistemi distribuiti è una forzatura e non rappresenta le reali caratteristiche del comportamento del sistema, ovvero la località, la distribuzione degli eventi e la relazione di dipendenza causale e non causale tra gli eventi.

2.1 Sistemi Elementari per le Reti di Petri