UNIVERSITÀ
DI TORINO

Università degli Studi di Torino

Corso di Laurea in Informatica

Algebre di Processi e le Origini della Nozione di Vera Concorrenza

Tesi di Laurea

Relatore/Relatrice
Prof. Cardone Felice

Candidato/a
Gismano Stefano
Matricola 917723

Anno Accademico 2022/2023

DICHIARAZIONE DI ORIGINALITÀ

Dichiaro di essere responsabile del contenuto dell'elaborato che presento al fine del conseguimento del titolo, di non avere plagiato in tutto o in parte il lavoro prodotto da altri e di aver citato le fonti originali in modo congruente alle normative vigenti in materia di plagio e di diritto d'autore. Sono inoltre consapevole che nel caso la mia dichiarazione risultasse mendace, potrei incorrere nelle sanzioni previste dalla legge e la mia ammissione alla prova finale potrebbe essere negata.

Indice

1 Introduzione

L'algebra di processi è un campo dell'informatica di grande importanza, che tratta la rappresentazione, descrizione e verifica dei processi e delle loro interazioni. Si tratta di un settore ancora molto discusso, che ha dato vita a molteplici paradigmi e teorie diverse tra loro, ognuna con i propri obiettivi, pregi e difetti. In questo contesto, il concetto di vera concorrenza si trova in stretta contrapposizione a quello di interleaving.

La diatriba fra questi due modelli è stata oggetto di dibattito all'interno di una mailing list dedicata alla concorrenza, in cui alcuni massimi esperti dell'argomento, negli ultimi mesi del 1990 hanno un'accesa e interessante discussione a riguardo. I principali attori in questo scenario sono Vaughan Pratt e Leslie Lamport.

Vaughan Pratt, professore al MIT dal 1972 al 1981 e alla Stanford University dal 1981 al 2000, dove ora è professore emerito, è un pioniere dell'informatica in una moltitudine di campi. Ha prodotto circa un centinaio di articoli nel corso della sua carriera, nonché contribuito in modo estensivo alla ricerca nei campi della matematica, della logica, dell'analisi degli algoritmi e persino della chimica. Egli è stato anche group leader dello Stanford Concurreny Group, alcuni membri del quale sono intervenuti all'interno della corrispondenza qui in esame. Attualmente si interessa di argomenti quali lo speech recognition, le tecnologie di propulsione di automobili, e cambiamento climatico.

Leslie Lamport è un matematico e informatico i cui studi sui sistemi concorrenti gli sono valsi il Premio Turing del 2013. Egli ha prodotto, al momento della scrittura di questo testo, 192 articoli, molti dei quali sulla logica temporale e sulla concorrenza tra processi, forse il più importante dei quali è "Time, Clocks and the Ordering of Events in a Distributed System" [21], del 1978, che tratta la sincronizzazione degli orologi logici e l'implementazione di una generica macchina a stato distribuito. Ha inoltre sviluppato il TLA+, un linguaggio di specifica formale pensato per la modellazione di sistemi concorrenti. Ai più è conosciuto per l'invenzione di LaTeX, linguaggio di markup open source estremamente popolare e usato, tra le altre cose, per la scrittura di questa tesi.

Questi due giganti dell'informatica si schierano rispettivamente a favore della vera concorrenza e dell'interleaving, nozioni che saranno al centro dell'attenzione in questa tesi, dando vita a un dibattito acceso in cui, passando

dall'invettiva e la presa in giro, si arriva a parlare della natura del concetto stesso di scienza.

2 Alcune nozioni fondamentali

Prima di iniziare con l'analisi della corrispondenza in sé, è bene introdurre alcuni concetti fondamentali per la comprensione del seguito[1].

2.1 Algebre di Processi e Sistemi di Transizione

I *processi* sono agenti che agiscono e interagiscono continuamente, ovvero sono in grado di farlo in qualunque momento e per qualunque lasso di tempo, con altri agenti simili e con l'ambiente a loro comune.

Dunque, i processi non sono necessariamente di natura esclusivamente informatica, sebbene questo sia il contesto in cui verranno studiati in seguito. Essi possono anche essere immaginati come appartenenti al mondo reale; ad esempio, delle persone che parlano tra loro, delle automobili in un incrocio, o delle api in un alveare. In ambito informatico, quindi, un processo può essere visto come una collezione strutturata di eventi, che eventualmente costituiscono un programma o una parte di esso. Un processo è in grado di interagire con altri processi simili, tendenzialmente tramite l'invio di messaggi.

Le *algebre dei processi* sono linguaggi matematicamente rigorosi con semantiche ben definite che permettono di descrivere e verificare proprietà di sistemi concorrenti che comunicano tra loro.

Per comprendere meglio di cosa si sta parlando, introduciamo una semplice algebra di esempio, che comprende alcune delle operazioni più comuni e utili della teoria della concorrenza. Innanzitutto definiamo delle azioni atomiche, ovvero delle azioni che sono considerate non-interrompibili, e le indichiamo con delle lettere minuscole $a, b, c, \ldots$. A partire da esse, possiamo definire nuovi processi mediante una definizione induttiva:

$$P, Q := nil \mid a \mid a.P \mid P + Q \mid P||Q \tag{1}$$

Gli operatori di questa algebra di esempio rappresentano quelli più importanti e comuni presenti nelle algebre più utilizzate: $a.P$ indica che il processo effettua l'azione a e poi procede eseguendo il processo P.

[1] Le definizioni seguenti sono tratte da [6] e [24].

$P + Q$ è l'operatore di somma, o scelta. Esso indica che il processo può procedere eseguendo P oppure eseguendo Q.

Infine, $P||Q$ è l'operatore di composizione parallela, ovvero rappresenta l'esecuzione concorrente del processo P e del processo Q.

Il concetto di *sistema di transizione* è estremamente importante per vari ambiti dell'informatica, tra cui la teoria degli automi e, appunto, le algebre dei processi. Forniamo qui una definizione tratta da [17].

Un sistema di transizione è una coppia $(Q, \rightarrow)$, dove Q è un insieme di *stati* e $\rightarrow$ è una relazione binaria su Q, chiamata l'insieme di *transizioni*.

Un sistema di transizione nominato è una tripla $(Q, \rightarrow, \Sigma)$, dove $(Q, \rightarrow)$ è un sistema di transizione, e a ogni transizione è assegnato un nome nell'insieme Σ.

Quelli che Keller chiama "sistemi di transizione nominati" sono più comunemente conosciuti come "sistemi di transizione etichettati", o "LTS".

2.2 Semantiche delle Algebre di Processi

Esistono molte diverse algebre di processi, ognuna delle quali adotta approcci differenti per la modellazione dei sistemi che intende descrivere.

In particolare, esse possono essere suddivise in base a quale tipologia di semantica utilizzano: operazionale, denotazionale o algebrica, sebbene ogni algebra possa ammettere più di un tipo di semantica. Le algebre più conosciute e utilizzate sono CCS (Calculus of Communicating Systems), CSP (Communicating Sequential Processes), e ACP (Algebra of Communicating Processes). Le fonti fondamentali che descrivono le algebre sopracitate sono "A Calculus of Communicating Systems", di Robin Milner [23], "Communicating Sequential Processes", di Tony Hoare [14], e "Process Algebra for Synchronous Communication", di Bergstra e Klop [2].

Una *semantica operazionale* modella un processo mediante un sistema di transizioni etichettate. Gli stati del sistema di transizioni sono termini dell'algebra di processi, mentre le etichette delle transizioni tra stati rappresentano le azioni o le interazioni che sono possibili a partire da un certo stato e lo stato che è raggiunto dopo che l'azione viene eseguita.

Data un'algebra di processi, la semantica operazionale dei suoi operatori è specificata induttivamente tramite delle cosiddette regole SOS, ovvero di Semantica Operazionale Strutturale. Tali specifiche prendono la forma

di regole di inferenza che determinano quali sono le transizioni valide all'interno di un LTS appartenente all'algebra in esame, e in cui le premesse e le conclusioni sono triple (P, α, Q), più spesso rappresentate come $P \xrightarrow{\alpha} Q$. Quindi, le regole per ogni operatore *op* saranno nella forma seguente, dove $\{i_1, \ldots, i_m\} \subseteq \{1, \ldots, n\}$ ed $E'_i = E_i$ quando $i \notin \{i_1, \ldots, i_m\}$:

$$\frac{E_{i_1} \xrightarrow{\alpha_1} E'_{i_1} \ldots E_{i_m} \xrightarrow{\alpha_m} E'_{i_m}}{op(E_1, \ldots, E_n) \xrightarrow{\alpha} C[E'_1, \ldots, E'_n]} \tag{2}$$

Nella regola precedente, il termine $C[\ ]$ indica il nuovo contesto in cui il nuovo sottotermine opererà dopo la riduzione, e α rappresenta l'azione eseguita dal sistema composto quando i componenti effettuano le azioni $\alpha_1, \ldots, \alpha_m$.

Per comprendere meglio il funzionamento delle regole SOS, riportiamo alcuni esempi riguardanti le operazioni di somma e di composizione parallela visti nell'algebra di esempio (1).

$$\frac{E \xrightarrow{\alpha} E'}{E + F \xrightarrow{\alpha} E'} \quad \frac{F \xrightarrow{\alpha} F'}{E + F \xrightarrow{\alpha} F'} \tag{3}$$

$$\frac{E \xrightarrow{\alpha} E'}{E||F \xrightarrow{\alpha} E'||F} \quad \frac{F \xrightarrow{\alpha} F'}{E||F \xrightarrow{\alpha} E||F'} \tag{4}$$

Vediamo dunque come la somma sia interpretabile tramite le regole SOS (3) come la possibilità del programma $E + F$ di eseguire alternativamente il processo E o il processo F. La composizione parallela, invece, è interpretabile tramite le regole SOS (4) come la possibilità del programma $E||F$ di eseguire i processi E ed F allo stesso momento, interlacciando in modo arbitrario le loro azioni.

Una *semantica denotazionale* mappa un linguaggio su un modello astratto tale che il significato di qualunque programma composto sia determinabile direttamente dal significato delle sue componenti.

Una *semantica algebrica* è definita da un insieme di leggi algebriche che catturano implicitamente la semantica intesa dai costrutti del linguaggio in considerazione. Dunque, le leggi che permettono di interpretare il significato di un processo sono gli assiomi di base di un sistema di equazioni. Ad esempio, per interpretare la somma di processi attraverso una semantica algebrica, è necessario definire una serie di assiomi che specificano il suo comportamento

rispetto all'uguaglianza:

$$P + Q = Q + P \tag{5}$$
$$(P + Q) + R = P + (Q + R) \tag{6}$$
$$P + P = P \tag{7}$$

Questi assiomi specificano che la somma tra processi gode della proprietà commutativa, associativa e di idempotenza.

2.3 CCS

Delle varie algebre presentate, quella più affine ai nostri scopi è CCS, e, tra i suoi operatori, quelli che utilizzeremo maggiormente in seguito sono la composizione parallela e la somma. L'insieme A delle azioni di base usate in CCS è composto da un insieme Λ di etichette e un insieme $\bar{\Lambda}$ di etichette complementari. A_τ denota $A \cup \{\tau\}$. La sintassi di CCS è la seguente:

$$P ::= nil \mid x \mid \mu.P \mid P \setminus L \mid P[f] \mid P_1 + P_2 \mid P_1||P_2 \mid rec\, x.\, P \tag{8}$$

dove $\mu \in A_\tau$, $L \subseteq \Lambda$, $f : A_\tau \to A_\tau$, $f(\bar{\alpha}) = \overline{f(\alpha)}$ e $f(\tau) = \tau$. Gli operatori di cui sopra rappresentano:

- Il processo nullo nil
- Il prefisso di azione $\mu.P$
- La scelta mista $(+)$
- La composizione parallela binaria $(||)$
- La restrizione $(P \setminus L)$
- La rietichettatura $(P[f])$
- Le definizioni ricorsive $(rec\, x.\, P)$

La semantica operazionale di CCS è descritta dalle seguenti regole SOS, in cui con le lettere latine $a, b, c...$ si indicano le azioni visibili, ovvero quelle che possono essere osservate dall'ambiente esterno, come la stampa di una

stringa, mentre con τ si indicano le azioni invisibili, ovvero quelle che rappresentano operazioni interne al sistema, come ad esempio la comunicazione tra processi. Con μ indichiamo invece le azioni generiche.

$$ACT \quad \frac{}{\mu.P \xrightarrow{\mu} P} \tag{9}$$

$$SUM1 \quad \frac{P_1 \xrightarrow{\mu} P_1'}{P_1 + P_2 \xrightarrow{\mu} P_1'} \tag{10}$$

$$SUM2 \quad \frac{P_2 \xrightarrow{\mu} P_2'}{P_1 + P_2 \xrightarrow{\mu} P_2'} \tag{11}$$

$$COM1 \quad \frac{P \xrightarrow{\mu} P'}{P||Q \xrightarrow{\mu} P'||Q} \tag{12}$$

$$COM2 \quad \frac{Q \xrightarrow{\mu} Q'}{P||Q \xrightarrow{\mu} P||Q'} \tag{13}$$

$$COM3 \quad \frac{P \xrightarrow{a} P' \quad Q \xrightarrow{\bar{a}} Q'}{P||Q \xrightarrow{\tau} P'||Q'} \tag{14}$$

$$RES \quad \frac{P \xrightarrow{\mu} P'}{P \setminus L \xrightarrow{\mu} P' \setminus L} \quad \mu, \bar{\mu} \notin L \tag{15}$$

$$REL \quad \frac{P \xrightarrow{\mu} P'}{P[f] \xrightarrow{f(\mu)} P'[f]} \tag{16}$$

$$REC \quad \frac{P[rec\ x.P/x] \xrightarrow{\mu} P'}{rec\ x.P \xrightarrow{\mu} P'} \tag{17}$$

A partire da questa semantica, dunque, è possibile costruire un LTS per qualunque processo. Un semplice esempio, che rappresenta il LTS per il processo $P + Q$, con $P \equiv a.c.nil$ e $Q \equiv b.c.nil$ è il seguente:

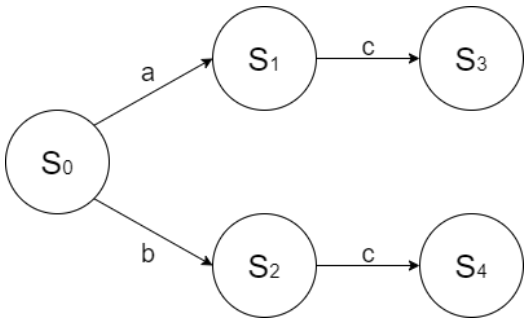

Figura 1: LTS per il processo $P + Q$

2.4 Ordini parziali e poset

Di fondamentale importanza è la nozione di *ordine parziale*, che verrà utilizzata molto spesso nella corrispondenza, e una definizione della quale forniamo qui sotto.

Un ordine parziale su un insieme S è la relazione $\leq$ su S che è riflessiva, anti-simmetrica e transitiva. La coppia $(S, \leq)$ è detta insieme parzialmente ordinato, o *poset*.

Dunque, in altre parole, un ordine parziale è tale perché non è necessariamente vero che, per ogni coppia di elementi x e y appartenenti all'insieme S,valga che $x \leq y$ o che $y \leq x$. Un ordine totale è un ordine parziale in cui qualunque coppia di elementi è comparabile.

Una definizione simile è quella di *pomset*, ovvero multi-insieme parzialmente ordinato. Tale termine è stato coniato da Pratt stesso in [32]. La differenza sostanziale tra un poset e un pomset è che il secondo è appunto un multi-insieme, ovvero è definito come una coppia (A, r) in cui A è un insieme e r è una funzione che associa ogni elemento di A a un numero naturale positivo che ne indica il numero di ripetizioni. Quindi, in un multi-insieme, ogni elemento può comparire più volte. Data questa definizione, Pratt asserisce in [30] che "possiamo chiamare un processo un insieme di pomset".

Dato un ordine parziale, una sua *estensione lineare* è un ordine totale che è compatibile con l'ordine parziale. Più precisamente, se $\leq$ è un ordine parziale e $\leq^*$ è un ordine totale, entrambi definiti sull'insieme S, $\leq^*$ è un'estensione di $\leq$ se per ogni $x, y \in S$, se $x \leq y$ allora $x \leq^* y$. In questo caso, si dice che l'ordine parziale è *contenuto* nell'ordine totale.

Dato un poset $(S, \leq)$, possiamo fornire una sua *linearizzazione*, ovvero un ordine totale degli elementi di S che rispetta la relazione $\leq$. Ogni poset

possiede almeno una linearizzazione, ma di solito quelle possibili sono multiple. Questo fenomeno è esplicitato dal Teorema di estensione di Szpilrajn, formulato appunto da Edward Szpilrajn in [35], il quale afferma che ogni ordine parziale è contenuto in un ordine totale ed è, inoltre, l'intersezione di tutte le sue estensioni totali.

2.5 Dualità tra schedule e automi

L'ambiente computazionale naturale per la vera concorrenza è quella delle *schedule*, o insiemi parzialmente ordinati di eventi. Per qualunque coppia di eventi, possiamo richiedere che uno di essi non inizi fino a che l'altro non è terminato, ovvero possiamo definire un *vincolo di precedenza*. Una collezione di vincoli simili compone una schedule [29].

Il modo in cui gli elaboratori sequenziali lavorano durante una transizione e poi si fermano su uno stato in modo alternato è solitamente rappresentato da un automa, un grafo i cui vertici rappresentano gli stati e gli archi sono transizioni da uno stato all'altro, eventualmente etichettati da attributi delle transizioni stesse. Possiamo definire dei vincoli di precedenza $i < j$, che indicano che l'evento i deve necessariamente accadere prima dell'evento j, quindi che j dipende causalmente da i, e compongono dunque un ordine parziale dell'insieme (o multi-insieme) di d eventi. Esiste una dualità tra le schedule e gli automi, per cui, per almeno una certa classe S di schedule e una classe A di automi, essi rappresentano la stessa scena perché S ed A sono equivalenti, nel senso che esiste una corrispondenza uno a uno tra le classi di isomorfismo di S e A. In altre parole, è sempre possibile passare da una schedule a un automa equivalente e viceversa.

Consideriamo un esempio di questa corrispondenza tra schedule e automi con un esempio che è direttamente rilevante per il tema di questa tesi. Si consideri la schedule in Figura 2 composta da due eventi inconfrontabili e quindi causalmente indipendenti.

Figura 2: Una schedule composta da due eventi indipendenti

Dato uno stato iniziale in cui nessuno dei due eventi è ancora accaduto, che possiamo identificare con l'insieme vuoto di eventi, l'accadere di a porta

a uno stato $\{a\}$ in cui a è accaduto, e analogamente per l'evento b. Abbiamo a questo punto la situazione seguente:

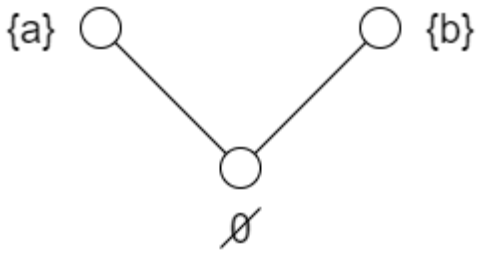

Figura 3: Diagramma parziale relativo alla schedule

Naturalmente, data l'indipendenza causale dei due eventi, dopo che uno di essi è accaduto l'altro può ancora accadere, e il diagramma precedente[2] si può completare nel seguente modo:

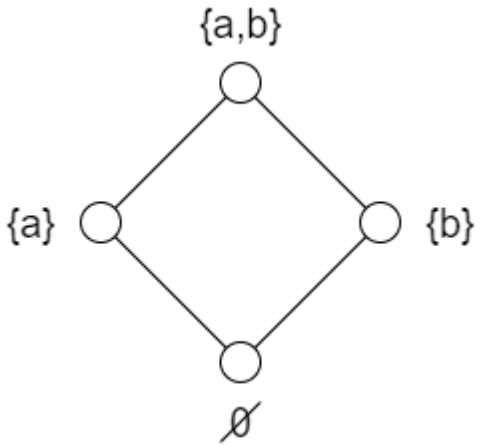

Figura 4: Diagramma completo relativo alla schedule

Questo insieme parzialmente ordinato consiste di quattro stati; le sue transizioni si identificano con gli eventi che fanno passare da uno stato al

[2]Si tratta infatti di un diagramma di Hasse dell'insieme parzialmente ordinato degli stati dell'automa associato alla schedule, che registrano gli eventi accaduti fino ad un certo istante dell'esecuzione del processo.

successivo. L'automa corrispondente alla schedule originale può essere infine rappresentato così:

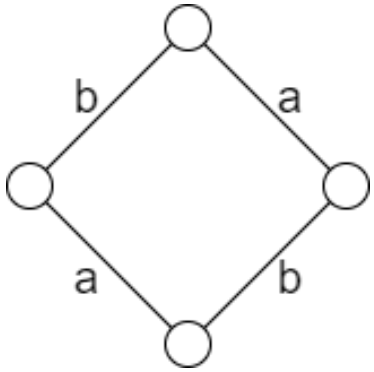

Figura 5: Automa relativo alla schedule

Questa costruzione illustra, in un caso particolarmente semplice, quella che Pratt [29] chiamerà, subito dopo il (e forse in conseguenza del) dibattito discusso in questa tesi, la *dualità automa-schedule*. Su questa si innesterà un'ulteriore osservazione secondo la quale, all'interno di Figura 4 e Figura 5, è possibile vedere un ulteriore elemento che rappresenta l'esecuzione concorrente di a e b.

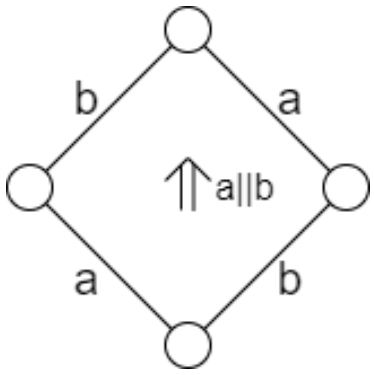

Figura 6: Automa a 2 dimensioni relativo alla schedule

Il nuovo elemento ha dimensione 2, se gli stati (rappresentati da punti) hanno dimensione 0, e le transizioni (rappresentati da linee) hanno dimensione 1: si identifica infatti $a||b$ con il rombo delimitato dai due semiperimetri che descrivono l'esecuzione interlacciata di a e b.

La definizione di questi *automi di dimensione superiore* dovuta a Pratt [29] e lo studio delle loro proprietà sarà lo sviluppo naturale del dibattito descritto nelle pagine seguenti.

È inoltre possibile definire una relazione binaria, simmetrica e non riflessiva $i\#j$ sugli eventi, chiamata *conflitto* e il cui significato è che i e j non possono essere eseguite entrambe. In una schedule, richiediamo che $i\#j$ e $j < k$ implichino che $i\#k$, cioè il conflitto è una condizione persistente. Oltre ai già citati poset, dunque, è possibile definire una struttura simile, chiamata *struttura di eventi* [25], come $(V, <, \#)$, dove V è l'insieme degli eventi.

2.6 Equivalenze Comportamentali

Infine, è importante specificare che uno degli obiettivi principali delle algebre dei processi è quello di dimostrare che due processi sono tra di loro equivalenti, ovvero se alcuni aspetti dei loro comportamenti osservabili esternamente sono compatibili. In questo contesto, esistono molte diverse nozioni di equivalenza, che sono suddivisibili in tre categorie: le equivalenze di traccia, le equivalenze di traccia decorata e le equivalenze di bisimulazione[3]. Esse, rispettivamente, considerano equivalenti i sistemi che:

- eseguono le stesse sequenze di azioni
- eseguono le stesse sequenze di azioni e dopo ogni sequenza sono pronti ad accettare lo stesso insieme di azioni
- eseguono le stesse sequenze di azioni e dopo ogni sequenza mostrano, ricorsivamente, lo stesso comportamento

Prendiamo in considerazione, ad esempio, i tre LTS in Figura 7 che rappresentano dei distributori automatici che ricevono input due monete e danno in output un caffè oppure un tè.

Ebbene, le equivalenze di traccia non distinguono alcuna delle macchinette, quelle di traccia decorata distinguono solamente la prima dalle altre due, mentre quelle di bisimulazione le distinguono tutte e tre.

Tra queste, la bisimulazione è l'equivalenza più fine e più spesso citata. Ne forniamo dunque una definizione formale: Una relazione $R \subseteq Q \times Q$ è

[3]Le varie equivalenze sono approfondite nel dettaglio in [8]

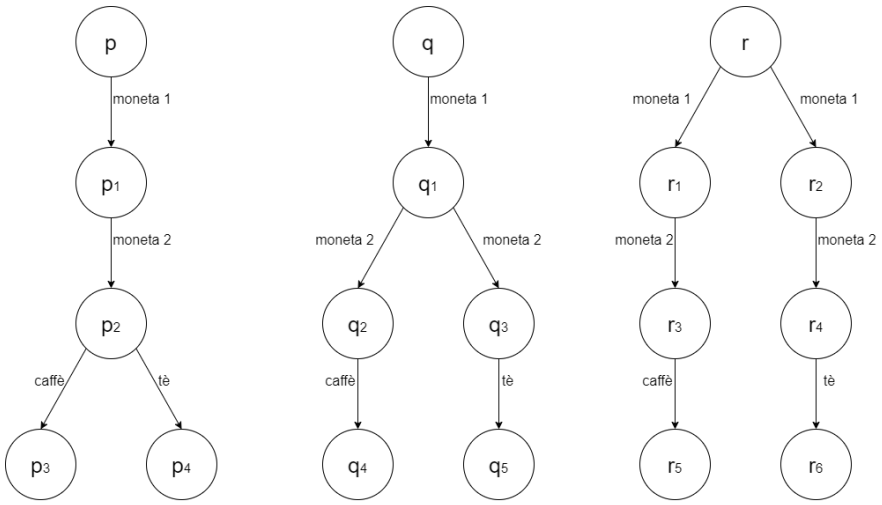

Figura 7: I tre distributori automatici

una *bisimulazione* se, per qualunque coppia di stati p e q tali che $\langle p, q\rangle \in R$, vale che:

1. per ogni $\mu \in A_\tau$ e $p \in Q$, se $p \xrightarrow{\mu} p'$ allora $q \xrightarrow{\mu} q'$ per qualche $q' \in Q$ t.c. $\langle p', q'\rangle \in R$;

2. per ogni $\mu \in A_\tau$ e $q \in Q$, se $q \xrightarrow{\mu} q'$ allora $p \xrightarrow{\mu} p'$ per qualche $p' \in Q$ t.c. $\langle p', q'\rangle \in R$;

dove A_τ rappresnta l'insieme $A \cup \{\tau\}$. Due stati p e q sono *bisimili* ($p \sim q$) se esiste una bisimulazione R tale che $\langle p, q\rangle \in R$.

Una definizione alternativa è la seguente:

Due stati $p, q \in Q$ sono bisimili ($p \sim q$) se e solo se per ogni $\mu \in A_\tau$:

1. se $p \xrightarrow{\mu} p'$ allora $q \xrightarrow{\mu} q'$ per qualche q' tale che $p' \sim q'$;

2. se $q \xrightarrow{\mu} q'$ allora $p \xrightarrow{\mu} p'$ per qualche p' tale che $p' \sim q'$.

3 I difetti degli ordini lineari

Il 21 Ottobre del 1990, David Luckham invia sulla concurreny mailing list un messaggio indirizzato a Vaughan Pratt. Luckham afferma di aver incontrato, parlando con alcuni colleghi, una tendenza a considerare gli insiemi di

tracce lineari "del tutto sufficienti per analizzare le computazioni distribuite o concorrenti" e che "gli ordini parziali non sono necessari". Pone quindi a Pratt due domande:

1. Quale sia il suo esempio preferito di un sistema in cui una rappresentazione con ordine parziale delle esecuzioni è superiore all'utilizzo di un insieme di tracce lineari

2. Se egli sia in disaccordo con le affermazioni dei colleghi di Luckham, e in che modo

Lo stesso giorno, Pratt risponde sostenendo che la convinzione che gli ordini lineari catturino quelli parziali dipenda da una moltitudine di premesse, la maggior parte delle quali deve valere contemporaneamente per essere affidabili. Tali presupposti valgono per sistemi molto semplici o astratti, ma si disperdono man mano si trattano sistemi più complessi e concreti. Le premesse individuate da Pratt sono sette:

1. Granularità fissa

2. Nessuna variabilità degli eventi atomici

3. Assenza di Autocorrenza

4. Processi single-poset

5. Assenza di competizioni

6. Modello a singolo osservatore

7. Tempo discreto

Con granularità si intende la quantità di lavoro che un processo svolge, ovvero il numero di istruzioni eseguite in un certo tempo [16]. La granularità variabile può insorgere in diversi modi, ad esempio analizzando un'azione atomica più a fondo e determinando che in realtà è suddivisibile in più azioni. Questo metodo è chiamato "raffinamento". Un altro modo è quello di prendere un programma le cui specifiche lo trattano come atomico e scoprire, eseguendolo, che ha una serie di effetti collaterali sul sistema, che potrebbero interlacciarsi con gli effetti collaterali di programmi simili. Per poter utilizzare gli ordini lineari, dunque, è necessario assumere che la granularità non si riveli mai essere variabile.

La variabilità degli eventi atomici indica il fenomeno per cui un evento, pur essendo atomico, se eseguito più volte non produce sempre esattamente gli stessi risultati.

L'autocorrenza è invece definita da Pratt come la presenza, in un sistema, di due eventi identici e concorrenti. Egli offre anche un esempio molto chiaro, in cui si immagina di chiedere due dollari a un cassiere: "Se i dollari arrivano sempre in modo sequenziale, allora non c'è dubbio sulla legittimità della stringa 11 come specifica per 'due dollari'. Ma che dire invece di $1||1$? [...] Con l'autocorrenza si può avere $a||a$, che le tracce non possono distinguere da aa". Questa situazione è risolvibile tramite la raffinamento delle azioni, ma anche così non possiamo distinguere, nel contesto delle tracce, il processo $TR||TR$, ovvero due esecuzioni parallele di T seguito da R, dallo stesso processo modificato in modo tale che una delle T debba sempre precedere entrambe le R.

Come abbiamo visto in precedenza, un processo può essere visto come un insieme di pomset. Un processo single-poset è dunque uno che è definito da un singolo poset. Il teorema che identifica i poset con la loro linearizzazione è basato sulla pre-condizione che un processo sia, appunto, single-poset, ma questo presupposto è raramente ottenibile nella pratica, ed è falso che un insieme di poset possa essere visto come l'unione dei rispettivi insiemi di linearizzazioni.

Una *competizione* si verifica quando il risultato dell'esecuzione di più processi all'interno di un sistema dipende dall'ordine in cui i processi stessi vengono eseguiti. Quando a e b sono in una competizione, il modello a tracce rileva solo $ab + ba$, ma il non-determinismo senza competizioni, che sceglie uno tra ab e ba, ha la stessa rappresentazione a tracce. Infatti, il modello a tracce è basato sul fatto che l'esecuzione mutualmente esclusiva e l'esecuzione concorrente siano la stessa cosa, per gli eventi atomici.

La maggior parte dei modelli della concorrenza assume che tutte le osservazioni siano effettuate da un solo osservatore, ma in realtà questi sono spesso molteplici, distribuiti tanto quanto il sistema che osservano, e possono unire le loro osservazioni in modi del tutto scollegati dal modello computazionale usato per provare la correttezza del sistema. In [27], Pratt e Plotkin entrano nel dettaglio di questo argomento, dimostrando come un gruppo di osservatori sequenziali che collaborano tra di loro possono distinguere qualunque pomset. In particolare, un gruppo di k osservatori è in grado di distinguere tutti i pomset di dimensione pari a k, dove la dimensione di un pomset è il numero minimo di linearizzazioni di quel pomset la cui intersezione è il

pomset stesso. L'esistenza di queste linearizzazioni è garantita dal Teorema di Szpilrajn.

Infine, il tempo deve essere discreto perché le tracce possano modellare l'interleaving, ma ovviamente il tempo reale non è discreto. Ad esempio, quale sarebbe esattamente l'insieme di tutti gli interlacciamenti di due copie dell'intervallo $[0, 1]$?

Pratt prosegue ragionando riguardo la naturalezza dei modelli. A suo parere, infatti, "i modelli dovrebbero cercare di essere ragionevolmente fedeli a ciò che modellano, se la matematica lo permette. Anche se il tuo modello innaturale funziona oggi, la mia sensazione è che i modelli innaturali siano più predisposti a fallire in futuro rispetto a quelli naturali". Come esempio, prendiamo due computer, uno negli Stati Uniti e uno in Europa, che comunicano tra di loro. In questo contesto, il tempo che intercorre tra due istruzioni in un computer è molto minore a quello necessario perché un'informazione venga passata da uno all'altro. Un modo naturale di modellare gli stream di istruzioni delle due macchine sarebbe quello di usare due sequenze separate, ma questo il modello delle tracce non lo permette, in quanto richiede che si debbano interlacciare le due sequenze in ogni modo possibile prima che si possa ragionare sul sistema.

In conclusione, Pratt critica il modo in cui l'informatica viene insegnata: molti studiosi sono esposti quotidianamente solo alla teoria della computazione sequenziale. Perciò, quando incorrono nella concorrenza, fanno l'unica cosa possibile: usano l'interleaving per ridurre il problema a un modello a loro noto.

> Posso dire, sulla base dell'aver lavorato con entrambi i modelli per molti anni, che i poset sono molto più flessibili e facili da usare rispetto alle tracce.

4 I vantaggi degli ordini parziali

Il 22 Ottobre, Rance Cleveland solleva un argomento che verrà a lungo discusso nelle mail successive: assumiamo di stare analizzando un sistema in cui un'azione atomica qualsiasi impieghi un'unità di tempo per essere eseguita. Allora, il processo $a||b$, ovvero l'esecuzione veramente parallela di a e b, dovrebbe impiegare anch'essa una sola unità di tempo, mentre $ab + ba$, cioè l'esecuzione di a seguito da b o di b seguito da a, dovrebbe impiegarne due. Tuttavia, il modello basato sulle tracce lineari, riterrebbe i due processi equivalenti, pur avendo evidentemente un tempo di esecuzione diverso. Cleveland è dunque stupito che delle persone interessate ai sistemi real-time possano considerare la linearizzazione un buon modello per rappresentare la concorrenza.

Pratt sfrutta l'intervento di Cleveland per chiarire il motivo per cui quelli che definisce "gli hacker della vera concorrenza" preferiscono i poset rispetto agli automi, quindi ai LTS: questi ultimi sono monodimensionali, e quindi possono esibire solo la struttura della concorrenza interleaving. Tale limitazione è esplorata da Pratt stesso in [29]. Qui, egli propone un modello geometrico in grado di rappresentare la vera concorrenza: i sistemi di transizione sono trattati come uno spazio a una dimensione composto da spigoli, cioè le transizioni, che si incontrano in vertici, cioè gli stati. La vera concorrenza tra n processi è rappresentata dalle dimensioni superiori: una cella n-dimensionale è utilizzata per rappresentare l'esecuzione concorrente di n processi sequenziali.

Un esempio dell'utilizzo di tale costrutto è quello presentato nel capitolo 2.5, in cui viene utilizzato lo spazio bidimensionale all'interno di un automa per rappresentare l'esecuzione concorrente di due eventi a e b. E' possibile fare lo stesso con più processi, ottenendo figure geometriche di dimensioni superiori. In ognuna di queste figure, il numero totale di celle è pari a 3^d, dove d è il numero di dimensioni della figura. Per l'esempio precedente, infatti, esistono in totale nove celle: quattro rappresentano gli stati, quattro le transizioni in cui uno solo dei due automi è attivo, e uno la transizione in cui entrambi gli automi sono in esecuzione, quindi la vera concorrenza. Il numero tre nasce dai possibili stati di ciascuna azione di un evento: iniziale, in transizione, finale, o $0, T, 1$. Ogni cella può perciò essere rappresentata come una d-tupla su $\{0, T, 1\}$. Dunque, per poter rappresentare la vera concorrenza al pari dei poset, gli automi devono essere espansi per essere pluridimen-

sionali. Pratt, negli anni successivi, tornerà nuovamente a trattare questo argomento in [28]. In questo articolo, egli identifica tre diversi metodi per modellare la programmazione concorrente. Il primo è quello della scultura: qualunque processo può essere concepito come l'insieme A di tutti gli eventi che è in grado di eseguire, e da qualche sottoinsieme dell'insieme $\{0, T, 1\}^A$ che costituisce i possibili stati del processo. A partire dal cubo così creato si "intaglia" il processo desiderato eliminando gli stati superflui, in modo simile a uno scultore con un blocco di marmo. Questo metodo, pur essendo concettualmente semplice, non è abbastanza per specificare un automa concorrente, sono necessari altri due approcci: la composizione, attraverso la quale processi complessi sono costruiti a partire da processi più piccoli con appositi operatori, tra cui la composizione parallela, e la trasformazione, che permette di modificare in modo appropriato processi già esistenti per costruirne di nuovi. Queste tre attività, scultura, composizione e trasformazione, sono simultaneamente compatibili e complementari, e possono essere usate come base per la programmazione concorrente.

A favore dei poset interviene anche Eike Best, il quale sostiene che talvolta gli ordini parziali permettono di definire un concetto più facilmente rispetto all'interleaving. E' ad esempio il caso del "ritardo finito", che significa "una transizione non può essere abilitata per sempre senza che essa occorra effettivamente" [18]. In un sistema sequenziale, il ritardo finito può essere espresso dalla massimalità di una sequenza di esecuzione. Consideriamo $a^*||b^*$ e $(a[\,]b)^*$, dove $[\,]$ è la scelta non-deterministica. La sequenza composta da infinite volte a contraddice la proprietà di ritardo finito in $a^*||b^*$. Infatti, essendo $a^*||b^*$ l'interleaving di a^* e b^*, quindi le stringhe composte rispettivamente da infinite occorrenze di a e infinite occorrenze di b, ad ogni passo un'azione b può accadere. Al contrario, la stessa stringa non contraddice la proprietà in $(a[\,]b)^*$, in quanto l'occorrenza di a è sempre alternativa a b. Si può comprendere meglio la distinzione notando che la stringa composta da infinite a, pur essendo massimale come stringa, non è massimale come ordine parziale di $a^*||b^*$, ma lo è come ordine parziale di $(a[\,]b)^*$. C'è però da dire che quello riportato da Best è un esempio di difficile interpretazione.

5 Ordini lineari contro ordini parziali o interleaving contro true concurrency?

Il 26 Ottobre, Albert R. Meyer indirizza la conversazione verso una nuova direzione: il vero problema non riguarda le differenze tra ordini lineari e ordini parziali, bensì quelle tra l'interleaving non-deterministico e la vera concorrenza. La diatriba tra ordini lineari e parziali rappresenta un dettaglio trascurabile, in quanto si concentra sul modo in cui i processi debbano essere rappresentati in modo astratto. Al contrario, discutere le differenze tra interleaving e true concurrency porta a individuare le limitazioni di modelli di concorrenza a interleaving quali CCS, CSP e ACP, già molto usati in vari campi.

Meyer ritiene che l'equazione $a||b = ab + ba$, identificata precedentemente da Cleveland, sia il cuore delle critiche mosse all'interleaving. Essa è infatti un assioma delle teorie basate sull'interleaving, ma porta a escludere dalle stesse alcuni concetti fondamentali:

1. Osservazioni di simultaneità: a e b possono essere osservati simultaneamente nella computazione di $a||b$, ma non in $ab + ba$.

2. Osservazioni della stessa computazione da parte di due o più osservatori sequenziali in locazioni distribuite: due osservatori di questo tipo che guardano una computazione di $a||b$ potrebbero notare tracce lineari differenti, in particolare uno potrebbe vedere ab nello stesso intervallo di tempo in cui l'altro vede ba, ma gli stessi osservatori vedrebbero sempre la stessa traccia in una qualunque computazione di $ab + ba$, cioè esattamente una tra ab e ba.

3. Raffinamento delle azioni atomiche: ciò che Pratt ha in precedenza chiamato "granularità variabile". Rifinire a in $a||b$ per farlo diventare il processo sequenziale cd porta a un processo con la traccia cbd, mentre fare lo stesso in $ab + ba$ non porta a tale traccia.

Queste proprietà sono molto importanti, così come le idee di durata delle azioni, di regione critica e di atomicità. E' dunque sensato, secondo Meyer, ritirarsi del tutto dal modello a interleaving semplice, o meglio, modificarlo in modo tale da poter inserire queste estensioni.

Allo stesso tempo, però, Meyer ritiene che neanche i modelli a pomset proposti da Pratt siano del tutto adatti per modellare la concorrenza.

Inoltre, i vari modelli proposti sono tutti basati su nozioni generalizzate di bisimulazione, e per Meyer questo tipo di equivalenza così fine non può essere giustificata, dal punto di vista computazionale. Egli ammette, in ogni caso, che modellare l'esecuzione concorrente tramite pomset risulti alquanto naturale.
Pratt, il giorno seguente, risponde a Meyer con alcuni commenti:

- Quando Meyer afferma che ogni ordine parziale è determinato unicamente dall'insieme delle sue linearizzazioni, egli si riferisce al già citato teorema di Szpilrajn, il quale è in realtà un teorema "fragile", dove un teorema "robusto" su una struttura resta vero quando si aggiungono dettagli alla struttura. Infatti, il teorema di Szpilrajn non vale né per un insieme di poset, né per un poset etichettato, ed entrambe queste strutture sono necessarie per rendere il concetto di poset utile per la modellazione di concorrenza.

- Riguardo l'ormai famosa equazione $a||b = ab + ba$, Pratt espone una proprietà interessante delle espressioni regolari: la loro logica non cambia quando si trattano le loro variabili come costanti che denotano sé stesse. Questa proprietà, tuttavia, cade non appena si aggiunge quasi qualunque altra operazione, indipendentemente che essa preservi o no la regolarità dell'espressione. Tali operazioni includono anche l'interlacciamento, per cui Pratt propone di riscrivere $a||b = ab + ba$ come implicazione condizionale:

  $$atomic(a)\ \&\ atomic(b) \rightarrow a||b = ab + ba \tag{18}$$

  o, più in generale,

  $$atomic(a)\ \&\ atomic(b) \rightarrow mutex(a, b) \tag{19}$$
  $$mutex(a, b) \rightarrow a||b = ab + ba \tag{20}$$

  dove *mutex* indica la proprietà di due azioni di essere mutualmente esclusive.

- Per quanto riguarda, invece, i dubbi di Meyer sulla bisimulazione, Pratt ammette di non essere qualificato quanto Meyer in questo campo, ma ritiene intuitivamente che la logica di Hennessy-Milner [12], utilizzata per specificare proprietà dei LTS e che giustifica tutte le distinzione effettuate dalla bisimulazione, non sia un linguaggio eccessivamente forte

nel contesto del debugging, in cui il programmatore va avanti e indietro in una computazione non-deterministica cercando ciò che ha causato un malfunzionamento e sperimentando facendo piccoli cambiamenti.

6 I pionieri della vera concorrenza

In una mail del 26 Ottobre, Ramu Iyer propone alcuni testi che identifica come pionieri nel campo della vera concorrenza. Essi sono:

- "Concurrency vs Interleaving: An Instructive Example" di Castellano, De Michelis e Pomello, del 1987 [5].
- "Concurrency and Interleaving are Equally Fundamental" di Benson, del 1987 [1].
- "Concurrency is More Fundamental than Interleaving" di Reisig, del 1988 [33]

E' interessare notare come, sin dagli albori di questo dibattito, ci sia stata una certa discordanza tra le opinioni degli studiosi; basti considerare la chiara somiglianza, e allo stesso tempo contrapposizione, tra i titoli degli articoli di Benson e Reisig.

In [5], vengono classificati i vari approcci alla concorrenza in due gruppi: quelli basati sulle semantiche interleaving, cioè che riducono la concorrenza alla sua simulazione sequenziale non-deterministica, e quelli basati sulle semantiche degli ordini parziali, in cui la concorrenza è modellata tramite l'indipendenza causale. Viene inoltre proposto un esempio di raffinamento di azione, usando $a||b = ab + ba$ come contesto. La conclusione a cui arrivano gli autori, identificata anche in una mail precedente da Pratt, è che la semantica interleaving è adeguata solo se il livello di astrazione al quale le azioni atomiche di un sistema distribuito sono definite è fisso; altrimenti si dovrebbero considerare le semantiche a ordini parziali.

Pratt ribatte proponendo un testo ancora precedente, ovvero la tesi di Ph.D di Irene Greif "Semantics of Communicating Parallel Processes" datata 1975 [11], in cui, appunto, viene sviluppato un linguaggio di specifica utilizzando per la prima volta gli ordini parziali per ordinare gli eventi del sistema. Esempi di utilizzo degli ordini parziali di poco successivi alla tesi di Greif sono alcuni articoli di Jan Grabowski e di Nielsen, Plotkin, Winskel,

del 1981 [25][10]; inoltre si dice che già Carl Adam Petri avesse sostenuto tempo addietro l'uso degli ordini parziali nel campo della concorrenza[4].

Pratt ammette di non aver egli stesso apprezzato i vantaggi offerti dagli ordini parziali fino al 1980, anno in cui si pone l'obiettivo di comprendere l'anomalia di Brock-Ackerman [4]. Si tratta di un'anomalia nel comportamento di due processi posti all'interno dello stesso sistema: prendiamo due processi $P1$ e $P2$, e un sistema S che può interagire con essi. Se osserviamo la mappatura da sequenza di input a sequenze di output dei due processi, essi risultano equivalenti, ma quando vengono inseriti all'interno del sistema S, si comportano in modo differente. Questo mostra che le mappature da input a output non sono sufficienti a determinare il comportamento composizionale di processi in un sistema concorrente. In altre parole, se consideriamo un processo concorrente come scatola nera, il semplice comportamento input-output non è abbastanza per ragionare sul comportamento del processo in un sistema più grande.

In seguito ai suoi studi riguardo l'anomalia, Pratt scrive "On the Composition of Processes" [31], in cui propone di formalizzare la soluzione di Brock e Ackerman alla loro anomalia in termini di multi-insiemi parzialmente ordinati, poi battezzati da Pratt stesso come "pomset". Nel 1986 viene anche pubblicato un suo articolo del tutto dedicato all'utilizzo dei pomset nel campo della concorrenza, il già citato "Modeling Concurrency with Partial Orders" [30]. In questo scritto, Pratt porta vari argomenti a favore dei pomset, alcuni dei quali già discussi all'interno delle mail precedenti. I principali sono i seguenti:

Le stringhe, viste come multi-insiemi linearmente ordinati di simboli di un qualche alfabeto, sono una tipologia di modellazione molto naturale per rappresentare la computazione sequenziale, se si intendono i simboli della stringa come stati, comandi e messaggi. Secondo Pratt, gli ordini parziali sono più efficaci per la modellazione di computazioni concorrenti, tuttavia quelli lineari sono più popolari a causa di alcuni fattori:

1. I linguaggi e le loro operazioni offrono un modello naturale per le corrispondenti strutture di controllo di un linguaggio di programmazione, e forniscono un modello di computazione familiare e ben conosciuto. In

[4]In una mail successiva, Eike Best citerà alcuni riferimenti importanti per la semantica a ordini parziali, tra cui [26], [13] e [3]. Fondamentale anche [15], che ha introdotto le reti di Petri negli Stati Uniti e ha definito costruzioni, come i grafi di occorrenze, che anticipano i modelli basati sugli ordini parziali.

questo modello, l'ordine lineare sugli elementi di una stringa è interpretato come l'ordine lineare temporale degli eventi, e le operazioni sui linguaggi possono essere interpretate come strutture di controllo.

2. Ogni poset è rappresentabile come l'insieme delle sue linearizzazioni

3. Gli ordini lineari appaiono fedeli alla realtà fisica, e qualunque scostamento dalla rigidità del sistema è considerato abbastanza secondario, nella pratica, da giustificare l'aderenza a un modello di ordine lineare.

Pratt offre delle controargomentazioni a tutti e tre i punti precedenti, in più aggiungendo motivi per preferire gli ordini parziali.

1. Il primo punto perde di forza quando si considera che l'articolo stesso offre operazioni analoghe a quelle dei linguaggi formali, le quali possono anch'esse venire interpretate come strutture di controllo di un linguaggio di programmazione.

2. Il secondo punto incorre in un problema quando si cerca di applicarlo ai processi modellati come insiemi di pomset, infatti non è generalizzabile per insiemi di pomset né di poset

3. L'idea che il mondo dell'ingegneria abbia un tempo lineare fallisce in almeno tre situazioni: sistemi complessi, eventi non atomici e sistemi relativistici.

    - Oltre un certo livello di complessità, diventa impensabile continuare a ragionare in termini di un orologio globale e di una sequenza lineare di eventi. Quando consideriamo un grande numero di computer che comunicano tra loro su canali il cui ritardo è di ordini di grandezza superiore all'orologio interno di ognuno dei computer, il concetto di tempo globale risulta non solo inaffidabile ma inutile. Non c'è motivo di interlacciare gli stream dei vari computer in uno solo, è molto più conveniente metterli uno accanto all'altro e considerare questa giustapposizione di stream un modello delle loro esecuzioni concorrenti.
    - Un evento non è necessariamente istantaneo: puotrebbe essere definito su un intervallo di tempo, un insieme di intervalli o in generale un insieme arbitrario di momenti, e tali eventi non possono essere ordinati in modo lineare.

- In un sistema non rigido, in cui i componenti si muovono uno rispetto all'altro, la simultaneità smette di esser ben definita, e due osservatori in movimento possono osservare ordini di occorrenza in contraddizione uno con l'altro. In generale, quindi, qualunque sistema soggetto ad effetti relativistici dovrebbe essere modellato tramite ordini parziali.

4. Il primo dei motivi per cui adottare gli ordini parziali è che alcuni concetti sono definibili solo per gli ordini parziali. Un esempio è l'ortocorrenza, che equivale al prodotto diretto di pomset, ma che non è definibile utilizzando gli ordini lineari.

5. Nel modello a interleaving, cosa esattamente è interlacciato dipende da quali eventi si considerano atomici. Dati due processi P e Q, dove P è composto dall'evento a e Q dall'evento b, il loro interlacciamento è $\{ab, ba\}$. Tuttavia, se gli stessi eventi a e b vengono visti da qualche altro osservatore come non atomici, quindi con dei sotto-eventi, allora l'interlacciamento di P e Q diventa più complesso. Nel modello degli ordini parziali, il fatto che due eventi siano concorrenti o meno non dipende dalla granularità delle azioni atomiche.

6. Infine, molto semplicemente, in alcune situazioni è più conveniente ragionare utilizzando i pomset piuttosto che le stringhe. Ad esempio, Pratt considera relativamente semplice l'assiomatizzazione della teoria equazionale dei pomset sotto le operazioni di concorrenza e concatenazione, ma la teoria corrispondente per le stringhe non è ancora mai stata assiomatizzata.

7 I filosofi della vera concorrenza

Finalmente, il 6 Novembre 1990, Leslie Lamport scrive una mail in cui non solo si schiera strettamente a favore dell'interleaving, ma critica l'atteggiamento degli altri membri della mailing list, e in particolare quello di Pratt. La mail si apre con alcune righe in cui Lamport, con grande ironia, commenta:

> Ammiro i filosofi. Hanno così tanto da insegnarci. Da Aristotele ho imparato che i corpi pesanti cadono più velocemente di quelli leggeri; Kant mi ha mostrato che la geometria non euclidea è impossibile; e Spinoza ha dimostrato che possono esistere al più sette pianeti. E ora, i filosofi sulla mailing list della concorrenza mi hanno detto tutte le cose che non posso fare perché uso una logica basata su un modello interleaving.

Lamport prosegue elencando tutte le cose che i suoi colleghi hanno ritenuto impossibili per qualcuno che utilizzi il modello interleaving, e offrendo, per ognuna, degli esempi di lavori che egli stesso ha svolto proprio utilizzando modelli interleaving:

- Non è possibile ragionare sui sistemi distribuiti: Lamport cita il suo articolo "An Assertional Correctness Proof of a Distributed Algorithm" [19], in cui porta una dimostrazione dell'algoritmo distribuito allora usato nell'Arpanet per mantenere le sue tabelle di routing.

- Non è possibile gestire cambiamenti nella granularità delle azioni atomiche: nell'articolo "Specifying Concurrent Program Modules" [20], Lamport fornisce una specifica di una coda in cui aggiungere o rimuovere un elemento è un'azione atomica, un'implementazione della suddetta coda in cui un elemento è inserito e rimosso dalla coda bit per bit, e una dimostrazione del fatto che l'implementazione rispetti la specifica.

- Non è possibile ragionare sul tempo reale non discreto: in un workshop del 1988, Lamport dà una specifica di un algoritmo distribuito in cui la computazione raggiunge e mantiene una configurazione corretta entro un certo tempo reale. Implementa inoltre l'algoritmo usando dei timer, e dimostra che l'implementazione soddisfa la specifica.

- Non è possibile ragionare su programmi senza assumere una granularità fissa: Lamport porta come prova un suo articolo molto recente [22], in

> cui dà una dimostrazione di correttezza rigorosa per il suo algoritmo del fornaio, il quale non assume nulla riguardo la granularità delle proprie operazioni.

La righe conclusive della mail sono nuovamente ironiche:

> Sono sicuro che i filosofi sappiano spiegare perché io non abbia veramente fatto queste cose. Sarò felice di ascoltare le loro spiegazioni, non appena sapranno usare i loro metodi filosoficamente approvati per ragionare formalmente su qualcosa di più complicato di una macchina per biscotti.

8 Teoria e pratica a confronto

Pratt, ai commenti sarcastici di Lamport, risponde nella maniera pacata che lo contraddistingue, senza affrontare in modo diretto i punti sollevati ma citando invece una discussione tra Amir Pnueli e Istvan Nemeti, tenutasi alla conferenza Logics of Programs 81. Pnueli aveva espresso il desiderio che ogni articolo sulla logica di programmazione dicesse anche qualcosa riguardo a come tale logica fosse da applicare nell'effettuare dimostrazioni di programmi, al che Nemeti aveva commentato che la struttura della società tecnologica semplicemente non funziona così:

> C'era un tale di nome Roentgen. Saresti potuto andare da lui e dire "perché traffichi con questi tuoi buffi aggeggi? Perché invece non provi a curare le persone che hanno il raffreddore?"

Questo per dire che nel mondo della scienza esistono sia teorici che tecnologi, nonché tutto uno spettro di figure intermedie, e i primi sono necessari ai secondi, perché le idee di base che impattano poi l'aspetto prettamente pragmatico trattato dei tecnologi proviene dalla teoria pura.

Robert J. Hall non è però convinto che questa citazione affronti direttamente i dubbi sollevati da Lamport: infatti egli aveva sostenuto che i teorici della mailing list avessero affermato delle falsità, e che avesse "suggerito in modo fraterno" che, per evitare di dire imprecisioni, si potesse cercare di mantenere un più stretto contatto tra teoria e pratica.

Pratt, tuttavia, rimane della propria opinione, e anzi, ritiene di aver affrontato le lamentele di Lamport nel modo più diretto possibile, in quanto

egli non ha puntato a nessuna affermazione specifica ma ha fatto un discorso generico. Dei molti argomenti trattati nelle mail incriminate, nessuno riguarda ciò di cui Lamport si lamenta, né è in conflitto con le prove che egli ha portato per supportare i traguardi da lui raggiunti. Inoltre, temi quali le criticità del teorema di Szpilrajn non sono solo curiosità matematiche, ma potenzialmente dei veri e propri problemi ingegneristici. Pratt propone dunque nuovamente i propri quesiti: come può una logica basata sugli insiemi di tracce:

1. Distinguere la competizione implicita in $a||b$ dalla situazione senza competizione implicita in $ab + ba$?

2. Ragionare sulle osservazioni effettuate da un gruppo di osservatori distribuiti che sono d'accordo su quali eventi sono accaduti ma non sul loro ordine?

3. Discutere i possibili interlacciamenti di due onde sinusoidali?

La risposta di Lamport è di nuovo sarcastica e graffiante: egli fa finta di scrivere a Roentgen stesso, dicendogli quanto sia triste che ci siano dei ciarlatani che hanno usato i suoi raggi X per giustificare teorie su "raggi invisibili" basate sulla fantasia piuttosto che sulla scienza, e chiedendosi quando i Sovietici faranno finalmente della ricerca valida nella genetica, dato che adesso "danno più valore alla correttezza filosofica che all'osservazione empirica", chiaramente riferendosi al modo di ragionare teorico di Pratt.

A questo secondo commento ironico di Lamport, Pratt risponde precisando la propria posizione: non si deve pensare che uno di loro sia a favore dell'interleaving e l'altro sia contro di esso, perché farlo sminuirebbe molto la sua opinione. Per spiegarsi meglio, quindi, ricorda un articolo che aveva scritto insieme a Ron Rivest [34], e che ai tempi era stato criticato da Lamport. L'articolo riguardava un dimostrazione di correttezza di una soluzione per il problema della mutua esclusione per due processi non affidabili, ovvero che possono fallire in qualunque momento e senza preavviso. Il succo della dimostrazione è che molti dei passaggi del codice scritto dai due rientrava in una di 6 classi, che permettevano una semplice analisi per casi, ognuno dei quali era risolvibile in modo relativamente semplice.

Lamport criticò il loro lavoro, dicendo che una prova come quella, basata sulla classificazione di interlacciamenti, era inappropriata. Inoltre, mostrò

una prova di correttezza della loro procedura basata su una teoria che aveva sviluppato egli stesso. Pratt e Rivest, però, ritenevano che la loro procedura fosse certamente solo una di altre a venire, e che lo sforzo di inventare una teoria dopo aver già scritto il programma fosse sprecato. Inoltre, la loro strategia per scoprire nuovi algoritmi del genere dipendeva proprio dalla natura automatica dell'analisi interleaving.

In seguito, Mike Fischer e Gary Peterson pubblicarono un'estensione del lavoro di Pratt e Rivest, per la quale Peterson creò una teoria "postuma" al fatto, come aveva già fatto Lamport con il lavoro originale. Fischer, però, disse a Pratt che, poiché avevano già verificato la correttezza del loro programma semplicemente eseguendo la procedura attraverso tutti i possibili interlacciamenti, la prova più convenzionale di Peterson, che doveva essere verificata manualmente, non aggiungeva nulla e anzi era più probabile contenesse delle lacune. Pratt ammette che tali teorie "a posteriori" abbiano una certa attrazione estetica, e potrebbero anche essere utili; con questo discorso vuole solo dimostrare di non essere uno zelota oppositore dell'analisi interleaving, e che non ha problemi a usarla quando è applicabile.

Conclude, questa volta, anch'egli con una punta di ironia: si augura che Lamport utilizzi una logica differente per provare la correttezza dei suoi algoritmi rispetto a quella che usa per provare che coloro che nei passati 25 anni sono passati dallo scrivere programmi concorrenti al ragionare in modo astratto su di essi siano diventati perciò dei ciarlatani.

9 Lamport e le teorie del tutto

Il 15 Novembre, Lamport invia finalmente una mail in cui affronta i quesiti posti da Pratt. La sua risposta a tutte e tre le domande è una sola: "Non lo so e non mi interessa. Queste questioni non si presentano mai, nel mio lavoro". Per discutere il perché della differenza di opinioni tra lui e Pratt, uno scienziato che, nonostante le prese in giro delle lettere precedenti, rispetta e considera intelligente, Lamport inizia una digressione sulla natura della scienza in generale.

Qualunque scienza si occupa del mondo reale, infatti ogni teoria scientifica consiste non solo di un formalismo matematico, ma anche di un modo di relazionare quel formalismo al mondo reale. Nel mondo dell'informatica, ad esempio, il mondo reale di solito consiste di programmi eseguiti da computer, mentre le teorie matematiche sono svariate, come le macchine di Turnig, la

logica temporale e il CCS. Inoltre, qualsiasi teoria scientifica che sia utile ha un dominio di applicazione limitato: una teoria del tutto è generalmente inutile. Infatti i lavori citati da Lamport non sono applicabili a qualunque contesto, il loro dominio è la specifica e verifica delle proprietà funzionali di sistemi concorrenti.

Gli informatici tendono ad essere vaghi, quando si tratta del dominio di applicabilità delle proprie teorie, e spesso pensano che siano utilizzabili in qualunque contesto. Un esempio di questo fenomeno è il CCS: molte persone pensano che le sue leggi algebriche siano buone per la verifica di algoritmi, ma in realtà, a detta non solo di Lamport ma anche di Robin Milner, il creatore stesso di CCS, non sono affatto adatte allo scopo. Non bisogna però pensare che una differenza nel dominio di due teorie porti una ad essere migliore o peggiore dell'altra.

Lamport identifica due ragioni per cui un concetto che è importante per una teoria A può essere invece assente da una teoria B:

1. Il concetto è irrilevante per il dominio di applicabilità di B

2. Il concetto appartiene al formalismo matematico di A e, sebbene le due teorie riguardino due domini di applicazione simili, il metodo con cui B traduce la realtà in un formalismo matematico rende il concetto irrilevante o senza significato.

Secondo Lamport, il secondo punto è quello più insidioso, perché gli studiosi di una determinata teoria tendono ad essere così abituati ad utilizzarla che confondono il suo formalismo matematico con la realtà stessa.

Tornando alle domande di Pratt, Lamport fa rientrare il quesito numero tre nella prima categoria, mentre il numero due nella seconda. La prima domanda, invece, quella riguardante $a||b$ e $ab + ba$, viene affrontata:

Una competizione è pericolosa quando fa comportare un circuito in maniera scorretta, ma, nel contesto della verifica di un circuito, è necessario solo verificare che esso si comporti in modo corretto, non incorretto; quindi non serve mai provare l'esistenza di una competizione, anche perché la specifica di un circuito descrive il suo comportamento esterno, mentre una competizione è qualcosa che accade al suo interno. Se invece esiste una competizione potenziale, che però non accade mai, per esempio grazie alle pre-condizioni del circuito, allora la prova conterrà un lemma la cui interpretazione fisica sarà l'assenza della situazione di competizione.

Tuttavia, il concetto di competizione non è irrilevante, in quanto una competizione sugli input potrebbe portare una componente del circuito a produrre un output non valido, come ad esempio 1/2 invece che 0 o 1. In questo caso, un modello matematico che permette solo gli output 0 e 1 sarebbe inadeguato, perché il dominio di applicazione della teoria non includerebbe il circuito reale. Quindi, il concetto di competizione è rilevante per modellare il circuito reale nel formalismo matematico, ma non appare nel formalismo stesso.
Lamport, nella conclusione della mail, torna a parlare dell'approccio che gli informatici hanno relativamente alla scienza. Egli sostiene infatti che molti informatici non siano degli scienziati, bensì dei matematici, in quanto lavorano nel mondo del formalismo teorico senza però considerare le sue applicazioni al mondo reale. Precisa infine che entrambe le figure sono importanti, ma i matematici devono capire di non essere degli scienziati, e non criticare questi ultimi quando trattano il mondo pragmatico e non quello puramente teorico.

10 Le risposte di Pratt e colleghi

La risposta di Pratt arriva tre giorni dopo, il 18 Novembre. Egli innanzitutto si congratula con lui per essersi difeso in modo ammirabile, tuttavia non capisce il motivo che lo ha spinto a rimproverare così duramente i matematici che criticano gli scienziati: Pratt considera le tecniche di Lamport efficaci, per quello che fanno, quindi continua a ritenere la sua crociata contro qualunque critica nei confronti dell'interleaving decisamente eccessiva. In ogni caso, si dice d'accordo con la maggior parte dei punti di Lamport, con alcune eccezioni:

- Quando Lamport dice "non lo so e non mi interessa. Queste questioni non si sollevano mai, nel mio lavoro", Pratt suggerisce che, sebbene la visione di Lamport riguardo la concorrenza sia adeguata per i suoi scopi, essa potrebbe risultare una camicia di forza. Ci sono aspetti della concorrenza che lui non trova degni di essere studiati, ma altre persone sì. Forse le implicazioni di quegli aspetti non si insinueranno mai nel mondo di Lamport, ma non si può esserne sicuri.

- L'idea di Lamport di usare un terzo output, 1/2 , oltre a 0 e 1, è per Pratt eccellente. E' infatti già stata esaminata in passato da altri, per esempio nella nozione di ST-bisimulazione di van Glabbeek e

Vaandrager [9]. L'idea di base della ST-bisimulazione è infatti quella di considerare non solo le azioni che sono già state eseguite, come nella bisimulazione regolare, ma anche quelle che sono in esecuzione, quindi iniziate ma non ancora terminate. Anche Pratt stesso, nel suo articolo per la conferenza POPL-91[29], utilizza tre stati diversi, che lui nomina $\{0, T, 1\}$, e che si riferiscono rispettivamente a "prima", "in transizione" e "dopo". In questo contesto, una competizione è caratterizzata dalla possibilità di avere 2 processi che sono entrambi nello stato T. La funzione della mutua esclusione è quella di escludere tale combinazione. Questa è la distinzione essenziale tra $a||b$ e $ab+ba$: entrambi permettono 8 delle $9 = 3^2$ combinazioni possibili in $(0, T, 1) \times (0, T, 1)$, ma solo la prima permette la nona combinazione (T, T).

- Se per Lamport i matematici non sono scienziati, Pratt ha invece un'opinione del tutto diversa. Il modo in cui la comunità scientifica gestisce i dati provenienti dal mondo reale è quello di inventare teorie attraverso cui quegli stessi dati vengono poi filtrati per produrre delle previsioni il più possibile accurate, e poi le teorie che risultano migliori vengono scelte attraverso una sorta di selezione naturale. Ebbene, le teorie così selezionate tendono ad essere molto simili alle teorie più di successo della matematica pura.

- Riguardo all'inutilità di una teoria del tutto, il fatto che la teoria della concorrenza abbia iniziato a somigliare pericolosamente a qualcosa di simile è già stato notato da vari studiosi. Questo è il risultato dell'astrazione "dai cavi, dal silicio e dai programmi" in favore di teorie che potrebbero essere applicate a domini più generali. Tuttavia, la teoria della concorrenza è una teoria del tutto allo stesso modo in cui lo sono la teoria dei numeri e quella dei gruppi. Pratt spiega poi ulteriormente la propria opinione con una frase molto semplice ma efficace:

  > Proprio come la teoria dei numeri è più della teoria del contare pecore e fagioli [...], così la teoria della concorrenza è più della teoria di cosa fanno ammassi di cavi e silicio concorrenti.

Nella parte finale della mail, Pratt tenta di prevedere il futuro della teoria della concorrenza: egli individua due strade possibili, una che chiama "conservativa" e una "liberale". Quella conservativa vuole tenere in mente i cavi

e il silicio come dominio di applicazione ultimo della ricerca, mentre quella liberale punta a sostituire il termine "computer science" con "information science", e cerca invece una teoria dell'elaborazione delle informazioni che possa essere applicata a una più ampia varietà di campi. Pratt si schiera fortemente a favore della strada liberale, e spiega di essere interessato più alla teoria che alla pratica perché, semplicemente, non è "mai stato bravo a scrivere un buon programma senza prima capire appieno la teoria su cui esso si basa".

Bill Rounds non si schiera né con Pratt né con Lamport: entrambi hanno ragione. Un modello matematico è sempre e solo quello; rappresenta un'astrazione di ciò che veramente percepiamo. I teoremi del modello fanno delle previsioni, che noi reinterpretiamo nel mondo reale, o almeno nella parte del mondo che ci interessa. I modelli migliori sono quelli che semplificano e vincolano la realtà abbastanza da poter fare delle previsioni molto accurate. A volte però il modello, in un particolare dominio, potrebbe non gestire dei fenomeni osservati, ed essere quindi contraddetto. Se si vuole prevedere anche questi nuovi fenomeni, bisogna ridefinire il modello matematico. Questo processo, nonostante sia doloroso per chi crede nel vecchio modello, è al centro del progresso scientifico.

Se prima aveva parlato di scienza, ora Rounds tratta invece l'ingegneria: nel mondo dei computer, è possibile creare sistemi del mondo reale che sono conformi alle nostre percezioni matematiche. Quindi, le macchine sono state progettare per specchiare la nostra concezione di computazione digitale, i linguaggi di programmazione permettono di rappresentare algoritmi matematici e così via. La teoria della concorrenza sembra essere a metà tra scienza e ingegneria. Possiamo usarla per "spiegare" le competizioni, o usarla per aiutarci a progettare dei programmi. I modelli matematici possono essere trasferiti in altri domini di applicabilità; allo stesso modo i concetti della concorrenza possono essere reinterpretati per tipi di dato diversi.

Il punto centrale del discorso di Rounds è semplicemente che bisogna sempre tenere una mente aperta, specialmente quando si tratta di modelli matematici.

A concludere il discorso aperto da Luckham ormai un mese prima è Haim Gaifman, il quale fa alcune osservazioni che riguardano, in generale, la diatriba tra Pratt e Lamport e poi, in particolare, le idee proposte da quest'ultimo.

Gaifman cerca di individuare il motivo che ha portato le opinioni dei due studiosi a divergere tanto: quando A, ovvero Lamport, sostiene di aver fatto qualcosa che B, Pratt, ha dimostrato essere impossibile, allora i casi possibili sono tre:

1. C'è un errore nel lavoro di A
2. C'è un errore nella dimostrazione di B
3. A e B stanno parlando di cose differenti

I primi due punti si possono risolvere facilmente trovando l'errore, mentre il terzo è quello in cui, dice Gaifman, cadono spesso i filosofi. Quando poi si chiarificano le cose, sovente si arriva alla conclusione che il vero problema era il modo corretto di definire certe nozioni, e a questo punto è necessario decidere qual è il modo più intuitivo ed efficiente. Pratt e Lamport sono caduti proprio in questo terzo caso, dato che "gli informatici condividono spesso il destino dei filosofi". Ciò che all'inizio si pensava fosse una contraddizione è in realtà una questione che riguarda i meriti relativi dei modelli delle tracce contro quelli degli ordini parziali. Lamport può certamente affermare che i suoi metodi sono più semplici ed efficienti per analizzare e provare la correttezza di algoritmi distribuiti, infatti i suoi molti lavori a riguardo ne sono la dimostrazione. "Sfortunatamente, ha la cattiva abitudine dei filosofi di iniziare con una presentazione imprecisa del problema".

Un'altra critica mossa nei confronti di Lamport è l'avere un difetto tipico della filosofia, ovvero la tendenza ad ancorare le proprie visioni a qualche grande principio fondamentale: in questo caso, vengono citate massime riguardo a cosa è o non è "buona scienza", il che, per Gaifman, è decisamente eccessivo e distoglie la discussione dal problema originale. In particolare, la frase di Lamport "Qualunque teoria scientifica utile ha un dominio di applicazione limitato. Una teoria del tutto è generalmente inutile" è, in un certo senso, una buona regola generale, ma ha delle evidenti eccezioni: un alto livello di descrizione che racchiude un'ampia gamma di fenomeni può essere più efficiente di una visione ristretta che prende in considerazione solo pochi casi specifici. Ogni matematico conosce casi in cui generalizzare un teorema porta a una dimostrazione più chiara e semplice. C'è anche da dire che una teoria del tutto è l'elusivo traguardo che ha spesso motivato grandi imprese scientifiche.

> Tutto questo non ha nessuna ripercussione diretta sul decidere se un modello interleaving o un modello di ordine parziale sia più adatto a ragionare su processi concorrenti. Ma cercando di includere principi filosofici generali, Lamport sembra essersi affidato a una prospettiva sulla scienza alquanto stretta, più simile alla visione di un ingegnere che altro.

11 Conclusioni

La concorrenza basata sull'interleaving è evidentemente un modello estremamente efficace. La maggior parte delle algebre di processi si basano su di essa, ed esistono moltissimi esperti, tra cui Leslie Lamport, che hanno raggiunto risultati importanti sfruttandola.

È difficile dire se sia una scelta valida quella di passare da un sistema simile, con strumenti ormai ben testati e largamente utilizzati, e che è stato dimostrato essere decisamente efficace nel suo campo, a un modello come quello della vera concorrenza, che è sì capace di trattare casi che l'interleaving non è in grado di gestire, e che potenzialmente è applicabile a campi più generali rispetto a quello "dei cavi e del silicio", ma che per essere utilizzato necessita l'abbandono dei suddetti strumenti a cui gran parte degli studiosi del campo della concorrenza sono ormai abituati.

Tuttavia, le due strade possibili per il futuro individuate da Pratt, quella conservativa e quella liberale, non sono necessariamente alternative: teoria e pratica sono entrambe importanti, così come lo sono la matematica e l'ingegneria. Inseguire una teoria più generica, potenzialmente utilizzabile in ogni campo che tratta lo scambio di informazioni, è certamente un percorso importante, e questo non invalida l'importanza dei modelli più specifici.

Nonostante i frequenti e accesi disaccordi tra le parti coinvolte, il dibattito tra Pratt e Lamport ha certamente avuto dei risvolti positivi, principalmente quello di aver fatto emergere un'idea per il trattamento della vera concorrenza con automi di dimensione superiore, che permettono il riutilizzo di tecniche algebriche potenti, come ad esempio in [7].

Bibliografia

[1] David B. Benson. «Concurrency and interleaving are equally fundamental». In: *Bull. EATCS* 33 (1987), p. 54.

[2] J.A. Bergstra e J.W. Klop. «Process algebra for synchronous communication». In: *Information and Control* 60.1 (1984), pp. 109–137. ISSN: 0019-9958. DOI: https://doi.org/10.1016/S0019-9958(84)80025-X. URL: https://www.sciencedirect.com/science/article/pii/S001999588480025X.

[3] Eike Best e César Fernández C. *Nonsequential processes : a Petri net view*. eng. EATCS monographs on theoretical computer science ; v.13. Berlin: Springer-Verlag, 1988. ISBN: 0387190309.

[4] J Brock e W Ackerman. *An anomaly in the specifications of nondeterministic packet systems*. Rapp. tecn. Technical Report Computation Structures Group Note CSG-33, MIT Lab. for ..., 1977.

[5] Luca Castellano, Giorgio De Michelis e Lucia Pomello. «Concurrency vs interleaving: An instructive example». In: *Bulletin of the European Association for Theoretical Computer Science (EATCS)* 31 (1987).

[6] Rocco De Nicola. «Process Algebras». In: *Encyclopedia of Parallel Computing*. A cura di David Padua. Boston, MA: Springer US, 2011, pp. 1624–1636. ISBN: 978-0-387-09766-4. DOI: 10.1007/978-0-387-09766-4_450. URL: https://doi.org/10.1007/978-0-387-09766-4_450.

[7] Lisbeth Fajstrup, Martin Raußen e Eric Goubault. «Algebraic topology and concurrency». In: *Theoretical Computer Science* 357.1 (2006). Clifford Lectures and the Mathematical Foundations of Programming Semantics, pp. 241–278. ISSN: 0304-3975. DOI: https://doi.org/10.1016/j.tcs.2006.03.022. URL: https://www.sciencedirect.com/science/article/pii/S030439750600274X.

[8] R. J. van Glabbeek. «The linear time - branching time spectrum». In: *CONCUR '90 Theories of Concurrency: Unification and Extension*. A cura di J. C. M. Baeten e J. W. Klop. Berlin, Heidelberg: Springer Berlin Heidelberg, 1990, pp. 278–297. ISBN: 978-3-540-46395-5.

[9] R. J. van Glabbeek e Frits Vaandrager. «Petri net models for algebraic theories of concurrency». In: *PARLE Parallel Architectures and Languages Europe*. A cura di J. W. de Bakker, A. J. Nijman e P. C. Treleaven. Berlin, Heidelberg: Springer Berlin Heidelberg, 1987, pp. 224–242. ISBN: 978-3-540-47181-3.

[10] Jan Grabowski. «On partial languages». In: *Fundamenta Informaticae* 4 (2 1981), pp. 427–498. DOI: 10.3233/fi-1981-4210.

[11] Irene Greif. *Semantics of Communicating Parallel Processes*. Rapp. tecn. 1975.

[12] Matthew Hennessy e Robin Milner. «On observing nondeterminism and concurrency». In: *Automata, Languages and Programming*. A cura di Jaco de Bakker e Jan van Leeuwen. Berlin, Heidelberg: Springer Berlin Heidelberg, 1980, pp. 299–309. ISBN: 978-3-540-39346-7.

[13] Carl E. Hewitt e Henry G. Baker. «Laws for Communicating Parallel Processes». In: *IFIP Congress*. 1977. URL: https://api.semanticscholar.org/CorpusID:9076304.

[14] C. A. R. Hoare. «Communicating Sequential Processes». In: *Commun. ACM* 21.8 (ago. 1978), pp. 666–677. ISSN: 0001-0782. DOI: 10.1145/359576.359585. URL: https://doi.org/10.1145/359576.359585.

[15] A.W. Holt. *Information Systems Theory Project*. Information Systems Theory Project. Rome Air Development Center, Air Force Systems Command, 1968. URL: https://books.google.it/books?id=yftBAAAAIAAJ.

[16] Kai Hwang. *Advanced Computer Architecture: Parallelism, Scalability, Programmability*. 1st. McGraw-Hill Higher Education, 1992. ISBN: 0070316228.

[17] Robert M. Keller. «Formal Verification of Parallel Programs». In: *Commun. ACM* 19.7 (lug. 1976), pp. 371–384. ISSN: 0001-0782. DOI: 10.1145/360248.360251. URL: https://doi.org/10.1145/360248.360251.

[18] Robert M. Keller. «Formal Verification of Parallel Programs». In: *Commun. ACM* 19.7 (lug. 1976), pp. 371–384. ISSN: 0001-0782. DOI: 10.1145/360248.360251. URL: https://doi.org/10.1145/360248.360251.

[19] Leslie Lamport. «An assertional correctness proof of a distributed algorithm». In: *Science of Computer Programming* 2.3 (1982), pp. 175–206. ISSN: 0167-6423. DOI: https://doi.org/10.1016/0167-6423(83)90014-X. URL: https://www.sciencedirect.com/science/article/pii/016764238390014X.

[20] Leslie Lamport. «Specifying Concurrent Program Modules». In: *ACM Trans. Program. Lang. Syst.* 5.2 (apr. 1983), pp. 190–222. ISSN: 0164-0925. DOI: 10.1145/69624.357207. URL: https://doi.org/10.1145/69624.357207.

[21] Leslie Lamport. «Time, Clocks, and the Ordering of Events in a Distributed System». In: *Commun. ACM* 21.7 (lug. 1978), pp. 558–565. ISSN: 0001-0782. DOI: 10.1145/359545.359563. URL: https://doi.org/10.1145/359545.359563.

[22] Leslie Lamport. «Win and Sin: Predicate Transformers for Concurrency». In: *ACM Trans. Program. Lang. Syst.* 12.3 (lug. 1990), pp. 396–428. ISSN: 0164-0925. DOI: 10.1145/78969.78970. URL: https://doi.org/10.1145/78969.78970.

[23] Robin Milner. *A Calculus of Communicating Systems*. Berlin, Heidelberg: Springer-Verlag, 1982. ISBN: 0387102353.

[24] Rocco De Nicola. «A gentle introduction to Process Algebras ?» In: 2013. URL: https://api.semanticscholar.org/CorpusID:18837826.

[25] Mogens Nielsen, Gordon D. Plotkin e Glynn Winskel. «Petri nets, event structures and domains, part I». In: *Theoretical Computer Science* 13.1 (1981). Special Issue Semantics of Concurrent Computation, pp. 85–108. ISSN: 0304-3975. DOI: https://doi.org/10.1016/0304-3975(81)90112-2. URL: https://www.sciencedirect.com/science/article/pii/0304397581901122.

[26] C. A. Petri. «Concepts of Net Theory». In: *Mathematical Foundations of Computer Science: Proceedings of Symposium and Summer School, Strbské Pleso, High Tatras, Czechoslovakia, September 3-8, 1973*. Mathematical Institute of the Slovak Academy of Sciences, 1973, pp. 137–146.

[27] Gordon D. Plotkin e Vaughan R. Pratt. «Teams can see pomsets». In: *Partial Order Methods in Verification*. 1990.

[28] Vaughan R. Pratt. «Higher dimensional automata revisited». In: *Mathematical Structures in Computer Science* 10.4 (2000), pp. 525–548. DOI: 10.1017/S0960129500003169.

[29] Vaughan R. Pratt. «Modeling Concurrency with Geometry». In: *Proceedings of the 18th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*. POPL '91. Orlando, Florida, USA: Association for Computing Machinery, 1991, pp. 311–322. ISBN: 0897914198. DOI: 10.1145/99583.99625. URL: https://doi.org/10.1145/99583.99625.

[30] Vaughan R. Pratt. «Modeling concurrency with partial orders». In: *International Journal of Parallel Programming* 15 (1986), pp. 33–71. URL: https://api.semanticscholar.org/CorpusID:12178098.

[31] Vaughan R. Pratt. «On the Composition of Processes». In: *Proceedings of the 9th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*. POPL '82. Albuquerque, New Mexico: Association for Computing Machinery, 1982, pp. 213–223. ISBN: 0897910656. DOI: 10.1145/582153.582177. URL: https://doi.org/10.1145/582153.582177.

[32] Vaughan R. Pratt. «Two-way Channel with Disconnect». In: *The Analysis of Concurrent Systems, Cambridge, UK, September 12-16, 1983, Proceedings*. A cura di B. Tim Denvir et al. Vol. 207. Lecture Notes in Computer Science. Springer, 1983, pp. 110–111. DOI: 10.1007/3-540-16047-7_39. URL: https://doi.org/10.1007/3-540-16047-7%5C_39.

[33] W. Reisig. «Concurrency is More Fundamental than Interleaving». In: *Bull. EATCS* (1988).

[34] Vaughan R. Pratt Ronald L. Rivest. «The Mutual Exclusion Problem for Unreliable Processes: Preliminary Report». In: *17th Annual Symposium on Foundations of Computer Science, Houston, Texas, USA, 25-27 October 1976*. IEEE Computer Society, 1976, pp. 1–8. DOI: 10.1109/SFCS.1976.33. URL: https://doi.org/10.1109/SFCS.1976.33.

[35] Edward Szpilrajn. «Sur l'extension de l'ordre partiel». fre. In: *Fundamenta Mathematicae* 16.1 (1930), pp. 386–389. URL: http://eudml.org/doc/212499.