Costruzioni per la semantica operazionale della concorrenza

Roberto Bruni and Ugo Montanari
Dipartimento di Informatica, Università diPisa

25 giugno 2010

1 Introduzione

Molte delle nostre attività quotidiane dipendono dall'uso dei moderni sistemi ICT. Si tratti di attività legate all'intrattenimento, alla comunicazione, alla salute, all'amministrazione o alla ricerca, il corretto funzionamento dei sistemi ICT assume un ruolo principale con conseguenze dirette sulle nostre vite. Trattandosi di sistemi complessi, per garantirne il buon comportamento è necessario disporre di modelli adeguati che possano essere usati per progettare, simulare, verificare e ingegnerizzare questi sistemi prima di realizzarli e diffonderli. Si badi però che, citando Edsger W. Dijkstra[1], *l'informatica non riguarda i computer più di quanto l'astronomia riguardi i telescopi*, nel senso che mentre il ruolo di ogni dispositivo hardware o software è sostanzialmente quello di realizzare un certo meccanismo di calcolo, per potenziare le nostre capacità di risolvere problemi grazie alla maggiore dimensione di dati trattabili e alla più elevata velocità nell'elaborarli, la sfida principale per lo scienziato informatico resta quella di analizzare ogni specifico problema e ideare il meccanismo di calcolo più adatto per studiarlo, affrontarlo e risolverlo.

Una caratteristica intrinseca dei moderni sistemi ICT, dai dispositivi multiprocessore alle reti peer-to-peer e alle piattaforme orientate ai servizi, è quella di essere sistemi *concorrenti*, cioè descrivibili in termini di entità computazionali in grado di evolversi in modo autonomo, di sincronizzarsi, di competere per l'acquisizione di risorse necessarie a completare un certo compito e di scambiarsi informazioni. Il funzionamento globale dei sistemi concorrenti dipende quindi da fattori difficilmente controllabili da chi progetta le singole componenti del sistema, quali la velocità e l'ordine di esecuzione delle azioni di certi processi o la politica di allocazione di dati e processi in sistemi distribuiti; inoltre questi fattori sono influenzati dal numero di processi in esecuzione e dalle comunicazioni che tentano di effettuare, di modo che certe applicazioni possono funzionare

[1] Edsger W. Dijkstra (1930–2002) è uno dei più celebri informatici, insignito nel 1972 del ACM Turing Award, il corrispondente del premio Nobel per l'informatica. Per maggiori informazioni si veda l'interessante sito E. W. Dijkstra Archive `http://userweb.cs.utexas.edu/users/EWD/` che raccoglie i suoi scritti.

in maniera anomala in modo intermittente, sotto particolari circostanze difficilmente replicabili, rendendo arduo individuare la natura del problema. Problemi tipici dei sistemi concorrenti sono: la dipendenza dell'esito di una computazione da un particolare ordine di esecuzione di certe azioni in processi differenti; il verificarsi di situazioni di stallo con processi che si bloccano mutuamente in attesa di determinati eventi, dove con il termine *evento* indichiamo l'esecuzione di una certa operazione. La *teoria della concorrenza* si occupa dello studio delle relazioni che intercorrono tra i diversi eventi che possono verificarsi nei sistemi concorrenti. La concorrenza è caratterizzata da un comportamento altamente non-deterministico dovuto all'elevato numero di computazioni possibili: come esempio di esplosione combinatoriale, eseguendo in modo concorrente 100 processi ciascuno con 10 stati possibili il sistema risultante da studiare potrebbe avere fino a 10^{100} stati possibili.

Lo scopo di questo scritto è quello di descrivere un approccio metodologico basato sulla *Teoria delle Categorie* [11] e mirato alla ricerca delle leggi fondamentali che regolano le computazioni dei modelli concorrenti: per analogia, si pensi ai modelli, certamente più familiari, sviluppati nelle Scienze Fisiche per lo studio del moto o nelle Scienze Naturali per studiare l'evolversi di una popolazione in un ambiente chiuso.

Riteniamo opportuno rimarcare che in questo scritto siamo interessati alla cosiddetta *semantica operazionale* di sistemi concorrenti, non alla cosiddetta *semantica astratta*. La prima ha lo scopo di studiare *modelli computazionali* che definiscano gli stati possibili del sistema, gli eventi che possono verificarsi, le transizioni di stato provocate dagli eventi e le relazioni tra eventi. La seconda riguarda generalmente proprietà estensionali dei sistemi, che vengono interpretati come oggetti matematici di un opportuno dominio. Sostanzialmente possiamo dire che la semantica operazionale riguarda il "come" un certo sistema evolve, mentre la semantica astratta riguarda "cosa" viene prodotto durante l'evoluzione. Per esempio, nella teoria dei linguaggi regolari, gli automi a stati finiti definiscono un modello computazionale per il riconoscimento di sequenze di simboli (dette *stringhe* in gergo informatico), mentre l'insieme delle stringhe riconosciute definisce il corrispondente modello astratto.

Una seconda annotazione di rilievo è che nel definire la semantica operazionale è essenziale adottare un livello di astrazione tale da concentrare l'analisi sulle dipendenze tra eventi e mantenere compatto il meccanismo di calcolo sottostante, senza introdurre aspetti semantici non necessari, che rischierebbero di compromettere la validità e la riusabilità dei modelli e del software studiati. In questo senso, preferiamo non fornire misure di tempo assoluto (discreto o continuo) o di durata degli eventi nel descrivere una computazione, perché questo la renderebbe troppo legata ad un particolare politica di esecuzione dei processi o dal carico del/i processore/i e renderebbe meno evidenti le dipendenze logiche tra gli eventi (a meno di esaminare tutte le computazioni possibili).

Mentre una computazione (deterministica) sequenziale è identificabile con una sequenza (totalmente ordinata) di eventi, nel caso di computazioni (deterministiche) concorrenti l'ordine tra gli eventi è *parziale*, perché due eventi concorrenti non sono vincolati da dipendenze causali e quindi sequenze di eventi

che differiscono solo per l'ordine relativo col quale vi compaiono gli stessi eventi concorrenti descrivono in sostanza la medesima *computazione concorrente*.

Per descrivere lo spazio di tutte le possibili computazioni, nel caso sequenziale dobbiamo prevedere che dopo un certo evento possano verificarsi più eventi futuri, ciascuno in mutua esclusione con gli altri (e con i loro discendenti). Nel caso concorrente, si passa da un ordine parziale di eventi a una cosiddetta *struttura di eventi* o, più precisamente, a un cosiddetto *dominio algebrico primo*.

In un certo senso, un modello computazionale fornisce una rappresentazione compatta dello spazio di tutte le possibili computazioni e modelli computazionali finiti possono generare spazi di computazione infiniti. Per esempio il linguaggio regolare riconosciuto da un automa a stati finiti può comprendere un numero infinito di sequenze. Nel caso di modelli concorrenti, è quindi importante che il modello computazionale permetta di identificare le diverse entità computazionali che compongono il sistema e agiscono in modo concorrente, perché questa informazione è essenziale nel determinare la relazione di concorrenza tra eventi appartenenti allo spazio delle computazioni.

Le caratteristiche principali dell'approccio che presentiamo sono quelle di: 1) assumere che la struttura (algebrica) degli stati determini la concorrenza del sistema; 2) generare in modo "automatico" lo spazio delle computazioni concorrenti a partire dal modello computazionale; 3) garantire che il concetto di simulazione tra modelli sia estendibile ai rispettivi spazi di computazione; 4) associare ad ogni possibile spazio delle computazioni una particolare istanza (anche infinita) del modello computazionale, garantita essere "migliore" di tutte le altre possibili istanze. Il punto 1) garantisce una buona generalità dell'approccio, nel senso che la metodologia e il concetto di concorrenza divengono parametrici rispetto alla struttura degli stati. Il punto 2) permette di caratterizzare le leggi della concorrenza in base alle leggi di composizione degli stati. Il punto 3) stabilisce la coerenza della costruzione dello spazio delle computazioni rispetto al modo di relazionare i modelli di computazione. Il punto 4) dimostra la bontà della costruzione, nel senso che ogni altro modo di definire la computazione del modello nello spazio di computazioni selezionato non potrebbe comunque offrire una descrizione più precisa. L'approccio algebrico infine fornisce un approccio composizionale per lo studio di sistemi complessi, nel senso che permette di sfruttare la struttura algebrica per applicare tecniche di analisi induttive o basata sul principio "divide et impera". Come vedremo, tutte queste caratteristiche sono ben esprimibili sfruttando concetti elementari della Teoria delle Categorie.

Per cominciare, analizziamo il caso banale di sistemi sequenziali con stati non strutturati. Il modello di computazione corrispondente è quello dei cosiddetti *sistemi di transizione*, che sostanzialmente sono dei grafi diretti.

Definizione 1.1 (Sistema di transizione)
Un sistema di transizione S *è una quadrupla* (S, T, src, trg)*, dove* S *è l'insieme degli stati,* T *l'insieme delle transizioni,* $src : T \to S$ *è la funzione sorgente, che assegna ad ogni transizione lo stato di partenza e* $trg : T \to S$ *è la funzione destinazione, che assegna ad ogni transizione lo stato di arrivo.*

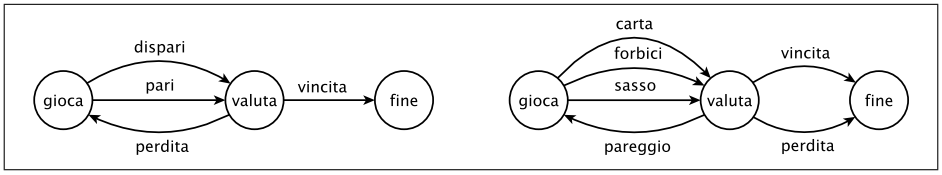


Figura 1: Due semplici sistemi di transizione: PD a sinistra e MC a destra

Con la notazione $t : s \to s'$ indichiamo una transizione $t \in T$ tale che $src(t) = s$ e $trg(t) = s'$. Talvolta il sistema di transizione è accompagnato da un ben preciso stato iniziale $s_0 \in S$, che determina lo stato di partenza di una qualsiasi computazione. Quando utile per disambiguare la notazione, usiamo il pedice S per esplicitare che gli insiemi di stati e transizioni e che le le funzioni sorgente e destinazione cui facciamo riferimento sono quelli di S. Inoltre, quando introduciamo apici come in S' assumiamo tacitamente che la corrispondente quadrupla sia (S', T', src', trg'). Infine, abusiamo la notazione scrivendo $s \in \mathsf{S}$ e $t \in \mathsf{S}$ invece che $s \in S_{\mathsf{S}}$ e $t \in T_{\mathsf{S}}$.

Per esempio, in Figura 1 troviamo due semplici sistemi di transizione con tre stati $\{\text{gioca}, \text{valuta}, \text{fine}\}$. Il sistema a sinistra, che chiamiamo PD da "pari o dispari", ha quattro transizioni $\{\text{dispari}, \text{pari}, \text{perdita}, \text{vincita}\}$. Stati e transizioni sono rappresentati rispettivamente con dei cerchi e degli archi orientati dalla sorgente verso la destinazione. Intuitivamente, le transizioni $\text{dispari}, \text{pari} : \text{gioca} \to \text{valuta}$ rappresentano due scelte alternative del giocatore; la transizione $\text{perdita} : \text{valuta} \to \text{gioca}$ rappresenta l'esito negativo del gioco che porta il giocatore alla possibilità di ritentare; la transizione $\text{vincita} : \text{valuta} \to \text{fine}$ rappresenta l'esito positivo del gioco, col giocatore che si ritiene soddisfatto. Il sistema a destra in Figura 1, che chiamiamo MC da "morra cinese" invece ha sei transizioni e intuitivamente rappresenta un gioco che offre tre scelte al giocatore $(\text{carta}, \text{forbici}, \text{sasso})$, con la possibilità di ripetere in caso di pareggio e terminazione in caso di vincita o perdita.

Computazionalmente, l'idea è che l'insieme S esprime tutte le possibili configurazioni del sistema, mentre le transizioni T circoscrivono i modi in cui il sistema può evolvere, passando da una configurazione a un'altra. Una computazione diviene quindi un cammino che parte da una certa configurazione iniziale e attraversa altre configurazioni rispettando le transizioni ammissibili. Quando da uno stato escono più archi, significa che sono disponibili più modi di evolvere anche se ogni computazione potrà seguire uno solo di essi. Formalmente, una computazione di lunghezza $n \in \mathbb{N}$ da uno stato s a uno stato s' è una sequenza di transizioni $(t_1 t_2 ... t_n)$ tale che per ogni $i \in [1, n]$ si ha $t_i : s_i \to s_{i+1}$ per opportuni stati $s_1, s_2, ..., s_{n+1}$ con $s = s_1$ e $s' = s_{n+1}$. Denotiamo tale computazione scrivendo $(t_1 t_2 ... t_n) : s \to^* s'$, dove il simbolo $*$ evidenzia che la relazione di raggiungibilità $\to^* \subseteq S \times S$ indotta dalle computazioni è la chiusura riflessiva e transitiva della relazione di raggiungibilità con una singola mossa indotta dalle transizioni. Come caso particolare, segnaliamo l'esistenza di computazioni vuote (cioè di lunghezza 0) $\langle\,\rangle_s : s \to s$ da ogni stato in se stesso.

Per esempio, nel sistema PD di Figura 1, le computazioni da gioca in gioca so-

no quella vuota e tutte le sequenze del tipo (dp_1 perdita dp_2 perdita...dp_k perdita) con $dp_i \in \{\text{dispari}, \text{pari}\}$, mentre le computazioni dallo stato gioca allo stato fine sono le sequenze del tipo (dp_1 perdita dp_2 perdita...dp_k perdita dp_{k+1} vincita).

Una prima cosa interessante da notare è che le computazioni di un sistema di transizione formano una categoria.

Definizione 1.2 (Categoria)
Una categoria $\mathbf{C}$ *è una sestupla* $(O, F, \delta_0, \delta_1, id_\_, \_;\_)$*, dove* O *è l'insieme degli oggetti,* F *l'insieme delle frecce,* $\delta_0 : F \to O$ *è la funzione sorgente,* $\delta_1 : F \to O$ *è la funzione destinazione,* $id_\_ : O \to F$ *è la funzione identità e* $\_;\_ : F \times F \to F$ *è una funzione parziale che definisce la composizione tra frecce, tali che tutte le seguenti condizioni sono rispettate:*

- *per ogni* $a \in O$ *vale* $\delta_0(id_a) = a$ *e* $\delta_1(id_a) = a$*;*
- *per ogni* $f, g \in F$ *la composizione* $f; g$ *è definita se e solo se* $\delta_1(f) = \delta_0(g)$ *e inoltre* $\delta_0(f; g) = \delta_0(f)$ *e* $\delta_1(f; g) = \delta_1(g)$*;*
- *per ogni* $f \in F$ *vale* $f; id_{\delta_1(f)} = f = id_{\delta_0(f)}; f$*;*
- *per ogni* $f, g, h \in F$ *tali che* $f; g$ *e* $g; h$ *sono definite vale* $(f; g); h = f; (g; h)$*.*

Scriviamo $f : a \to b$ per $f \in F$ con $\delta_0(f) = a$ e $\delta_1(f) = b$ e quando sia utile usiamo $\mathbf{C}$ come pedice per disambiguare la notazione e scriviamo $\mathbf{C}'$ per intendere $(O', F', \delta'_0, \delta'_1, id'_\_, \_;'\_)$. Sostanzialmente una categoria è una struttura matematica per descrivere spazi di frecce chiusi rispetto a una certa nozione di composizione: ogni freccia ha una sorgente e una destinazione, due frecce si compongono quando la destinazione dell'una coincide con la sorgente dell'altra e la loro composizione è ancora una freccia della categoria. La composizione, quando definita, è associativa. Agli oggetti della categoria sono associate frecce particolari, dette *identità*, che si compongono in maniera neutra con le altre frecce.

Nella teoria delle categorie spesso le proprietà delle frecce si esprimono disegnando opportuni diagrammi e dicendo che questi diagrammi *commutano*: questo equivale a richiedere che, presi due qualsiasi oggetti nel diagramma, ogni composizione di frecce che porta da un oggetto all'altro dia come risultato la stessa freccia. Per esempio, gli assiomi di associatività e identità relativi alla composizione di frecce vengono espressi richiedendo semplicemente che i diagrammi in Figura 2 commutino.

Esempi classici di categorie sono: la categoria **Set** che ha insiemi come oggetti e funzioni (totali) come frecce; la categoria **Rel** che ha insiemi come oggetti e relazioni come frecce; la categoria **Mon** i cui oggetti sono monoidi $(M, \otimes, 1)$ e le cui frecce sono omomorfismi tra monoidi; la categoria **CMon** i cui oggetti sono monoidi commutativi $(C, \oplus, 1)$ e le cui frecce sono omomorfismi tra monoidi; la categoria **Graph** che ha grafi come oggetti e omomorfismi tra grafi come frecce, tutti con le usuali nozioni di identità e di composizione. A volte siamo interessati a considerare categorie formate selezionando solo alcuni

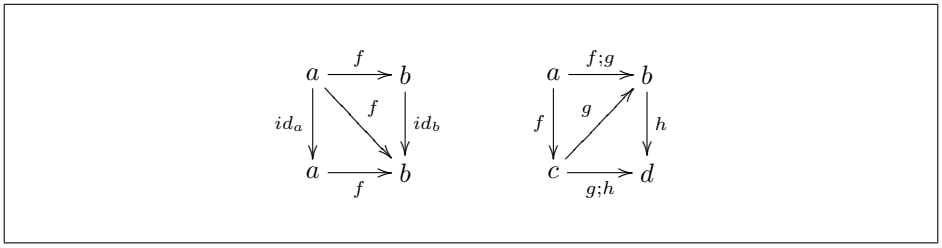


Figura 2: Proprietà della composizione tra frecce in una categoria espressi come diagrammi che commutano

specifici elementi e frecce di un'altra categoria: in questo caso si parla di *sotto-categoria*. Per esempio, **Set** è una sottocategoria di **Rel**, perché ogni funzione tra insiemi definisce una relazione. Si parla di sotto-categoria *piena* (*full*) di una categoria $\mathbf{C}$, quando la sotto-categoria comprende tutte le frecce di $\mathbf{C}$ tra gli oggetti selezionati.

La teoria delle categorie permette di rifrasare e generalizzare nozioni familiari della teoria degli insiemi in termini di proprietà di (diagrammi composti da) frecce, con la caratteristica che le definizioni sono sempre date *a meno di isomorfismo*: se un oggetto soddisfa una certa proprietà allora la soddisfa anche ogni oggetto isomorfo ad esso. In particolare, le definizioni diventano più eleganti e meno specializzate perché non devono fissare esplicitamente gli oggetti di interesse. Ricordiamo che una freccia $f : a \to b$ della categoria $\mathbf{C}$ è un *isomorfismo* se esiste una freccia $f^{-1} : b \to a \in \mathbf{C}$, detta *inversa* di f, tale che $f; f^{-1} = id_a$ e $f^{-1}; f = id_b$. Se esiste, l'inversa di f è unica ed è essa stessa un isomorfismo. Due oggetti sono detti *isomorfi* se esiste un isomorfismo tra essi.

Un'altra caratteristica della teoria delle categorie è quella di rendere immediata la definizione di costruzioni *duali* a quelle note. Infatti, data una qualsiasi categoria $\mathbf{C}$ è sempre possibile definire la categoria $\mathbf{C}^{op}$ invertendo la direzione delle frecce di $\mathbf{C}$: la categoria $\mathbf{C}^{op}$ ha gli stessi oggetti di $\mathbf{C}$ e per ogni freccia $f : a \to b \in \mathbf{C}$ ha una freccia $f^{op} : b \to a$, detta *duale di* f. Le identità di $\mathbf{C}^{op}$ sono le frecce duali delle identità in $\mathbf{C}$ e la composizione è definita da $f^{op}; g^{op} = (g; f)^{op}$.

Si noti che il concetto di morfismo tra grafi ci fornisce un modo naturale per relazionare due sistemi di transizione, infatti un omomorfismo $h : \mathsf{S} \to \mathsf{S}'$ manda stati di S in stati di S', transizioni di S in transizioni di S' rispettando sorgente e destinazione di ogni transizione (nel senso che per ogni $t : s_1 \to s_2 \in \mathsf{S}$ si ha $h(t) : h(s_1) \to h(s_2) \in \mathsf{S}'$). Quindi l'esistenza del morfismo $h : \mathsf{S} \to \mathsf{S}'$ indica che ogni computazione di S può essere riprodotta anche in S'.

Per esempio, relativamente alla Figura 1 un possibile morfismo $h_1 : \mathsf{PD} \to \mathsf{MC}$ è quello che manda ogni stato in quello omonimo e che è definito sulle transizioni come $h_1(\text{dispari}) = \text{carta}$, $h_1(\text{pari}) = \text{sasso}$, $h_1(\text{perdita}) = \text{pareggio}$ e $h_1(\text{vincita}) = \text{vincita}$. Viceversa, un possibile morfismo $h_2 : \mathsf{MC} \to \mathsf{PD}$ è quello che manda ogni stato in quello omonimo e che è definito sulle transizioni come $h_2(\text{carta}) = h_2(\text{forbici}) = h_2(\text{sasso}) = \text{pari}$, $h_2(\text{pareggio}) = \text{perdita}$,

$h_2(\text{vincita}) = h_2(\text{perdita}) = \text{vincita}$. Per quanto detto prima a proposito degli isomorfismi, è evidente che i nomi degli stati e delle transizioni sono totalmente irrilevanti nell'approccio categoriale, anche se utili per la comprensione dei modelli e delle loro computazioni.

Nel definire la categoria delle computazioni, possiamo prendere come oggetti della categoria gli stati del sistema di transizione e come frecce le sequenze di transizioni che permettono di passare da uno stato all'altro. Le identità sono cammini vuoti che partono da un certo stato e la composizione di due computazioni $(t_1 t_2 ... t_n) : s \to^* s'$ e $(t'_1 t'_2 ... t'_m) : s' \to^* s''$ è la computazione $(t_1 t_2 ... t_n t'_1 t'_2 ... t'_m) : s \to^* s''$.

Formalmente, possiamo caratterizzare la corrispondenza tra un sistema di transizione e la sua categoria di computazioni come una *costruzione universale*, a garanzia che la categoria individuata è quella che meglio descrive lo spazio di computazioni. Per fare questo abbiamo bisogno di introdurre alcuni concetti elementari della teoria delle categorie.

La prima nozione è quella di omomorfismo tra categorie, in gergo chiamato *funtore*, che permette di immergere una categoria all'interno di un'altra rispettandone la struttura matematica, ovvero preservando identità e composizione.

DEFINIZIONE 1.3 (FUNTORE)
Date due categorie $\mathbf{C}$ *e* $\mathbf{D}$, *un* funtore $\mathcal{F} : \mathbf{C} \to \mathbf{D}$ *è una coppia di morfismi* $\mathcal{F} = (\mathcal{F}_O : O_{\mathbf{C}} \to O_{\mathbf{D}}, \mathcal{F}_F : F_{\mathbf{C}} \to F_{\mathbf{D}})$ *tale che:*

- *per ogni* $f \in F_{\mathbf{C}}$ *vale* $\mathcal{F}_O(\delta_0(f)) = \delta_0(\mathcal{F}_F(f))$ *e* $\mathcal{F}_O(\delta_1(f)) = \delta_1(\mathcal{F}_F(f))$*;*
- *per ogni* $a \in O_{\mathbf{C}}$ *vale* $\mathcal{F}_F(id_a) = id_{\mathcal{F}_O(a)}$*;*
- *per ogni* $f, g \in F_{\mathbf{C}}$ *tali che* $f ;_{\mathbf{C}} g$ *è definita, vale* $\mathcal{F}_F(f ;_{\mathbf{C}} g) = \mathcal{F}_F(f) ;_{\mathbf{D}} \mathcal{F}_F(g)$.

Per rendere la notazione più snella, scriviamo semplicemente $\mathcal{F}(a)$ e $\mathcal{F}(f)$ invece di $\mathcal{F}_O(a)$ e $\mathcal{F}_F(f)$ senza pericolo di ambiguità. Alcuni semplici esempi di funtori sono: l'inclusione di **Set** in **Rel** che manda ogni insieme su se stesso e ogni funzione $f : A \to B$ sulla relazione $\{(a, f(a)) \mid a \in A\} \subseteq A \times B$; la proiezione $\mathcal{P}_O : \mathbf{Graph} \to \mathbf{Set}$ che manda ogni grafo nell'insieme dei suoi nodi e ogni omomorfismo tra grafi nella sua componente definita sui nodi. La categoria **Cat** è la categoria i cui oggetti sono categorie e le cui frecce sono funtori; infatti è immediato osservare che per ogni categoria esiste un ovvio funtore identità e che i funtori possono essere composti mediante composizione delle sottostanti coppie di morfismi. Dato che le computazioni formano una categoria, il concetto di funtore ci fornisce anche il modo più naturale di relazionare due spazi di computazione. Si noti anche che l'immagine attraverso un funtore di un diagramma che commuta è ancora un diagramma che commuta.

Un primo funtore di particolare interesse è $\mathcal{U}^{\cdot} : \mathbf{Cat} \to \mathbf{Graph}$ definito considerando, per ogni categoria, il grafo formato da tutti gli oggetti e tutte le frecce di quella categoria ma dimenticando la "struttura aggiuntiva", cioè il fatto che alcune frecce sono identità e che le frecce possono essere composte. La definizione è ben data, osservando che ogni funtore tra due categorie sottintende

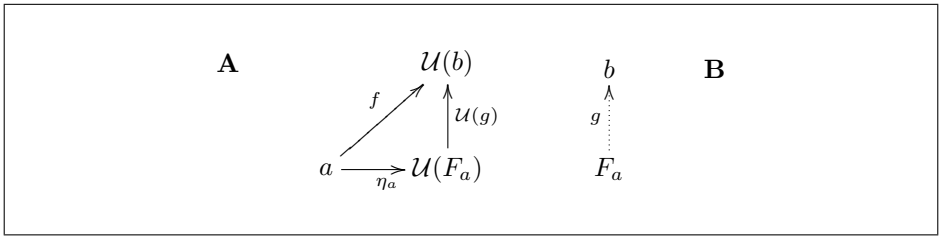


Figura 3: Diagramma per l'oggetto libero F_a

un omomorfismo tra i corrispondenti grafi: in più il funtore preserva anche la struttura aggiuntiva.

La seconda nozione che ci serve è quella di oggetto libero.

Definizione 1.4 (Oggetto libero)
Dato un funtore $\mathcal{U} : \mathbf{B} \to \mathbf{A}$ e un oggetto $a \in \mathbf{A}$ si dice che un certo oggetto $F_a \in \mathbf{B}$ è libero *su $\mathbf{A}$ se esiste una freccia $\eta_a : a \to \mathcal{U}(F_a)$ tale che per ogni altro oggetto $b \in \mathbf{B}$ e freccia $f : a \to \mathcal{U}(b) \in \mathbf{A}$ esiste un'unica freccia $g : F_a \to b \in \mathbf{B}$ tale che $\eta_a;\mathcal{U}(g) = f$ (si veda la Figura 3, dove la punteggiatura della freccia g ne simboleggia l'unicità).*

La proprietà che definisce un oggetto libero è detta *proprietà universale* e la freccia η_a è detta *freccia universale*. Sostanzialmente la proprietà universale ci garantisce che l'oggetto F_a è il migliore (a meno di isomorfismo) dei rappresentanti possibili che possiamo trovare per a all'interno di $\mathbf{B}$ rispetto al funtore $\mathcal{U}$. Per capire meglio, immaginiamo che un morfismo $f : a \to a'$ di una categoria definisca un modo di rappresentare l'oggetto a attraverso a', ovvero di vedere a' come un'approssimazione di a. L'idea è di cercare la migliore approssimazione possibile dell'oggetto a tra gli oggetti che appartengono all'immagine di $\mathcal{U}(\mathbf{B})$. Quindi ogni oggetto $b \in \mathbf{B}$ tale che esiste una freccia $f : a \to \mathcal{U}(b) \in \mathbf{A}$ offre implicitamente un possibile candidato (cioè $\mathcal{U}(b)$). La proprietà universale garantisce che il candidato $\mathcal{U}(F_a)$ (mediante η_a) fornisce l'approssimazione più vicina ad a, perché consente di decomporre in modo univoco ogni altra approssimazione possibile $f : a \to \mathcal{U}(b) \in \mathbf{A}$ di a attraverso $g : F_a \to b$ in $\mathbf{B}$. Si noti che, come avevamo notato prima a proposito delle caratteristiche della teoria delle categorie, un oggetto libero quando esiste non è necessariamente unico, ma se ve ne sono più di uno questi sono tutti isomorfi tra loro.

Nel caso di interesse, la categoria delle computazioni gioca il ruolo di $\mathbf{B}$ e la categoria dei modelli computazionali quello di $\mathbf{A}$. Il funtore $\mathcal{U}$ associa un modello ad ogni computazione e dato un modello a, l'oggetto libero F_a è lo spazio delle computazioni che meglio descrive a tra quelli che potremmo scegliere in $\mathbf{B}$.

La terza nozione è quella di *aggiunzione* tra due categorie. Lo scenario comprende due categorie $\mathbf{A}$ e $\mathbf{B}$ con due funtori $\mathcal{F} : \mathbf{A} \to \mathbf{B}$ e $\mathcal{U} : \mathbf{B} \to \mathbf{A}$ tali che il funtore $\mathcal{F}$ manda ogni oggetto a in un oggetto libero $\mathcal{F}(a)$ rispetto a $\mathcal{U}$.

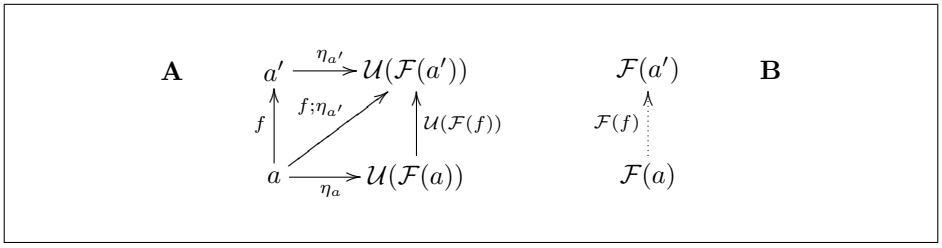


Figura 4: Diagramma per l'aggiunzione $\mathcal{F} \dashv \mathcal{U}$

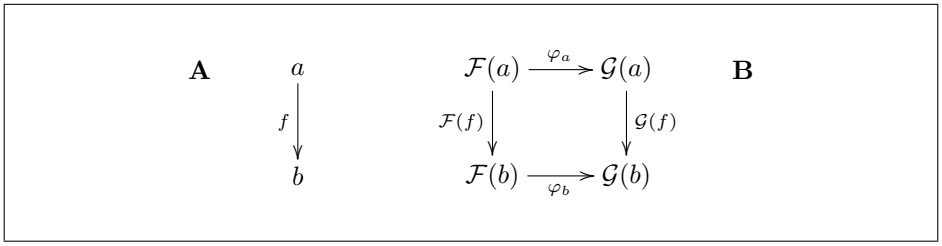


Figura 5: Diagramma che esprime la naturalità di $\varphi : \mathcal{F} \Rightarrow \mathcal{G} : \mathbf{A} \to \mathbf{B}$.

Definizione 1.5 (Aggiunto sinistro)
Sia dato un funtore $\mathcal{U} : \mathbf{B} \to \mathbf{A}$. Se per ogni oggetto $a \in \mathbf{A}$ esiste un oggetto libero $F_a \in \mathbf{B}$ con freccia universale $\eta_a : a \to \mathcal{U}(F_a)$, allora possiamo definire un funtore $\mathcal{F} : \mathbf{A} \to \mathbf{B}$, detto aggiunto sinistro *di $\mathcal{U}$, come segue:*

- *per ogni oggetto $a \in \mathbf{A}$, poniamo $\mathcal{F}(a) = F_a$;*
- *per ogni freccia $f : a \to a' \in \mathbf{A}$, definiamo come $\mathcal{F}(f) : F_a \to F_{a'}$ l'unica freccia g tale che $\eta_a;\mathcal{U}(g) = f;\eta_{a'}$, la cui esistenza e unicità sono garantite dalla proprietà universale di F_a rispetto alla freccia $f;\eta_{a'}$ (si veda la Figura 4).*

Si noti che $\mathcal{F}$ è un funtore perché rispetta identità e composizione grazie alla condizione di unicità della proprietà universale. Il funtore $\mathcal{U} : \mathbf{B} \to \mathbf{A}$ è detto *aggiunto destro*, la coppia $(\mathcal{F}, \mathcal{U})$ è detta *aggiunzione* e si usa la notazione $\mathcal{F} \dashv \mathcal{U}$ per evidenziarlo. La famiglia di frecce $\{\eta_a\}_{a \in \mathbf{A}}$ è detta *unit* dell'aggiunzione e costituisce una cosiddetta *trasformazione naturale*.

Definizione 1.6 (Trasformazione naturale)
Dati due funtori $\mathcal{F}, \mathcal{G} : \mathbf{A} \to \mathbf{B}$, una trasformazione naturale *$\varphi : \mathcal{F} \Rightarrow \mathcal{G}$ è una famiglia $\{\varphi_a : \mathcal{F}(a) \to \mathcal{G}(a)\}_{a \in \mathbf{A}}$ di frecce in $\mathbf{B}$ tali che per ogni $f : a \to b \in \mathbf{A}$ vale $\varphi_a;\mathcal{G}(f) = \mathcal{F}(f);\varphi_b$ (si veda la Figura 5). Una trasformazione naturale φ tale che per ogni $a \in \mathbf{A}$ la freccia φ_a è un isomorfismo è detta* isomorfismo naturale.

È immediato verificare che la unit di un'aggiunzione $\mathcal{F} \dashv \mathcal{U}$ definisce una trasformazione naturale $\eta : 1_{\mathbf{A}} \Rightarrow \mathcal{F};\mathcal{U}$ tra il funtore identità e quello ottenuto componendo i due aggiunti (infatti, per definizione $\mathcal{F}(f)$ è tale che

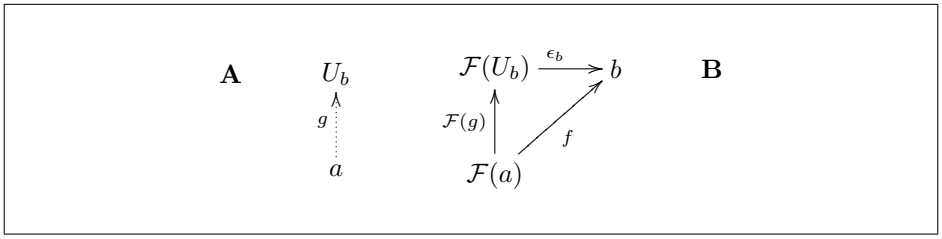


Figura 6: Diagramma per l'oggetto libero U_b

$\eta_a; \mathcal{U}(\mathcal{F}(f)) = f; \eta_{a'}$). Sfruttando il concetto di dualità si può facilmente dare una definizione alternativa ma del tutto equivalente di aggiunzione, basata sull'esistenza di una cosiddetta *counit* $\epsilon : \mathcal{U}; \mathcal{F} \Rightarrow 1_{\mathbf{B}}$, a partire dal funtore $\mathcal{F}$ e dal diagramma di oggetto libero (duale) in Figura 6: in questo caso $\mathcal{U}(b) = U_b$ e data $f : b' \to b \in \mathbf{B}$ si ha che $\mathcal{U}(f)$ è l'unica freccia g tale che $\mathcal{F}(g); \epsilon_b = \epsilon_{b'}; f$. Infine menzioniamo il fatto che date due aggiunzioni $\mathcal{F} \dashv \mathcal{U} : \mathbf{A} \to \mathbf{B}$ e $\mathcal{F}' \dashv \mathcal{U}' : \mathbf{B} \to \mathbf{C}$ la composizione degli aggiunti definisce ancora un'aggiunzione $\mathcal{F}; \mathcal{F}' \dashv \mathcal{U}'; \mathcal{U} : \mathbf{A} \to \mathbf{C}$.

Tornando all'esempio delle computazioni sequenziali, abbiamo che l'aggiunto sinistro associa ad ogni sistema di transizione S la categoria $\mathcal{C}(\mathsf{S})$ tale che $O_{\mathcal{C}(\mathsf{S})} = S_{\mathsf{S}}$ e le cui frecce sono ottenute a partire da quelle di S per chiusura[2] rispetto alle regole di inferenza

$$\frac{t : a \to b \in T_{\mathsf{S}}}{t : a \to b \in F_{\mathcal{C}(\mathsf{S})}} \qquad \frac{a \in O_{\mathsf{S}}}{id_a : a \to a \in F_{\mathcal{C}(\mathsf{S})}} \qquad \frac{f : a \to b,\ g : b \to c \in F_{\mathcal{C}(\mathsf{S})}}{f; g : a \to c \in F_{\mathcal{C}(\mathsf{S})}}$$

e soggette agli assiomi equazionali delle categorie (ovvero $id_a; f = f = f; id_b$ per ogni $f : a \to b \in F_{\mathcal{C}(\mathsf{S})}$ e $(f; g); h = f; (g; h)$ ogni volta che la composizione di f, g e h è ben definita). Sostanzialmente le frecce di $\mathcal{C}(\mathsf{S})$ sono identità $id_s = (\,)_s$ oppure sequenze finite $t_1; t_2; ...; t_n = (t_1\, t_2 ... t_n)$ di transizioni "componibili" di S. L'aggiunto destro "dimentica" la struttura algebrica delle frecce di una categoria e associa semplicemente ad ogni categoria $\mathbf{C} = (O, F, \delta_0, \delta_1, id_{\_}, \_; \_)$ il grafo soggiacente $\mathcal{U}^{\cdot}(\mathbf{C}) = (O, F, \delta_0, \delta_1)$. La unit dell'aggiunzione $\eta_{\mathsf{S}} : \mathsf{S} \to \mathcal{U}^{\cdot}(\mathcal{C}(\mathsf{S}))$ manda ogni stato in se stesso e ogni transizione $t : a \to b$ nella omonima transizione di $\mathcal{U}^{\cdot}(\mathcal{C}(\mathsf{S}))$. Infatti, presa una qualsiasi categoria $\mathbf{C}$ e un omomorfismo $h : \mathsf{S} \to \mathcal{U}^{\cdot}(\mathbf{C})$ è chiaro che h determina in modo univoco il funtore $\mathcal{C}(h) : \mathcal{C}(\mathsf{S}) \to \mathbf{C}$ perché definisce quale freccia della categoria $\mathbf{C}$ associare ad ogni transizione di S, e quindi determina per composizione l'immagine di tutte le altre frecce di $\mathcal{C}(\mathsf{S})$ dato che $\mathcal{C}(h)(id_a) = id_{h(a)}$ e $\mathcal{C}(h)(f; g) = \mathcal{C}(h)(f); \mathcal{C}(h)(g)$.

Classici esempi di aggiunzioni che sfrutteremo nei capitoli successivi sono quella tra **Set** e **CMon**, denotata $\_^{\oplus} \dashv \mathcal{U}_{\oplus}$ che associa ad ogni insieme S il monoide commutativo $(S^{\oplus}, \oplus, \varnothing)$ liberamente generato a partire dagli elementi di S e quella tra **Set** e **Mon**, denotata $\_^{\otimes} \dashv \mathcal{U}_{\otimes}$ che associa ad ogni insieme

[2]Nel senso che l'insieme delle frecce $F_{\mathcal{C}(\mathsf{S})}$ è il più piccolo insieme chiuso rispetto alle regole date.

S il monoide $(S^{\otimes}, \otimes, 0)$ liberamente generato a partire dagli elementi di S. Intuitivamente, $S^{\oplus}$ e $S^{\otimes}$ corrispondono rispettivamente all'insieme dei multi-insiemi (finiti) su S e all'insieme delle stringhe su S: il primo ha l'insieme vuoto $\varnothing$ come elemento neutro e l'unione tra multi-insiemi come prodotto, il secondo ha la stringa vuota 0 come elemento neutro e la concatenazione tra stringhe (giustapposizione di stringhe) come prodotto.

Infine rimarchiamo che così come la nozione di isomorfismo fornisce l'abituale criterio di uguaglianza tra insiemi, la nozione di isomorfismo naturale permette di definire un conveniente criterio di uguaglianza tra categorie. Due categorie $\mathbf{A}$ e $\mathbf{B}$ sono *isomorfe* se esiste un (funtore) isomorfismo tra esse in **Cat**; invece sono *equivalenti* se esistono due funtori $\mathcal{F} : \mathbf{A} \to \mathbf{B}$ e $\mathcal{G} : \mathbf{B} \to \mathbf{A}$ con due isomorfismi naturali $\varphi_{\mathbf{A}} : 1_{\mathbf{A}} \Rightarrow \mathcal{F};\mathcal{G} : \mathbf{A} \to \mathbf{B}$ e $\varphi_{\mathbf{B}} : 1_{\mathbf{B}} \Rightarrow \mathcal{G};\mathcal{F} : \mathbf{B} \to \mathbf{A}$. (Banalmente, due categorie isomorfe sono anche equivalenti). La nozione di equivalenza tra categorie è generalmente preferibile a quella di isomorfismo come criterio di uguaglianza tra categorie, perché la seconda è troppo dipendente da eventuali scelte arbitrarie compiute nell'immergere gli oggetti di $\mathbf{A}$ dentro $\mathbf{B}$ (o viceversa), mentre la nozione di equivalenza è perfettamente stabile rispetto alla caratterizzazione di proprietà (universali) "a meno di isomorfismi".

2 Reti di Petri e processi commutativi

Le *reti di Petri* [17, 18], introdotte nella tesi di dottorato di Carl A. Petri[3], sono il modello per la concorrenza più semplice e studiato, che ha riscosso un grosso successo anche in ambito interdisciplinare, trovando significative applicazioni in chimica, medicina, biologia, economia, fisica oltre che in matematica, informatica e ingegneria. Le reti di Petri sono largamente impiegate anche in ambito industriale, per il controllo di processi e del flusso di informazioni, e ci sono standard ISO basati su di esse. Anche se ad oggi esistono molteplici varianti delle reti di Petri, ideate per accrescerne l'espressività o per trattare il tempo e la probabilità, noi ci concentriamo sul modello minimale, delle *reti posto/transizione* (*reti P/T* in breve). Per differenza con i sistemi di transizione visti in precedenza, in breve le reti P/T permettono di avere molteplici istanze degli stati attive contemporaneamente e di avere transizioni che sincronizzano l'evoluzione di queste istanze, anche creandone di nuove o estinguendone di esistenti.

Sostanzialmente, una rete P/T è un grafo bipartito che ha dei nodi *posto*, dei nodi *transizione* e degli archi diretti tra essi (*archi di flusso*): possono esserci archi tra posti e transizioni e viceversa, ma non tra posti e posti o tra transizioni e transizioni; inoltre gli archi sono etichettati con un intero positivo, detto *peso*. I posti di ingresso di una transizione sono quelli dai quali parte un arco verso

[3]Carl Adam Petri (1926-) è un celebre matematico e informatico tedesco autore di significativi contributi nell'ambito di teorie e modelli per la coordinazione e l'interazione, tra i quali spiccano quelli relativi allo sviluppo della teoria delle reti che portano il suo nome. Nel 2007 è stato insignito con la "Academy Gold Medal of Honor" (medaglia accademica d'oro d'onore) dalla Academy of Transdisciplinary Learning and Advanced Studies (ATLAS, `http://www.theatlasnet.org/`).

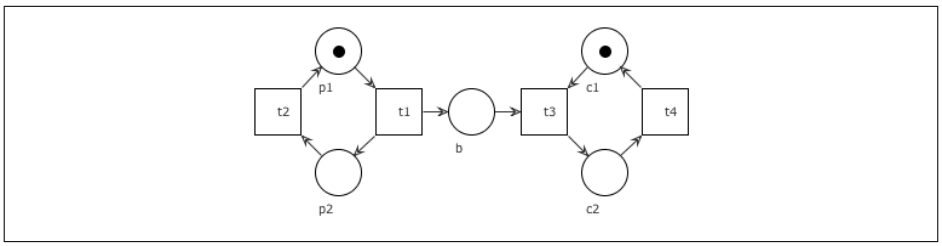


Figura 7: Sistema produttore-consumatore come rete P/T

la transizione e i posti di uscita sono quelli verso i quali esce un arco dalla transizione. I posti possono contenere un certo numero di *marche* (*token*) e lo stato della rete, detto *marcatura*, è dato dalla distribuzione delle marche sui posti. In Figura 7 riportiamo una classica rete P/T per modellare un sistema che comprende un produttore, un consumatore che interagiscono attraverso un *buffer* di comunicazione. Nella figura, i posti sono cerchi, le transizioni sono rettangoli e le marche sono pallini neri e tutti gli archi hanno implicitamente peso unario. La marcatura iniziale della rete comprende una marca in p_1 e una marca in c_1.

Una transizione è *abilitata* se la marcatura attuale comprende tante marche quante richieste dai posti di ingresso della transizione. L'esecuzione di una transizione abilitata è chiamata *scatto* e rimuove le marche dai posti di ingresso introducendo nuove marche sui posti di uscita, tutto in accordo ai pesi degli archi di flusso. Una caratteristica importante è che se ci sono abbastanza marche da far scattare più transizioni in modo concorrente, allora è possibile eseguire tutte queste transizioni simultaneamente mediante un *passo* della rete. Uno scatto è un caso particolare di passo. Nell'esempio, il produttore può essere in fase di rilascio (p_1) o di produzione (p_2). Le transizioni t_1 e t_2 regolano la sua attività, di modo che spostandosi dalla fase di rilascio a quella di produzione, venga prodotta anche una marca nel buffer b. Simmetricamente, il consumatore alterna una fase di attesa (c_1) di un prodotto dal buffer b con una fase di consumo (c_2) e le sue attività sono regolate dalle transizioni t_3 e t_4. La Figura 8 illustra alcune possibili evoluzioni della rete mediante passi. È facile notare che, anche se la rete è finita, le marcature raggiungibili da quella iniziale sono infinite, perché il produttore può continuare a generare un numero illimitato di marche nel buffer b senza che il consumatore ne prelevi alcuna (per esempio, si consideri l'evoluzione che alterna scatti di t_1 a quelli di t_2).

Formalmente, detto S l'insieme dei posti della rete, possiamo definire una marcatura come un multi-insieme $u : S \to \mathbb{N}$, in modo che $u(a)$ indichi il numero di marche presenti nella piazza a. Analogamente, gli archi di ingresso e di uscita di una transizione determinano due multi-insiemi delle marche richieste (in ingresso) e prodotte (in uscita) per uno scatto. Quindi una rete P/T può essere descritta semplicemente dicendo che generalizza il concetto di sistema di transizione al caso in cui gli stati sono strutturati come multi-insiemi. Ricordando che i multi-insiemi su S sono elementi di $S^{\oplus}$ (il monoide commutativo liberamente

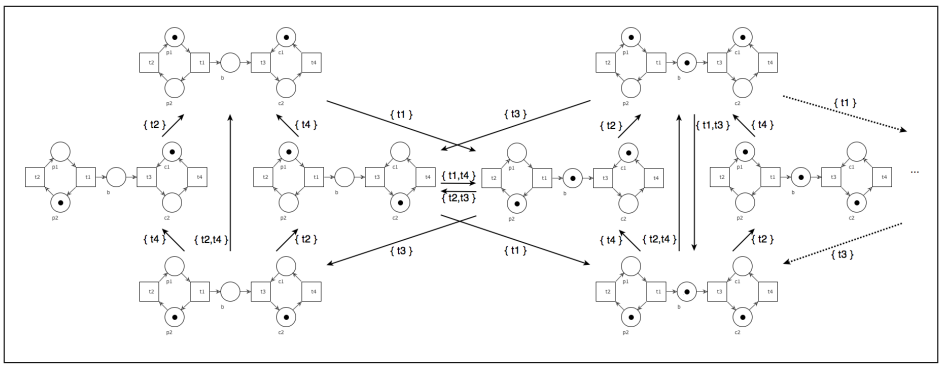


Figura 8: Alcune evoluzioni del sistema produttore-consumatore

generato da S), scriviamo $n_1a_1 \oplus n_2a_2 \oplus ... \oplus n_ka_k$ per il multi-insieme u tale che $u(a_i) = n_i$ per ogni $i \in [1, k]$ e formalizziamo l'unione $\oplus$ tra multi-insiemi ponendo $(u \oplus v)(a) = u(a) + v(a)$ per ogni $a \in S$. Inoltre, dato $m \in \mathbb{N}$, scriviamo $m \cdot u$ per il multi-insieme tale che $(m \cdot u)(a) = m \cdot u(a)$ per ogni $a \in S$.

Definizione 2.1 (rete P/T)
Una rete P/T è una quadrupla $N = (S, T, \partial_0, \partial_1)$, dove S è l'insieme dei posti, *T è l'insieme delle* transizioni *e le funzioni $\partial_0, \partial_1 : T \to S^{\oplus}$ assegnano, rispettivamente, ingressi e uscite ad ogni transizione.*

In modo del tutto equivalente, possiamo definire una rete P/T dicendo che è un sistema di transizione $N = (S^{\oplus}, T, \partial_0, \partial_1)$ il cui insieme degli stati è un monoide commutativo liberamente generato. Estendiamo alle reti le convenzioni enunciate per i sistemi di transizione. Per esempio, scriviamo $t : u \to v$ per $t \in T$ con $\partial_0(t) = u$ e $\partial_1(t) = v$ e ricordiamo che talvolta una rete P/T è accompagnata da una ben precisa *marcatura iniziale* $u_0 \in S^{\oplus}$ che determina le condizioni di partenza di una qualsiasi computazione. Con la notazione introdotta, scriviamo $p_1 \oplus c_1$ per la marcatura iniziale della rete di Figura 7, $t_1 : p_1 \to p_2 \oplus b$ e $t_2 : p_2 \to p_1$ per le transizioni del produttore e $t_3 : b \oplus c_1 \to c_2$ e $t_4 : c_2 \to c_1$ per quelle del consumatore.

Per definire il concetto di passo, indichiamo l'inclusione tra multi-insiemi con $\subseteq$ e la differenza tra multi-insiemi con $\ominus$; quindi $u \subseteq v$ se $u(a) \leq v(a)$ per ogni posto a, e $v \ominus u$ è definito quando $u \subseteq v$ e indica l'unico multi-insieme u' tale che $v = u \oplus u'$. Inoltre, dato un multi-insieme (finito) di transizioni $X : T \to \mathbb{N}$, introduciamo la notazione $\partial_i(X) = \sum_{t \in T} X(t) \cdot \partial_i(t)$ per $i \in [0, 1]$ e diciamo che X è *abilitato (concorrentemente)* nella marcatura u se $\partial_0(X) \subseteq u$.

Definizione 2.2 (Passo)
Date due marcature u e v e un multi-insieme (finito) di transizioni $X : T \to \mathbb{N}$ di una rete N, con X abilitato in u, diciamo che la rete N evolve da u a v con il passo *X, denotato da $u\ [X\rangle\ v$, se $v = u \ominus \partial_0(X) \oplus \partial_1(X)$.*

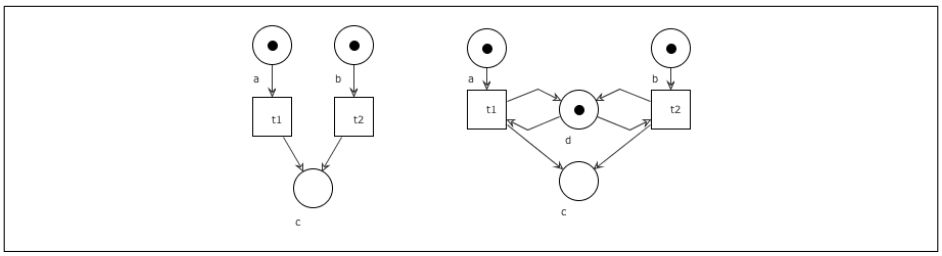


Figura 9: Concorrenza e alternanza

Per la rete di Figura 7, l'unico passo iniziale possibile è $p_1 \oplus c_1 \ [t_1\rangle \ p_2 \oplus b \oplus c_1$. Dallo stato $u_1 = p_2 \oplus b \oplus c_1$ sono invece possibili i soli passi:

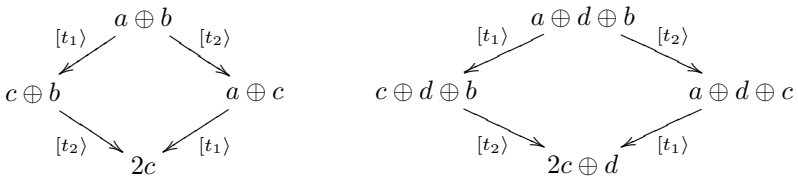
$$u_1 \ [t_2\rangle \ p_1 \oplus b \oplus c_1 \qquad u_1 \ [t_3\rangle \ p_2 \oplus c_2 \qquad u_1 \ [t_2 \oplus t_3\rangle \ p_1 \oplus c_2$$

Una *sequenza di passi* da u a u' è una sequenza $u_1 \ [X_1\rangle \ u_2 \ [X_2\rangle \ u_3 ... u_n \ [X_n\rangle \ u_{n+1}$, tale che $u = u_1$ e $u' = u_{n+1}$ con $n \geq 0$ (se $n = 0$ allora la sequenza si dice *vuota* e necessariamente $u = u'$). Sono esempi di sequenze:

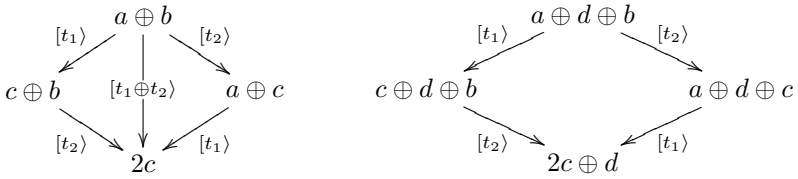
$p_1 \oplus c_1$
$p_1 \oplus c_1 \ [t_1\rangle \ p_2 \oplus b \oplus c_1 \ [t_2 \oplus t_3\rangle \ p_1 \oplus c_2 \ [t_1 \oplus t_4\rangle \ p_2 \oplus b \oplus c_1$
$p_1 \oplus c_1 \ [t_1\rangle \ p_2 \oplus b \oplus c_1 \ [t_2\rangle \ p_1 \oplus b \oplus c_1 \ [t_1\rangle \ p_2 \oplus 2b \oplus c_1 \ [t_2\rangle \ p_1 \oplus 2b \oplus c_1 \ [t_1\rangle \ p_2 \oplus 3b \oplus c_1$

Come sempre facciamo con le entità matematiche che consideriamo, la nozione di morfismo che ci interessa è quella che preserva la loro struttura. Nel caso dei sistemi di transizione questo significa preservare sorgente e destinazione. Nel caso delle reti P/T, significa preservare sorgente, destinazione e struttura monoidale degli stati. Quindi un morfismo tra reti P/T è sostanzialmente un morfismo tra i sistemi di transizione $(S^{\oplus}, T, \partial_0, \partial_1)$ a $(S'^{\oplus}, T', \partial'_0, \partial'_1)$ la cui componente sugli stati è un omomorfismo tra monoidi. Chiamiamo **Petri** la categoria delle reti P/T e loro morfismi. Trattandosi di monoidi liberamente generati, ricordiamo che l'omomorfismo sulle marcature è univocamente determinato dalle immagini dei singoli posti di S. In generale un morfismo tra reti P/T può mandare un posto di S in una marcatura su S'. Come caso particolare, chiamiamo *morfismi elementari* quelli che mandano ogni posto di S in un posto di S'.

Esaminiamo le possibili interpretazioni di una rete P/T dal punto di vista computazionale. In prima istanza, si potrebbe vedere la rete N come un sistema di transizione che ha $S_N^{\oplus}$ come insieme degli stati e l'insieme degli scatti $\{u \ [t\rangle \ v \mid t \in T_N, u \in S_N^{\oplus}, \partial_0(t) \subseteq u\}$ come insieme delle transizioni, con $src(u \ [t\rangle \ v) = u$ e $trg(u \ [t\rangle \ v) = v$. Nel caso poi sia nota una marcatura iniziale, è utile restingersi a considerare solo gli stati (marcature) raggiungibili e gli scatti tra esse per ridurre le dimensioni del sistema. Però gli scatti offrono una visione riduttiva, perché non evidenziano la concorrenza presente nel sistema. Per esempio, confrontando le due reti di Figura 9 si nota che i sistemi associati ad esse sarebbero isomorfi: entrambi permettono di eseguire prima t_1 e poi t_2 oppure prima t_2 e poi t_1.



Considerando i passi $\{u\ [X\rangle\ v \mid X \in T_N^{\oplus}, u \in S_N^{\oplus}, \partial_0(X) \subseteq u\}$ come transizioni (invece che gli scatti) è possibile notare come nella rete di sinistra sia possibile eseguire t_1 e t_2 in maniera concorrente, al contrario della rete di destra. A livello del sistema di transizione questo viene evidenziato dalla presenza di una diagonale nel "diamante" associato all'esecuzione di t_1 e t_2: nel caso di sinistra si parla di *diamante pieno*, nel caso di destra di *diamante vuoto*.



Il sistema di transizioni ottenuto considerando i passi viene detto *grafo delle marcature* e indicato con $\mathcal{M}(N)$. Si noti che sostanzialmente $\mathcal{M}(N)$ estende la struttura monoidale alle transizioni della rete e può essere caratterizzato come aggiunto sinistro tra la categoria **Petri** e la sua sotto-categoria che ha: (i) come oggetti quelle reti P/T le cui transizioni formano un monoide commutativo e dove ingressi e uscite sono omomorfismi tra monoidi; e (ii) come frecce quei morfismi la cui componente sulle transizioni è un omomorfismo tra monoidi. Data una rete P/T N, il grafo delle marcature $\mathcal{M}(N)$ è un sistema di transizioni che ha come stati le marcature di N ($S_{\mathcal{M}(N)} = S_N^{\oplus}$) e le cui transizioni sono ottenute a partire da quelle di N per chiusura rispetto alle regole di inferenza

$$\frac{t : u \to v \in T_N}{t : u \to v \in T_{\mathcal{M}(N)}} \qquad \frac{u \in S_N^{\oplus}}{id_u : u \to u \in T_{\mathcal{M}(N)}} \qquad \frac{f : u \to v,\ g : u' \to v' \in T_{\mathcal{M}(N)}}{f \oplus g : u \oplus u' \to v \oplus v' \in T_{\mathcal{M}(N)}}$$

e soggette agli assiomi equazionali dei monoidi commutativi (ovvero $id_{\varnothing} \oplus f = f$, $f \oplus g = g \oplus f$ e $(f \oplus g) \oplus h = f \oplus (g \oplus h)$ per ogni $f, g, h \in T_{\mathcal{M}(N)}$). Inoltre le funzioni sorgente, destinazione e identità devono essere omomorfismi tra monoidi, quindi le frecce sono soggette anche all'assioma $id_{u \oplus v} = id_u \oplus id_v$ per ogni $u, v \in S_{\mathcal{M}(N)}$ (per quanto riguarda sorgente e destinazione, la regola di inferenza per $f \oplus g$ garantisce la proprietà di omomorfismo per costruzione).

A questo punto si potrebbe pensare di generare lo spazio delle computazioni concorrenti partendo dal grafo delle marcature sfruttando la costruzione $\mathcal{C}$ vista precedentemente, cioè considerando la categoria $\mathcal{C}(\mathcal{M}(N))$. Dal punto di

vista categoriale la costruzione è soddisfacente, perché la composizione di aggiunti sinistri è ancora un aggiunto sinistro. Dal punto di vista della teoria della concorrenza invece tale costruzione non è soddisfacente, perché distingue computazioni di fatto concorrenti sulla base dell'ordine di trascrizione dei loro passi. Per esempio, nel caso del diamante pieno riportato sopra vorremmo identificare tutte e tre le sequenze

$$a \oplus b \; [t_1\rangle \; c \oplus b \; [t_2\rangle \; 2c \qquad a \oplus b \; [t_1 \oplus t_2\rangle \; 2c \qquad a \oplus b \; [t_2\rangle \; a \oplus c \; [t_1\rangle \; 2c$$

perché esiste un'unica computazione concorrente da $a \oplus b$ a $2c$. Invece nel caso del diamante vuoto è giusto distinguere le sequenze

$$a \oplus d \oplus b \; [t_1\rangle \; c \oplus d \oplus b \; [t_2\rangle \; 2c \oplus d \qquad a \oplus d \oplus b \; [t_2\rangle \; a \oplus d \oplus c \; [t_1\rangle \; 2c \oplus d$$

perché il posto d introduce una "corsa" tra t_1 e t_2 nell'uso dell'unica marca disponibile e quindi: (1) sussiste una dipendenza causale tra il primo e secondo evento di ogni sequenza; (2) le due sequenze sono mutuamente esclusive, perché la corsa viene vinta da t_1 oppure da t_2.

2.1 Costruzione universale per processi commutativi

L'idea di base dell'approccio è di estendere la struttura algebrica degli stati al livello delle computazioni in modo funtoriale, cioè facendo commutare le operazioni indotte con la struttura categoriale delle computazioni (identità e composizione). Nel caso delle reti P/T, la struttura degli stati è quella di un monoide commutativo e quindi dobbiamo dotare le computazioni di una struttura monoidale commutativa le cui operazioni soddisfino gli assiomi di funtorialità, ovvero nella scelta dello spazio delle computazioni dobbiamo passare dalle categorie alle categorie monoidali [11].

Definizione 2.3 (Categoria monoidale (stretta))
Una categoria monoidale (stretta) *è una tripla* $(\mathbf{C}, \_\otimes\_, e)$ *dove* $\mathbf{C}$ *è la cosiddetta categoria* sottostante *(con composizione* $\_;\_$ *e identità* $id\_$*),* $\_\otimes\_ : \mathbf{C} \times \mathbf{C} \to \mathbf{C}$ *è un funtore detto* prodotto tensore, *associativo sia sugli oggetti che sulle frecce, ed* e *è un oggetto di* $\mathbf{C}$ *che si comporta come elemento neutro rispetto al prodotto tensore (*id_e *è neutro per il prodotto tensore di frecce).*

L'aggettivo "stretta" è riferito al fatto che gli assiomi di monoidalità per $\_ \otimes \_$ ed e sono richiesti valere esattamente invece che a meno di isomorfismo naturale. Di seguito assumiamo implicitamente che la monoidalità sia sempre stretta.

Definizione 2.4 (Funtore monoidale (stretto))
Siano $\mathbf{C}$ *e* $\mathbf{C}'$ *due categorie monoidali. Un* funtore monoidale (stretto) *da* $\mathbf{C}$ *a* $\mathbf{C}'$ *è un funtore* $\mathcal{F} : \mathbf{C} \to \mathbf{C}'$ *tale che* $\mathcal{F}(e) = e'$ *e* $\mathcal{F}(f \otimes g) = \mathcal{F}(f) \otimes' \mathcal{F}(g)$ *per ogni* $f, g \in \mathbf{C}$.

$$\frac{u \in S_N^{\oplus}}{id_u : u \to u \in F_{\mathcal{T}(N)}} \qquad \frac{t \in T_N}{t : \partial_0(t) \to \partial_1(t) \in F_{\mathcal{T}(N)}}$$

$$\frac{\alpha : u \to v,\ \beta : u' \to v' \in F_{\mathcal{T}(N)}}{\alpha \oplus \beta : u \oplus u' \to v \oplus v' \in F_{\mathcal{T}(N)}} \qquad \frac{\alpha : u \to u',\ \beta : u' \to v \in F_{\mathcal{T}(N)}}{\alpha ; \beta : u \to v \in F_{\mathcal{T}(N)}}$$

Figura 10: Regole di inferenza per $F_{\mathcal{T}(N)}$

Anche questa volta, l'aggettivo "stretto" è riferito al fatto che le condizioni di uguaglianza sono richieste esattamente invece che a meno di isomorfismo naturale. Di seguito assumiamo tacitamente che la monoidalità del funtore sia sempre stretta.

Una categoria monoidale $(\mathbf{C}, \_ \otimes \_, e)$ è detta *simmetrica stretta* se il prodotto tensore è anche commutativo. Più avanti avremo bisogno di considerare anche categorie monoidali simmetriche "non strette", dove la commutatività del tensore è richiesta a meno di un isomorfismo naturale: per evitare ambiguità manteniamo quindi la distinzione tra categorie monoidali simmetriche strette e categorie monoidali simmetriche (non strette). Di seguito indichiamo una categoria monoidale simmetrica stretta con la notazione $(\mathbf{C}, \_ \oplus \_, e)$, usando $\oplus$ invece di $\otimes$ per evidenziare che la commutatività è stretta. Chiamiamo **SSMC** la categoria delle categorie monoidali simmetriche strette e funtori monoidali.

Dato che gli stati delle reti P/T si basano su monoidi commutativi e liberamente generati, invece di considerare la categoria **SSMC**, che contiene troppi oggetti, dobbiamo prendere la sua sotto-categoria piena $\mathbf{SSMC}^{\oplus}$ che comprende solo categorie monoidali simmetriche strette i cui insiemi degli oggetti siano monoidi commutativi liberamente generati. In questo modo, il funtore (aggiunto destro) $\mathcal{U}^{\oplus} : \mathbf{SSMC}^{\oplus} \to \mathbf{Petri}$ è definito banalmente ponendo $\mathcal{U}^{\oplus}((S^{\oplus}, F, \delta_0, \delta_1, id, \_;\_), \_ \oplus \_, e) = (S, F, \delta_0, \delta_1)$, cioè scegliendo i "generatori" del monoide $S^{\oplus}$ come posti della rete e prendendo le frecce della categoria come transizioni (ma "dimenticando" la loro struttura categoriale e monoidale). L'estensione di $\mathcal{U}^{\oplus}$ alle frecce di $\mathbf{SSMC}^{\oplus}$ è immediata, dato che un funtore monoidale $\mathcal{F} : (\mathbf{C}, \oplus, e) \to (\mathbf{C}', \oplus', e') \in \mathbf{SSMC}^{\oplus}$ può essere visto direttamente come morfismo tra le reti $\mathcal{U}^{\oplus}(\mathbf{C}, \oplus, e)$ e $\mathcal{U}^{\oplus}(\mathbf{C}', \oplus', e')$.

Viceversa, l'aggiunto sinistro $\mathcal{T} : \mathbf{Petri} \to \mathbf{SSMC}^{\oplus}$ è definito come segue. Per ogni rete N, la categoria $\mathcal{T}(N)$ è tale che $O_{\mathcal{T}(N)} = S_N^{\oplus}$ e le cui frecce sono ottenute a partire dalle transizioni di N per chiusura rispetto alle regole di inferenza in Figura 10, modulo gli assiomi equazionali in Figura 11 (che devono essere soddisfatti per tutte le frecce α, α', β, β', δ che possono essere composte correttamente e per tutti i multi-insiemi u e v).

Intuitivamente: id_u descrive la computazione vuota relativa alla marcatura u, la freccia $t : \partial_0(t) \to \partial_1(t)$ descrive lo scatto $\partial_0(t)\ [t\rangle\ \partial_1(t)$ della transizione t nella minima marcatura che possa abilitarla, la composizione $\alpha ; \beta$ descrive la computazione ottenuta eseguendo α e β in sequenza e la composizione $\alpha \oplus \beta$ descrive la computazione ottenuta eseguendo in modo concorrente le computazioni α e β. Come casi particolari: la composizione $t \oplus id_u$ descrive lo scatto

monoide:	$id_\varnothing \oplus \alpha = \alpha \quad (\alpha \oplus \beta) \oplus \delta = \alpha \oplus (\beta \oplus \delta) \quad \alpha \oplus \beta = \beta \oplus \alpha$
categoria:	$\alpha; id_u = \alpha = id_v; \alpha \qquad (\alpha; \beta); \delta = \alpha; (\beta; \delta)$
funtorialità:	$id_u \oplus id_v = id_{u \oplus v} \qquad (\alpha; \beta) \oplus (\alpha'; \beta') = (\alpha \oplus \alpha'); (\beta \oplus \beta')$

Figura 11: Assiomatizzazione equazionale di $\mathcal{T}(N)$

$\partial_0(t) \oplus u \; [t\rangle \; \partial_1(t) \oplus u$ e la composizione $t_1 \oplus t_2 \oplus ... \oplus t_n \oplus id_u$ descrive il passo $\partial_0(t_1 \oplus t_2 \oplus ... \oplus t_n) \oplus u \; [t_1 \oplus t_2 \oplus ... \oplus t_n\rangle \; \partial_1(t_1 \oplus t_2 \oplus ... \oplus t_n) \oplus u$.

La unit $\eta_N : N \to \mathcal{U}^\oplus(\mathcal{T}(N))$ dell'aggiunzione è definita come l'identità sulle marcature e manda ogni transizione $t \in N$ nella omonima transizione di $\mathcal{U}^\oplus(\mathcal{T}(N))$. Analogamente a quanto visto nel caso sequenziale, presa una qualsiasi categoria monoidale $\mathbf{C} \in \mathbf{SSMC}^\oplus$ e un qualsiasi morfismo di reti $f : N \to \mathcal{U}^\oplus(\mathbf{C})$, è immediato verificare che questo determina univocamente il funtore monoidale $\mathcal{T}(f) : \mathcal{T}(N) \to \mathbf{C}$ che si comporta come f sugli oggetti di $\mathcal{T}(N)$ e che è definito induttivamente sulle frecce come segue:

$$\mathcal{T}(f)(id_a) = id_{\mathcal{T}(f)(a)} \quad \mathcal{T}(f)(\alpha \oplus \beta) = \mathcal{T}(f)(\alpha) \oplus \mathcal{T}(f)(\beta) \quad \mathcal{T}(f)(\alpha; \beta) = \mathcal{T}(f)(\alpha); \mathcal{T}(f)(\beta)$$

L'aggiunzione $\mathcal{T} \dashv \mathcal{U}^\oplus$, oltre ad essere soddisfacente dal punto di vista "categoriale", ha un ben preciso significato computazionale, dato che $\mathcal{T}(N)$ risulta essere isomorfa [13] alla categoria monoidale $\mathcal{CP}(N)$ dei cosiddetti *processi commutativi* di N, definiti concretamente da Best e Devillers [2] come sequenze di scatti modulo una particolare relazione di equivalenza (che permette di scambiare l'ordine di due scatti quando le corrispondenti transizioni siano abilitate concorrentemente). Per esempio, dato un qualsiasi passo $t_1 \oplus t_2 \oplus ... \oplus t_n : u \to v$ si osservi che la funtorialità di $\oplus$ permette di identificare tutte le sequenze di passi da u a v che complessivamente comprendano l'esecuzione di tutte (e sole) le transizioni $t_1, t_2, ..., t_n$. Lo si può dimostrare facilmente per induzione su n: il caso $n = 1$ è banale; per il caso induttivo, preso un qualsiasi $t_i : u_i \to v_i$ e detto $\alpha = \bigoplus_{j \in [1,n], j \neq i} t_j : u' \to v'$ possiamo scrivere $t_1 \oplus t_2 \oplus ... \oplus t_n = t_i \oplus \alpha = (t_i; id_{v_i}) \oplus (id_{u'}; \alpha)$ sfruttando la commutatività e la proprietà delle identità e infine $(t_i; id_{v_i}) \oplus (id_{u'}; \alpha) = (t_i \oplus id_{u'}) \; ; \; (id_{v_i} \oplus \alpha)$ per la funtorialità. Per ipotesi induttiva α rappresenta una qualsiasi permutazione degli scatti relativi alle $n - 1$ transizioni $t_1, ..., t_{i-1}, t_{i+1}, ..., t_n$ e dato che t_i è stato scelto arbitrariamente $t_1 \oplus t_2 \oplus ... \oplus t_n$ rappresenta una qualsiasi permutazione degli scatti relativi alle n transizioni $t_1, ..., t_n$.

Riprendendo l'esempio del sistema produttore-consumatore, mostriamo come la costruzione $\mathcal{T}(N)$ identifica espressioni diverse che corrispondono alla stessa computazione concorrente. Per evitare un uso intensivo di parentesi, assumiamo per convenzione che $\oplus$ abbia precedenza su ;. Assumiamo che la marcatura di partenza sia $p_2 \oplus b \oplus c_1$ (raggiunta dalla marcatura iniziale dopo il passo $t_1 \oplus id_{c_1}$).

$$
\begin{array}{rcll}
t_2 \oplus (t_3; t_4) & = & (t_2; id_{p_1}) \oplus (t_3; t_4) & \text{(identità)} \\
& = & t_2 \oplus t_3 \,;\, id_{p_1} \oplus t_4 & \text{(funtorialità)} \\
& = & (id_{p_2}; t_2) \oplus (t_3; id_{c_2}) \,;\, id_{p_1} \oplus t_4 & \text{(identità)} \\
& = & id_{p_2} \oplus t_3 \,;\, t_2 \oplus id_{c_2} \,;\, id_{p_1} \oplus t_4 & \text{(funtorialità)} \\
& = & id_{p_2} \oplus t_3 \,;\, (t_2; id_{p_1}) \oplus (id_{c_2}; t_4) & \text{(funtorialità)} \\
& = & id_{p_2} \oplus t_3 \,;\, (t_2 \oplus t_4) & \text{(identità)}
\end{array}
$$

Andando a confrontare la costruzione $\mathcal{T}$ con $\mathcal{C};\mathcal{M}$ si scopre che la differenza sostanziale risiede nell'assioma di funtorialità $(\alpha;\beta)\oplus(\alpha';\beta') = (\alpha\oplus\alpha');(\beta\oplus\beta')$ delle categorie monoidali che quindi caratterizza il passaggio dalla semantica "alternante" basata su sequenze di passi a quella concorrente basata su processi commutativi

2.2 Colimiti e composizionalità

Nella teoria delle categorie, le proprietà universali vengono usate anche per caratterizzare (a meno di isomorfismi) certi modi di comporre i sistemi (oggetti della categoria). Un aspetto interessante delle costruzioni universali è che preservano queste composizioni, come chiarito di seguito.

La composizione di sistemi viene definita scegliendo un particolare insieme di oggetti e di frecce tra loro nella categoria $\mathbf{C}$ e cercando, se esiste, un oggetto che possa riassumere (approssimare) quella struttura meglio di tutti gli altri, nel senso di poter fattorizzare in modo univoco ogni altra approssimazione. L'insieme di oggetti e frecce è detto *diagramma*, una sua approssimazione è detta *(co)cono* e quella universale, quando esiste, è detta *(co)limite*. Il prefisso "co" serve a sottolineare che sfruttando la dualità si possono definire due tipi di costruzioni: i co-limiti sono preservati dagli aggiunti sinistri, i limiti dagli aggiunti destri. Dato che a noi interessa comporre modelli e poi generare lo spazio delle computazioni tramite aggiunti sinistri, di seguito ci concentriamo sui co-limiti. Trattandosi di proprietà universali, è superfluo ricordare che il (co)limite è unico a meno di isomorfismi.

Formalmente, un diagramma $D : G \to \mathbf{C}$ può essere definito come un omomorfismo tra grafi da un certo grafo G alla categoria $\mathbf{C}$ (vista come grafo). Un co-cono commutativo con base D consiste di un oggetto $c \in \mathbf{C}$ assieme ad una famiglia di frecce $\{p_a : D(a) \to c\}_{a\in G}$, una per ogni nodo di G, tali che per ogni arco $f : a \to b \in G$ si ha $p_a = D(f); p_b$ (si veda la Figura 12, sinistra).

Definizione 2.5 (Co-limite)
Dato un diagramma $D : G \to \mathbf{C}$, un co-cono commutativo $(c, \{p_a : D(a) \to c\}_{a\in G})$ con base D è detto co-limite *se per ogni altro co-cono commutativo $(c', \{p'_a : D(a) \to c'\}_{a\in G})$ con base D esiste un'unica freccia $q : c \to c'$ tale che per ogni oggetto $a \in G$ vale $p_a = q; p'_a$ (si veda la Figura 12, destra).*

Abusando la notazione, di seguito usiamo $\Delta(D)$ per indicare sia il co-cono colimite $(c, \{p_a : D(a) \to c\}_{a\in G})$ che il solo oggetto c vertice del co-cono.

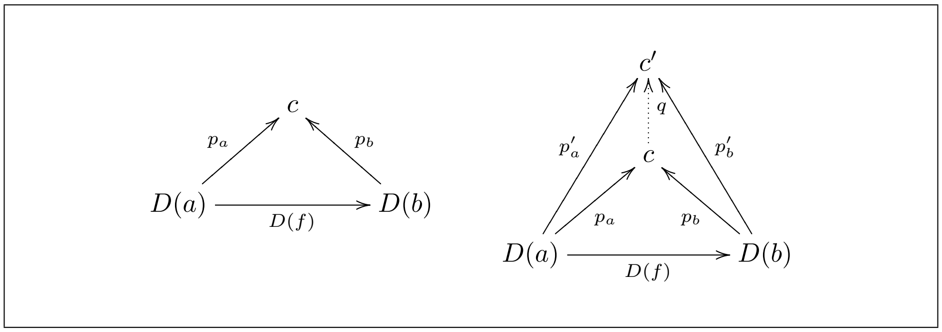


Figura 12: Esemplificazione di un co-cono commutativo (a sinistra) e un co-limite (a destra)

Per illustrare il concetto di co-limite, l'esempio classico è quello che considera il grafo comprendente due nodi a e b (e nessuna freccia). Un diagramma in **Set** è dato da due insiemi $D(a)$ e $D(b)$. Un co-cono commutativo consiste di un insieme C e due funzioni $p_a : D(a) \to C$ e $p_b : D(b) \to C$. L'unione disgiunta $D(a) \uplus D(b)$ con le ovvie iniezioni $i_1 : D(a) \to D(a) \uplus D(b)$ e $i_2 : D(b) \to D(a) \uplus D(b)$ definisce un co-limite. Infatti è facile vedere che dato un qualsiasi altro insieme C con due funzioni $p_a : D(a) \to C$ e $p_b : D(b) \to C$, la freccia q è definita univocamente scegliendo $q(i_1(x)) = p_a(x)$ per ogni $x \in D(a)$ e $q(i_2(y)) = p_b(y)$ per ogni $y \in D(b)$: la definizione è sicuramente ben posta perché i_1 e i_2 sono iniettive ed è completa perché ogni elemento di $D(a) \uplus D(b)$ è nell'immagine $i_1(D(a))$ o nell'immagine $i_2(D(b))$. Questo è un altro esempio di come la teoria delle categorie permetta di generalizzare concetti ben noti della teoria degli insiemi.

Nel caso di composizione di modelli, un co-limite di particolare interesse è quello che permette di definire l'unione di due strutture indentificando però una parte comune. Questa costruzione si chiama *pushout* e corrisponde a prendere il co-limite associato al grafo del tipo $\bullet \longleftarrow \bullet \longrightarrow \bullet$.

Per esempio, il sistema di transizioni PD della Figura 1 può essere ottenuto come co-limite del diagramma in Figura 13 (sinistra) e la rete del sistema produttore consumatore può essere ottenuta come co-limite del diagramma in Figura 13 (destra), dove i morfismi tra modelli sono rappresentati con frecce tratteggiate.

Il fatto che l'aggiunto sinistro preserva i colimiti, significa che possiamo indifferentemente comporre modelli mediante un certo colimite e poi costruire lo spazio delle computazioni relativo al risultato della composizione, oppure possiamo prima calcolare gli spazi delle computazioni dei modelli di partenza e poi comporli calcolando il colimite. Formalmente, la situazione comprende un'aggiunzione $\mathcal{F} \dashv \mathcal{U} : \mathbf{A} \to \mathbf{B}$ (con unit η) tra due categorie e un diagramma $D : G \to \mathbf{A}$ per il quale esiste il colimite $\Delta(D) \in \mathbf{A}$. Sotto queste condizioni siamo garantiti che: (1) il colimite del diagramma $\mathcal{F}(D(G)) \in \mathbf{B}$ esiste; (2) $\mathcal{F}(\Delta(D))$ è un colimite di $\mathcal{F}(D(G))$. Questa proprietà può essere dimostrata in molti modi, sfruttando caratterizzazioni diverse delle aggiunzioni e proprietà di composizione degli aggiunti. Di seguito diamo una descrizione intuitiva che

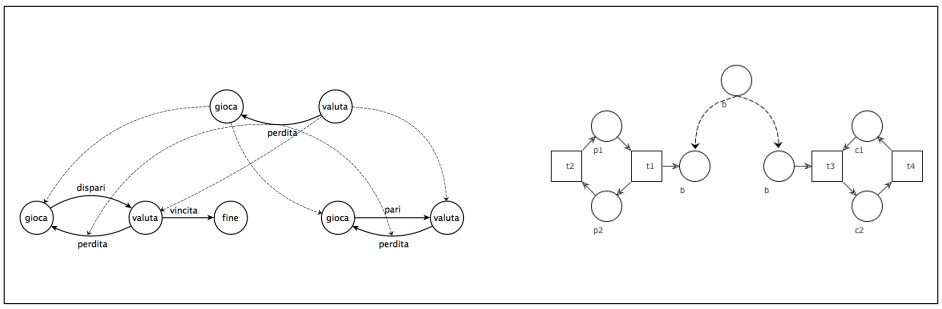


Figura 13: Composizione di sistemi di transizione (sinistra) e di reti (destra)

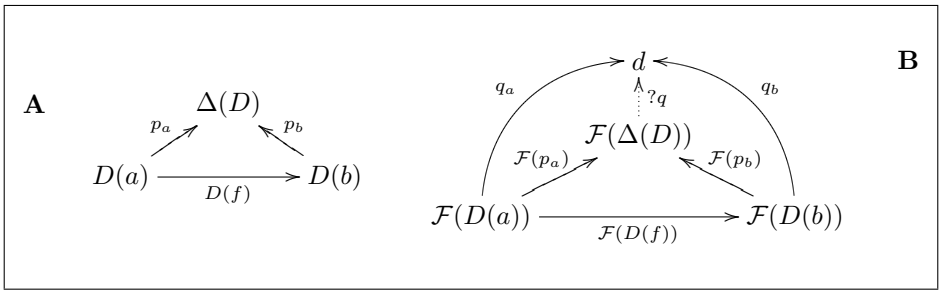


Figura 14: Ricerca del colimite del diagramma $\mathcal{F}(D(G))$

sfrutta solo le definizioni elementari già presentate.

Dato il colimite $(\Delta(D), \{p_a : D(a) \to \Delta(D)\}_{a \in G})$ di D in $\mathbf{A}$, come prima cosa, si noti che $\mathcal{F}(\Delta(D))$ è un co-cono del diagramma $\mathcal{F}(D(G))$ ($\mathcal{F}(\Delta(D))$ commuta, perché tutti i funtori preservano i diagrammi che commutano). Quindi resta da dimostrare che per ogni altro co-cono $(d, \{q_a : \mathcal{F}(D(a)) \to d\}_{a \in G})$ in $\mathbf{B}$ del diagramma $\mathcal{F}(D(G))$ esiste un'unica freccia $q : \mathcal{F}(\Delta(D)) \to d$ tale che $\mathcal{F}(p_a); q = q_a$ per ogni $a \in G$ (si veda la Figura 14).

Per prima cosa dimostriamo l'esistenza di q e poi la sua unicità. Per definire q possiamo proiettare il co-cono $(d, \{q_a : \mathcal{F}(D(a)) \to d\}_{a \in G})$ su $\mathbf{A}$ mediante $\mathcal{U}$. Pre-componendo le frecce $\mathcal{U}(q_a)$ con $\eta_{D(a)} : D(a) \to \mathcal{U}(\mathcal{F}(D(a))$ si ottiene un co-cono $(\mathcal{U}(d), \{\eta_{D(a)}; \mathcal{U}(q_a) : D(a) \to \mathcal{U}(d)\}_{a \in G} \in \mathbf{A}$ per il diagramma D. Dato che $\Delta(D)$ è un colimite di D deve esistere una freccia $p : \Delta(D) \to \mathcal{U}(d)$ tale che $p_a; p = \eta_{D(a)}; \mathcal{U}(q_a)$ per ogni $a \in G$ (si veda la Figura 15).

A questo punto poniamo $q : \mathcal{F}(\Delta(D)) \to d \in \mathbf{B}$ definita come l'unica freccia tale che $p = \eta_{\Delta(D)}; \mathcal{U}(q)$, la cui esistenza è garantita dall'aggiunzione $\mathcal{F} \dashv \mathcal{U}$ e facciamo vedere che $\mathcal{F}(p_a); q = q_a$ (per ogni a). Per dimostrare l'uguaglianza, consideriamo l'unica freccia $r : \mathcal{F}(D(a)) \to d \in \mathbf{B}$ tale che $p_a; p = \eta_{D(a)}; \mathcal{U}(r)$ (la cui esistenza è garantita dall'aggiunzione $\mathcal{F} \dashv \mathcal{U}$, si veda la Figura 16) e mostriamo che $\mathcal{F}(p_a); q = r = q_a$.

Data la proprietà universale che caratterizza r, basta verificare separatamente le uguaglianze $p_a; p = \eta_{D(a)}; \mathcal{U}(\mathcal{F}(p_a); q)$ e $p_a; p = \eta_{D(a)}; \mathcal{U}(q_a)$. Per la prima uguaglianza, si ha $\eta_{D(a)}; \mathcal{U}(\mathcal{F}(p_a); q) = \eta_{D(a)}; \mathcal{U}(\mathcal{F}(p_a)); \mathcal{U}(q) = p_a; \eta_{\Delta(D)}; \mathcal{U}(q) = p_a; p$ (si veda la Figura 16). Per la seconda uguaglianza, si veda direttamente la

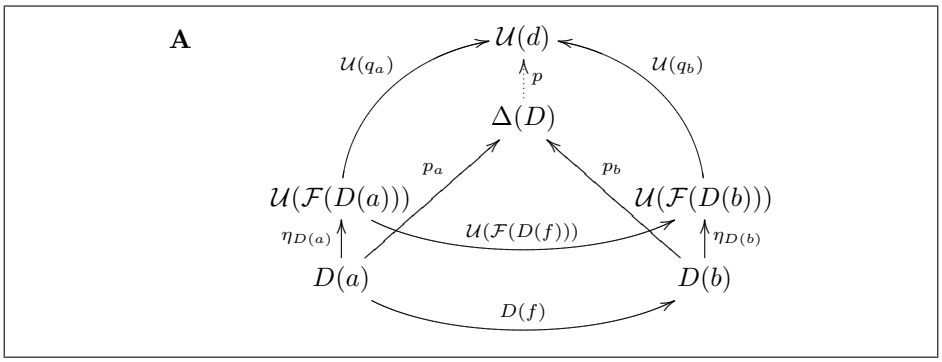


Figura 15: Esistenza di $p : \Delta(D) \to \mathcal{U}(d) \in \mathbf{A}$

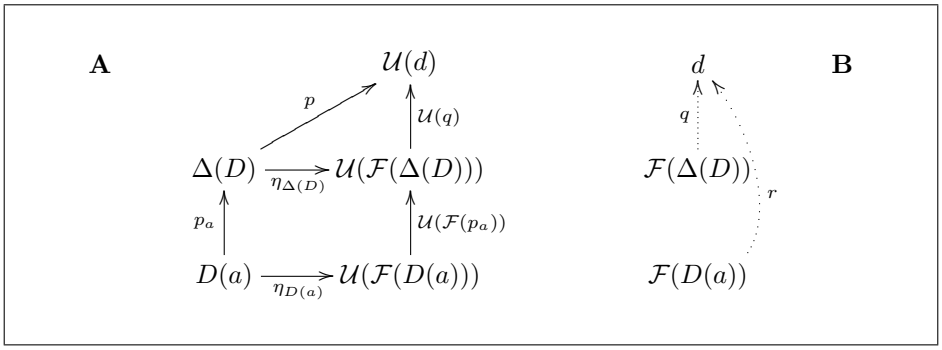


Figura 16: Caratterizzazione di $q : \mathcal{F}(\Delta(D)) \to d \in \mathbf{B}$

Figura 15.

L'unicità di q può essere dimostrata assumendo che esista un'altra freccia $q' : \mathcal{F}(\Delta(D)) \to d \in \mathbf{B}$ che soddisfa $\mathcal{F}(p_a); q' = q_a$ per ogni $a \in G$ e dimostrando che deve valere necessariamente $q = q'$. Prendiamo $p' = \eta_{\Delta(D)}; \mathcal{U}(q')$. Abbiamo $p_a; p' = p_a; \eta_{\Delta(D)}; \mathcal{U}(q') = \eta_{D(a)}; \mathcal{U}(\mathcal{F}(p_a)); \mathcal{U}(q') = \eta_{D(a)}; \mathcal{U}(\mathcal{F}(p_a); q') = \eta_{D(a)}; \mathcal{U}(q_a) = p_a; p$. Ma allora, per l'unicità del colimite, deve essere $p = p' = \eta_{\Delta(D)}; \mathcal{U}(q')$ e per la proprietà dell'aggiunzione deve essere $q = q'$ (perché q è l'unica freccia che soddisfa $p = \eta_{\Delta(D)}; \mathcal{U}(q)$).

3 Reti di Petri e processi non commutativi

I processi commutativi offrono una visione della concorrenza su reti P/T che segue la cosiddetta *filosofia collettiva*, dove tutte le marche presenti nella stessa piazza sono sostanzialmente indistinguibili. Questo si contrappone alla cosiddetta *filosofia individuale*, secondo la quale ogni marca deve essere ben distinta dalle altre perché rappresenta una ben precisa risorsa del sistema, prodotta in certe circostanze da determinati eventi e che ad esempio può essere annotata con la storia passata degli eventi che l'hanno prodotta. Per capire la differenza, immaginiamo che la marcatura iniziale del sistema produttore-consumatore di Figura 7 sia $p_1 \oplus c_1 \oplus c_2$. In questo caso abbiamo un solo produttore ma due consumatori distinti: uno immediatamente pronto per ricevere e l'altro nella fase di consumo. Se consideriamo la sequenza $t_1 \oplus id_{c_1 \oplus c_2}$; $id_{p_1} \oplus t_3 \oplus t_4$ è evidente che solo il produttore inizialmente in c_1 può aver consumato la marca prodotta in b da t_1. Però gli assiomi dei processi commutativi consentono di dimostrare l'uguaglianza $t_1 \oplus id_{c_1 \oplus c_2}$; $id_{p_1} \oplus t_3 \oplus t_4 = t_1 \oplus id_{c_1} \oplus t_4$; $id_{p_1 \oplus c_1} \oplus t_3$, dove il secondo termine mostra che ciascuno dei due consumatori potrebbe aver consumato la marca prodotta da t_1. Per risolvere questa ambiguità abbiamo bisogno di tracciare l'identità delle risorse attraverso le varie fasi della computazione: per esempio, vorremmo che dopo il passo $t_1 \oplus id_{c_1} \oplus t_4$ le due marche in c_1 fossero distinte dal fatto che una era quella presente inizialmente e l'altra quella prodotta da t_4.

A livello computazionale, un modo di procedere è quello di considerare i cosiddetti *processi deterministici* introdotti da Goltz e Reisig [9]. Questi possono essere descritti immaginando di "srotolare" la rete durante l'esecuzione delle transizioni, creando nuove istanze dei posti invece che anonime marche e dove la corrispondenza tra la rete "srotolata", chiamata *rete di occorrenze*, e quella originale è descritta da un'opportuna proiezione (un morfismo tra reti). Le reti di occorrenze soddisfano proprietà di aciclicità e di determinismo in grado di garantire che se siamo in grado di distinguere le risorse iniziali, allora saremo anche in grado di distinguere le risorse disponibili in qualsiasi fase della computazione.

Definizione 3.1 (Rete di occorrenze)
Una rete di occorrenze (deterministica) *è una rete P/T finita e aciclica* $P = (S, T, \partial_0, \partial_1)$ *tale che per ogni transizione* $t \in T$, $\partial_0(t)$ *e* $\partial_1(t)$ *sono insiemi (tutti*

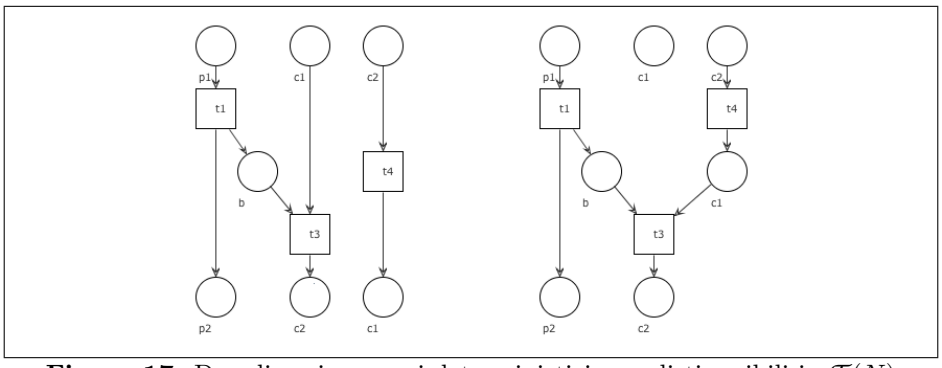


Figura 17: Due diversi processi deterministici non distinguibili in $\mathcal{T}(N)$

gli archi hanno peso unario), e per ogni coppia di transizioni $t_0 \neq t_1 \in T$, $\partial_i(t_0) \cap \partial_i(t_1) = \varnothing$, *per* $i \in [0, 1]$.

Sostanzialmente, ogni posto ha al più un arco entrante e al più un arco uscente. I posti senza archi entranti rappresentano risorse disponibili inizialmente nel sistema. I posti senza archi uscenti rappresentano risorse disponibili al termine dell'esecuzione della rete. L'arco entrante in un posto, quando presente, identifica la transizione (evento) che produce la risorsa, quello uscente, quando presente, l'unico evento che la consuma.

Definizione 3.2 (Processo deterministico)
Un processo deterministico *della rete* $N \in$ **Petri** *è un morfismo di reti* $\pi : P \to N$, *dove* P *è una rete di occorrenze deterministica e* π *è elementare (manda posti in posti, invece che posti in marcature).*

Il morfismo π è spesso rappresentato semplicemente etichettando posti e transizioni di P con la loro immagine attraverso π. Due processi $\pi : P \to N$ e $\pi' : P' \to N$ sono isomorfi se esiste un isomorfismo tra reti $f : P \to P'$ tale che $f; \pi' = \pi$. La Figura 17 mostra i due processi del sistema produttore-consumatore discussi in precedenza: chiaramente i due processi non sono isomorfi, ma non è possibile distinguerli in $\mathcal{T}(N)$. La differenza è essenziale se si vuole descrivere correttamente le computazioni concorrenti, dato che nel processo a destra le occorrenze delle transizioni t_4 e t_3 risultano sequenzializzate (cioè t_3 *dipende causalmente* da t_4) mentre nel processo a sinistra t_4 e t_3 sono concorrenti.

Dal punto di vista algebrico, l'intuizione è di rinunciare alla commutatività del prodotto tensore, in modo da distinguere termini quali $id_{c_1} \otimes t_3$ e $t_3 \otimes id_{c_1}$, che corrispondono rispettivamente al fatto che il "primo" consumatore resta fermo mentre il "secondo" accede al buffer e viceversa il "primo" accede al buffer e il "secondo" resta fermo. La costruzione dello spazio di computazioni all'interno della categoria delle categorie monoidali e funtori monoidali non è però soddisfacente. Per esempio, considerando la marcatura $2b \oplus 2c_1$ non potremmo

esprimere il passo concorrente dove il "primo" consumatore accede alla "seconda" marca in b, mentre il "secondo" consumatore accede alla "prima" marca in b (l'unico passo concorrente è $t_3 \otimes t_3$ e le sequenze $id_b \otimes t_3 \otimes id_{c_1}; id_{c_2} \otimes t_3$ e $id_{c_1} \otimes t_3 \otimes id_b; id_{c_2} \otimes t_3$ vengono distinte). L'idea è di considerare categorie monoidali simmetriche non strette, in modo da poter permutare le marcature come più conveniente. Sotto indichiamo con $\_2 \otimes \_1$ il tensore che scambia l'ordine dei due argomenti (cioè $(\_2 \otimes \_1)(f, g) = g \otimes f$).

Definizione 3.3 (Categoria monoidale simmetrica)
Una categoria monoidale simmetrica (non stretta) *è una quadrupla* $(\mathbf{C}, \_\otimes\_, e, \gamma)$ *dove* $(\mathbf{C}, \_ \otimes \_, e)$ *è una categoria monoidale e* $\gamma : \_1 \otimes \_2 \Rightarrow \_2 \otimes \_1$ *è un isomorfismo naturale che soddisfa i cosiddetti assiomi di coerenza di Kelly-MacLane (per ogni* $a, b, c \in \mathbf{C}_O$*):*

$$\gamma_{a\otimes b,c} = (id_a \otimes \gamma_{b,c}); (\gamma_{a,c} \otimes id_b) \qquad \gamma_{a,b}; \gamma_{b,a} = id_{a\otimes b}$$

Una categoria monoidale simmetrica $(\mathbf{C}, \_ \otimes \_, e, \gamma)$ *è stretta quando* γ *è la trasformazione naturale identità.*

Chiamiamo *simmetria* una qualsiasi freccia di $\mathbf{C}$ che possa essere ottenuta come composizione (con $\_;\_$ e $\_ \otimes \_$) di frecce identità e frecce della famiglia γ. In sostanza, gli assiomi di coerenza servono a garantire che tutte le simmetrie che effettuano la stessa permutazione di oggetti siano identificate.[4] L'assioma di naturalità associato a γ impone che, per ogni coppia di frecce $f : a \to b, g : c \to d \in \mathbf{C}$ sia verificato $(f \otimes g); \gamma_{b,d} = \gamma_{a,c}; (g \otimes f)$, ovvero che il tensore sia commutativo a meno di isomorfismo naturale (γ). Si noti che se $\gamma_{a,b} : a \otimes b \to b \otimes a = id_{a\otimes b}$, allora $a \otimes b = b \otimes a$, cioè, si ritrova la definizione di categoria monoidale simmetrica stretta data precedentemente.

Definizione 3.4 (Funtore monoidale simmetrico)
Siano $\mathbf{C}$ *e* $\mathbf{C}'$ *due categorie monoidali simmetriche. Un* funtore monoidale simmetrico (stretto) $\mathcal{F} : \mathbf{C} \to \mathbf{C}'$ *è un funtore monoidale tale che*[5] $\mathcal{F}(\gamma_{a,b}) = \gamma'_{\mathcal{F}(a),\mathcal{F}(b)}$ *per ogni* $a, b \in O_{\mathbf{C}}$.

Chiamiamo **SMC** la categoria delle categorie monoidali simmetriche e rispettivi funtori.

Per catturare i processi deterministici è stata proposta una prima costruzione $\mathcal{P}(N)$ che: 1) mantiene la struttura di monoide commutativo delle marcature; 2) data una marcatura assume implicitamente che le marche nello stesso posto siano ordinate totalmente; 3) origina una struttura di categoria monoidale simmetrica (non stretta) sulle computazioni che permette di stabilire implicitamente quale marca è consumata da quale transizione (o preservata); 4) sfrutta le simmetrie della categoria per permutare esplicitamente le marche disponibili

[4] A questo proposito si noti che l'assioma $\gamma_{e,a} = id_a$ è derivabile dagli altri. Infatti $id_a = id_{e\otimes a} = \gamma_{e,a}; \gamma_{a,e} = \gamma_{e\otimes e,a}; \gamma_{a,e} = (id_e \otimes \gamma_{e,a}); (\gamma_{e,a} \otimes id_e); \gamma_{a,e} = \gamma_{e,a}; \gamma_{e,a}; \gamma_{a,e} = \gamma_{e,a}$.

[5] Si noti che nel caso di categorie monoidali simmetriche strette la condizione è automaticamente soddisfatta perché $\mathcal{F}$ è un funtore e quindi deve preservare le identità.

e influire sul modo in cui queste devono essere consumate. Le frecce della corrispondente categoria $\mathcal{P}(N)$ sono dette *processi concatenabili* e differiscono dai processi deterministici solo perché richiedono che, nella sorgente e nella destinazione, le marche nello stesso posto fossero ordinate totalmente. Tecnicamente, le simmetrie della categoria monoidale simmetrica $\mathcal{P}(N)$ soddisfano gli assiomi aggiuntivi: (i) $\gamma_{a,b} = id_{a\oplus b}$ per ogni posto $a \neq b$ di N; e (ii) $\gamma_{a,b}; t = t = t; \gamma_{c,d}$, se $t : a \otimes b \to c \otimes d$ e se $a \otimes b = b \otimes a$ e $c \otimes d = d \otimes c$. Il difetto principale della costruzione $\mathcal{P}(N)$ è di non essere estendibile a nessun funtore tra **Petri** e **SMC**: senza addentrarci nei dettagli tecnici, può succedere che dato un morfismo $f : N \to N'$ (tale da mandare posti diversi di N su marcature non disgiunte di N') sia impossibile definire il funtore $\mathcal{P}(f)$, perché esso dovrebbe mandare la stessa computazione concorrente di $\mathcal{P}(N)$ su frecce diverse di $\mathcal{P}(N')$.

Una seconda costruzione $\mathcal{Q}$ ha migliorato la situazione assumendo un ordine totale su tutte le marche (anche di posti diversi), ovvero considerando un marcature che sono stringhe invece di multi-insiemi. Le frecce di $\mathcal{Q}(N)$ sono dette *processi fortemente concatenabili* e risolvono in parte l'ambiguità che rende impossibile l'estensione funtoriale di $\mathcal{P}$. Infatti, la costruzione $\mathcal{Q}$ può essere estesa alle frecce di **Petri** in modo da definire uno *pseudo-funtore*, cioè un morfismo che preserva la struttura categoriale (identità e composizione) solo in maniera non stretta (attraverso un isomorfismo naturale diverso dall'identità). Più precisamente, $\mathcal{Q}$ preserva le identità in modo stretto, ma non la composizione.

L'aspetto interessante della costruzione $\mathcal{Q}(N)$ è quello di introdurre, per ogni transizione $t : u \to v \in N$, una famiglia di frecce: una freccia $t_{\overline{u},\overline{v}}$ per ogni possibile linearizzazione $\overline{u}$ e $\overline{v}$ della sorgente e destinazione di t. In questo modo, i morfismi devono fissare la corrispondenza tra tutte le possibili "implementazioni linearizzate" di t e la coerenza tra queste corrispondenze può essere assiomatizzata facilmente grazie alla simmetrie.

La costruzione che presentiamo sfrutta l'idea alla base di $\mathcal{Q}$, ma sposta la responsabilità di fissare l'implementazione dallo spazio computazionale al modello di partenza: invece di considerare reti P/T prendiamo come modello computazionale di riferimento le cosiddette *pre-reti*, dove il monoide dei posti è liberamente generato (non commutativo). Il passaggio da reti e pre-reti avviene scegliendo un'implementazione arbitraria per ogni transizione: ogni implementazione permette di costruire spazi di computazione equivalenti (cioè la semantica è indipendente dall'implementazione scelta) e riconducibili a quello definito da $\mathcal{Q}$. Il vantaggio immediato è quello di poter costruire lo spazio delle computazioni di una pre-rete attraverso una costruzione libera.

Definizione 3.5 (Pre-rete)
Una pre-rete è una quadrupla $R = (S, T, \zeta_0, \zeta_1)$, dove S è l'insieme dei posti*, T è l'insieme delle* transizioni *e le funzioni $\zeta_0, \zeta_1 : T \to S^{\otimes}$ assegnano, rispettivamente, stringhe di ingresso e di uscita ad ogni transizione.*

Per esempio, una pre-rete R' che implementa il sistema produttore consumatore è quella che ha lo stesso $\{p_1, p_2, b, c_1, c_2\}$ come insieme dei posti e transizioni $t'_1 : p_1 \to p_2 b$, $t'_2 : p_2 \to p_1$, $t'_3 : bc_1 \to c_2$ e $t'_4 : c_2 \to c_1$. Un'altra implementa-

zione R'' dello stesso sistema potrebbe considerare invece $t_1'' : p_1 \to bp_2$ invece di t_1' e $t_3'' : c_1 b \to c_2$ invece di t_3'.

I morfismi tra pre-reti sono definiti nel modo ovvio, richiedendo che la componente sugli stati sia un omomorfismo tra monoidi. Chiamiamo **PreReti** la categoria delle pre-reti e dei loro morfismi.

Intuitivamente, l'interpretazione computazionale di una pre-rete è quella di fissare un ben preciso ordine col quale consumare la marche dai posti di ingresso e produrle nei posti di uscita. La nozione di morfismo è quindi più stringente rispetto a quella delle reti P/T perché viene richiesto di preservare l'ordine col quale le marche vengono prelevate dai posti. In questo senso, è evidente che esiste un banale funtore "dimenticante" $\mathcal{A} : \mathbf{PreReti} \to \mathbf{Petri}$ che dimentica l'ordinamento scelto sui posti di ingresso e di uscita delle transizioni. Di seguito, data una stringa $w \in S^{\otimes}$ indichiamo con $\mu(w) \in S^{\oplus}$ il corrispondente multi-insieme tale che per ogni $a \in S$ il valore $\mu(w)(a)$ corrisponde al numero di occorrenze dell'elemento a nella stringa w.

Definizione 3.6
Il funtore $\mathcal{A} : \mathbf{PreReti} \to \mathbf{Petri}$ *è definito sugli oggetti come* $\mathcal{A}(S, T, \zeta_0, \zeta_1) = (S, T, \partial_0, \partial_1)$ *con* $\partial_i(t) = \mu(\zeta_i(t))$ *per ogni* $i \in [0, 1]$ *e* $t \in T$*; e sulle frecce* $f : R \to R'$ *in modo tale che*[6] $\mathcal{A}(f)(a) = \mu(f(a))$ *per ogni* $a \in S_R$ *e* $\mathcal{A}(f)(t) = f(t)$ *per ogni* $t \in T_R$.

Inoltre, se consideriamo la categoria **Reti** che comprende sia reti P/T che pre-reti come oggetti e i cui morfismi rispettano la struttura monoidale degli stati (permettendo quindi anche morfismi da pre-reti a reti), allora possiamo caratterizzare la forte corrispondenza tra i due modelli mediante un tipo particolare di aggiunzione, chiamata *riflessione*. Infatti le categorie **PreReti** e **Petri** sono entrambe sottocategorie piene di **Reti** e l'inclusione $\mathcal{I} : \mathbf{Petri} \to \mathbf{Reti}$ di **Petri** in **Reti** ha un aggiunto sinistro $\widehat{\mathcal{A}}$: $\mathbf{Reti} \to \mathbf{Petri}$ tale che $\mathcal{I}; \widehat{\mathcal{A}} = 1_{\mathbf{Petri}}$, ovvero la counit è l'isomorfismo identità (più precisamente $\widehat{\mathcal{A}}_{|\mathbf{PreReti}} = \mathcal{A}$ e $\widehat{\mathcal{A}}_{|\mathbf{Petri}} = 1_{\mathbf{Petri}}$). Sostanzialmente **Petri** è il quoziente della categoria **Reti** modulo la commutatività della struttura monoidale degli stati delle reti.

Dato che le pre-reti operano su stringhe, il candidato naturale per lo spazio algebrico delle computazioni è la sottocategoria piena di **SMC** delle categorie monoidali simmetriche $\mathbf{C}$ il cui monoide degli oggetti $O_{\mathbf{C}}$ è liberamente generato. Denotiamo questa categoria con $\mathbf{SMC}^{\otimes}$. La presenza delle simmetrie serve a garantire che le marche possono essere riarrangiate liberamente per consentire alle transizioni di prelevarle in un qualsiasi ordine, indipendentemente da come siano state generate.

Ancora una volta il funtore monoidale (aggiunto destro) $\mathcal{U}^{\otimes} : \mathbf{SMC}^{\otimes} \to \mathbf{PreReti}$ è definito banalmente sugli oggetti, ponendo $\mathcal{U}^{\otimes}((S^{\otimes}, F, \delta_0, \delta_1), \otimes, e, \gamma) = (S, T, \delta_0, \delta_1)$, ovvero scegliendo i "generatori" del monoide $S^{\otimes}$ come posti della pre-rete e prendendo le frecce della categoria come transizioni. Per quanto riguarda le frecce $F : \mathbf{C} \to \mathbf{C}' \in \mathbf{SMC}^{\otimes}$, è facile osservare che queste possono

[6] Ricordiamo che un omomorfismo tra monoidi liberamente generati è completamente definito fissando le immagini dei generatori del monoide.

$$\frac{w \in S_R^{\otimes}}{id_w \colon w \to w \in F_{\mathcal{Z}(R)}} \qquad \frac{t \in T_R}{t \colon \zeta_0(t) \to \zeta_1(t) \in F_{\mathcal{Z}(R)}} \qquad \frac{w, w' \in S_R^{\otimes}}{c_{w,w'} \colon ww' \to w'w \in F_{\mathcal{Z}(R)}}$$

$$\frac{\alpha \colon \bar{u} \to \bar{v},\ \beta \colon \bar{u}' \to \bar{v}' \in F_{\mathcal{Z}(R)}}{\alpha \otimes \beta \colon \bar{u}\bar{u}' \to \bar{v}\bar{v}' \in F_{\mathcal{Z}(R)}} \qquad \frac{\alpha \colon \bar{u} \to \bar{v},\ \beta \colon \bar{v} \to \bar{v}' \in F_{\mathcal{Z}(R)}}{\alpha ; \beta \colon \bar{u} \to \bar{v}' \in F_{\mathcal{Z}(R)}}$$

Figura 18: Regole di inferenza $F_{\mathcal{Z}(R)}$.

monoide:	$id_0 \otimes \alpha = \alpha = \alpha \otimes id_0 \quad (\alpha \otimes \beta) \otimes \delta = \alpha \otimes (\beta \otimes \delta)$
categoria:	$\alpha ; id_{\bar{v}} = \alpha = id_{\bar{u}} ; \alpha \quad (\alpha ; \beta) ; \delta = \alpha ; (\beta ; \delta)$
funtorialità:	$id_w \otimes id_{w'} = id_{ww'} \quad (\alpha ; \beta) \otimes (\alpha' ; \beta') = (\alpha \otimes \alpha') ; (\beta \otimes \beta')$
coerenza:	$c_{w,w'} ; c_{w',w} = id_{ww'} \quad c_{ww',w''} = (id_w \otimes c_{w',w''}) ; (c_{w,w''} \otimes id_{w'})$
naturalità:	$(\alpha \otimes \alpha') ; c_{\bar{v},\bar{v}'} = c_{\bar{u},\bar{u}'} ; (\alpha' \otimes \alpha)$

Figura 19: Assiomatizzazione equazionale di $\mathcal{Z}(R)$

essere viste direttamente come morfismi tra le pre-reti $\mathcal{U}^{\otimes}(\mathbf{C})$ e $\mathcal{U}^{\otimes}(\mathbf{C}')$. L'aggiunto sinistro $\mathcal{Z} : \mathbf{PreReti} \to \mathbf{SMC}^{\otimes}$ di $\mathcal{U}^{\otimes}$ è definito come segue. Per ogni pre-rete R, la categoria $\mathcal{Z}(R)$ è tale che $O_{\mathcal{Z}(R)} = S_R^{\otimes}$ e le cui frecce sono ottenute a partire dalle transizioni di R per chiusura rispetto alle regole di inferenza in Figura 18, modulo gli assiomi equazionali di Figura 19 (che devono essere soddisfatti per tutte le frecce $\alpha, \alpha', \beta, \beta', \delta$ che possono essere composte correttamente e per tutte le stringhe $w, w', w'', \bar{u}, \bar{v}, \bar{u}', \bar{v}'$). La unit $\eta_R : R \to \mathcal{U}^{\otimes}(\mathcal{Z}(R))$ dell'aggiunzione è definita come l'identità sulle marcature e manda ogni transizione $t \in R$ nella omonima transizione di $\mathcal{U}^{\otimes}(\mathcal{Z}(R))$.

La costruzione $\mathcal{Z}$ è quindi completamente soddisfacente dal punto di vista categoriale: non solo è funtoriale e quindi preserva i morfismi tra pre-reti, ma essendo un'aggiunto sinistro garantisce la composizionalità della semantica preservando le composizioni di pre-reti definite tramite colimiti.

Un secondo risultato che giustifica la costruzione $\mathcal{Z}$ è che essa caratterizza esattamente lo spazio dei processi fortemente concatenabili delle pre-reti, definiti in modo ovvio, modificando la definizione di rete di occorrenza deterministica in modo da considerare l'ordinamento degli ingressi e uscite delle transizioni (stringhe senza ripetizioni invece di insiemi).

Per esempio, assumendo che l'ordinamento totale su ingressi e uscite dei processi in Figura 17 sia quello posizionale (da sinistra verso destra) e che una marca di ingresso immobile segua implicitamente ogni marca di uscita prodotta dalla computazione, i corrispondenti processi fortemente concatenabili dell'implementazione R' possono essere scritti rispettivamente come due frecce distinte di $\mathcal{Z}(R')$:

$$\begin{array}{lcl} t'_1 \otimes id_{c_1} \otimes t'_4 \,;\, id_{p_2} \otimes t'_3 \otimes id_{c_1} & : & p_1c_1c_2 \to p_2c_2c_1 \\ t'_1 \otimes id_{c_1} \otimes t'_4 \,;\, id_{p_2 b} \otimes \gamma_{c_1,c_1} \,;\, id_{p_2} \otimes t'_3 \otimes id_{c_1} & : & p_1c_1c_2 \to p_2c_2c_1 \end{array}$$

Si noti come la simmetria γ_{c_1,c_1} viene usata per stabilire "quale" consumatore accede alla marca in b.

Un terzo risultato interessante stabilisce che prese due reti qualsiasi $R, R' \in$ **PreReti** tali che la rete $\mathcal{A}(R)$ è isomorfa a $\mathcal{A}(R')$, allora anche le categorie monoidali $\mathcal{Z}(R)$ e $\mathcal{Z}(R')$ sono isomorfe. Quindi tutte le diverse implementazioni della stessa rete originano spazi di computazione isomorfi. Per vederlo, consideriamo l'isomorfismo $\phi : \mathcal{A}(R) \to \mathcal{A}(R')$. Chiaramente, ϕ deve mandare posti in posti (non in marcature generiche) e quindi per ogni transizione $t \in R$ e $i \in [0,1]$ si ha che $\phi(\zeta_i(t))$ e $\zeta_i(\phi(t))$ sono stringhe uguali a meno di una qualche permutazione $\gamma(i,t) : \phi(\zeta_i(t)) \to \zeta_i(\phi(t))$. A questo punto definiamo un funtore monoidale $\Phi : \mathcal{Z}(R) \to \mathcal{Z}(R')$ ponendo (per ogni posto $a \in R$, stringhe $w, w' \in S_R^{\otimes}$ e transizione $t \in T_R$):

$$\Phi(a) = \phi(a) \qquad \Phi(c_{w,w'}) = c_{\phi(w),\phi(w')} \qquad \Phi(t) = \gamma(0,t); \phi(t); \gamma(1,t)^{-1}$$

Ripetendo il procedimento a partire dall'inversa $\phi^{-1} : \mathcal{A}(R') \to \mathcal{A}(R)$ di ϕ si costruisce un funtore monoidale $\Phi^{-1} : \mathcal{Z}(R') \to \mathcal{Z}(R)$ che è inverso di Φ.

Un quarto risultato stabilisce che lo spazio computazionale $\mathcal{Q}(N)$ dei processi fortemente concatenabili di una rete P/T N può essere ottenuto quozientando opportunamente lo spazio computazionale $\mathcal{Z}(R)$ dei processi fortemente concatenabili di una qualsiasi pre-rete R che implementa N (ovvero tale che $\mathcal{A}(R) = N$). Infatti, $\mathcal{Q}(\mathcal{A}(R))$ è isomorfa alla categoria ottenuta quozientando $\mathcal{Z}(R)$ rispetto all'assioma

$$t = s_0; t; s_1$$

per ogni transizione $t : \bar{u} \to \bar{v}$ e ogni coppia di simmetrie $s_0 : \bar{u} \to \bar{u}$ e $s_1 : \bar{v} \to \bar{v}$. Intuitivamente, in $\mathcal{Q}(\mathcal{A}(R))$ vengono introdotte tutte le possibili linearizzazioni $t_{\bar{u},\bar{v}} : \bar{u} \to \bar{v}$ per ogni transizione $t : u \to v \in \mathcal{A}(R)$ e poi, per renderle coerenti, si quozientano le linearizzazioni rispetto alle simmetrie in modo che $t_{\bar{u},\bar{v}} = s; t_{\bar{u}',\bar{v}'}; s'$ per ogni $s : \bar{u} \to \bar{u}'$ e $s' : \bar{v} \to \bar{v}'$. Invece in $\mathcal{Z}(R)$ si sceglie una sola, arbitraria linearizzazione $t_{\bar{u},\bar{v}}$ e si ottengono tutte le altre linearizzazioni per composizione con le simmetrie. Però la presenza di simmetrie $s : w \to w$ che sono automorfismi non banali finisce col generare più frecce di quelle presenti in $\mathcal{Q}(\mathcal{A}(R))$, da cui la necessità di quozientare. Per esempio, considerando una rete P/T N con transizione $t : 2a \to b$, esiste un'unica linearizzazione $t_{aa,b} : aa \to b$ e l'assiomatizzazione di $\mathcal{Q}$ impone $t_{aa,b} = \gamma_{a,a}; t_{aa,b}$, mentre presa una pre-rete R che implementa N usando $t' : aa \to b$ si ha $t' \neq \gamma_{a,a}; t'$ in $\mathcal{Z}(R)$.

3.1 Categoria "comma" e semantica per "srotolamento"

I processi che abbiamo visto descrivono computazioni concorrenti ma deterministiche. In letteratura sono state definite opportune varianti "non-deterministiche" dei processi in modo da catturare con un singolo oggetto matematico, chiamato *dominio algebrico primo (coerente e finitario)*, l'insieme di tutti gli eventi originabili da una rete relazionandoli in termini di *dipendenza causale*, *concorrenza* e *conflitto* (mutua esclusione). La prima costruzione di questo tipo è stata proposta nel lavoro seminale di Mogens Nielsen, Gordon Plotikin e Glynn Winskel [16]: partendo dalla categoria **Safe** di reti i cui posti possono contenere

al più una marca, si definisce una sotto-categoria **Occ** di reti di occorrenze non-deterministiche mediante un tipo particolare di aggiunzione detta co-riflessione (avente l'identità come unit); una seconda co-riflessione relaziona **Occ** con la categoria **PES** delle cosiddette *strutture di eventi prime* che è dimostrata essere equivalente alla categoria **Dom** dei domini algebrici primi, completando la catena di co-riflessioni da **Petri** a **Dom**. Successivamente l'approccio è stato esteso al caso di reti P/T, stabilendo in questo caso un'aggiunzione invece che una co-riflessione [14]. Questo tipo di semantica viene detta per "srotolamento" (*unfolding*) e presuppone la presenza di una marcatura iniziale u_0. Intuitivamente, le transizioni della rete di occorrenze non-deterministica definiscono gli eventi possibili. Due eventi dipendendono causalmente se sono connessi da una sequenza di archi di flusso e sono in conflitto diretto se hanno un posto di ingresso in comune. Il conflitto è *ereditario*, nel senso che se un evento e è in conflitto con e_1 e l'evento e_2 dipende causalmente da e_1 allora l'evento e è in conflitto anche con e_2. Due eventi (distinti) che non sono in conflitto e che non dipendono causalmente sono concorrenti. Senza avere lo spazio per addentrarci nei dettagli tecnici, nel caso delle reti P/T la semantica algebrica può essere riconciliata con la semantica per srotolamento costruendo un opportuno dominio ottenuto per completamento (per rappresentare computazioni infinite) della cosiddetta categoria "comma", che ha come oggetti i processi deterministici $p : u_0 \to u$ con sorgente la marcatura iniziale u_0 e come frecce tra due oggetti $p : u_0 \to u$ e $p' : u_0 \to u'$ quei processi $q : u \to u'$ tali che $p; q = p'$. Intuitivamente, la costruzione definisce un dominio i cui elementi sono processi ordinati per prefisso. I domini ottenuti coi due procedimenti (srotolamento o completamento della categoria comma) sono isomorfi.

Nel caso delle pre-reti, possiamo definire senza particolari difficoltà la categoria co-riflessiva **PreOcc** delle pre-reti di occorrenze (non-deterministiche) e un funtore da questa a **PES** in modo da ottenere un dominio isomorfo a quella che otterremmo per completamento della categoria "comma" basata su $\mathcal{Z}$. Tuttavia, pur se le pre-reti ammettono una semantica algebrica più elegante di quella delle reti P/T, per quanto riguarda la semantica per "srotolamento" rimane un problema aperto stabilire se il funtore che collega **PreOcc** a **PES** possa o meno essere esteso per definire un'aggiunzione tra le due categorie.

4 Conclusioni

L'obiettivo di questo scritto è quello di mostrare come l'eleganza formale di certe costruzioni matematiche possa diventare un importante strumento nel definire semantiche adeguate per sistemi concorrenti e relazionarle con quelle esistenti. Per esempio, abbiamo visto che è importante garantire la funtorialità delle costruzioni in modo da preservare la struttura dei modelli e che quando un funtore definisce un'aggiunzione siamo garantiti che la costruzione produca il modello semantico migliore in una classe di modelli prescelti. Da questo punto di vista, le costruzioni $\mathcal{C}$, $\mathcal{T}$, $\mathcal{M}$ e $\mathcal{Z}$ sono completamente soddisfacenti, mentre $\mathcal{P}$ e $\mathcal{Q}$ lo sono meno.

Nel caso di semantiche algebriche, ci sono ulteriori aspetti di interesse che è bene sottolineare. Per prima cosa le costruzioni sono tanto più interessanti quanto si possono ricollegare a concetti e spazi di computazione già noti, mostrando eventuali inefficienze e suggerendo come migliorarli. Questo è il caso dei processi commutativi e delle varie nozioni di processi concatenabili presenti in letteratura. Il secondo aspetto è che le costruzioni indicano le leggi algebriche fondamentali che regolano le computazioni concorrenti. Questo è il caso della funtorialità del prodotto tensore o degli assiomi di naturalità e coerenza delle simmetrie.

Un tema importante che abbiamo tralasciato per motivi di spazio è che le costruzioni potrebbero essere descritte a livello di *teorie di modelli* (cioè al livello del paradigma di specifica di certi sistemi) invece che al livello delle categorie di modelli. Per esempio, utilizzando opportune logiche equazionali possiamo facilmente definire le teorie algebriche per sistemi di transizione, monoidi, reti, categorie, categorie monoidali. Ad ogni teoria corrisponde una categoria di modelli, le cui frecce rispettano la struttura algebrica sottostante. Le teorie più complesse possono essere espresse in maniera compatta usando una costruzione detta *prodotto tensore fra teorie*: per esempio, la teoria delle categorie monoidali può essere definita come il prodotto tra le teorie delle categorie e dei monoidi. Le teorie possono essere messe in relazione tramite morfismi di teorie (da teorie con "meno struttura" verso teorie con "più struttura") che inducono dei funtori "dimenticanti" tra le corrispondenti categorie di modelli (dai modelli più strutturati a quelli più semplici). Questi funtori ammettono un aggiunto sinistro che ricostruisce, in modo libero, la minima struttura necessaria per arricchire i modelli della teoria "meno strutturata".

Un ultimo punto è quello della generalizzazione della metodologia per trattare sistemi più complessi. A partire dalla costruzione $\mathcal{T}$, l'approccio algebrico è stato esteso a sistemi di riscrittura di termini, dove gli stati sono definiti da teorie equazionali arbitrarie che la costruzione libera estende opportunamente per operare sulle riscritture concorrenti. I termini vengono rappresentati come frecce di una categoria cartesiana liberamente generata dalle segnatura e dagli assiomi della teoria. Pertanto, sia la struttura algebrica di stati e computazioni, sia la composizione di computazioni risultano espresse in termini di categorie cartesiane. Di conseguenza, i modelli di calcolo adeguati sono categorie doppie (corrispondenti al prodotto tensore tra la teoria delle categorie e se stessa). La riscrittura concorrente di termini è stata introdotta da Josè Meseguer con il nome di *rewriting logic* [12]. Ancora una volta la semantica corrispondente può essere definita come una aggiunzione e gode quindi della proprietà di conservare i colimiti. È stato dimostrato [10] che tale semantica corrisponde esattamente a quella introdotta in precedenza con una costruzione *ad hoc* da Gérard Boudol [3], basata sulla nozione di *permutation equivalence*. Tuttavia, la semantica concorrente così ottenuta non ha una naturale corrispondenza con domini algebrici primi e pertanto è discutibile se si tratti veramente di concorrenza. La corrispondenza con i domini algebrici primi può essere invece garantita utilizzando modelli basati su *sesqui-categorie* [5]. La metodologia vista può anche essere estesa al caso della *programmazione logica* [6].

Origini del materiale e riferimenti per approfondimenti La distinzione tra semantica "collettiva" e "individuale" è stata introdotta, a livello di terminologia, da Rob van Glabbeek e Gordon Plotkin [8]. Per la semantica "collettiva" i riferimenti principali sono il lavoro di Eike Best e Raymond Devillers dove vengono introdotti i processi commutativi [2] e il lavoro seminale di José Meseguer e Ugo Montanari [13] che introduce la metodologia presentata in questo scritto caratterizzando i processi commutativi mediante l'aggiunto $\mathcal{T}$. Per la semantica "individuale", il concetto di processo non-sequenziale è stato introdotto da Ursula Goltz e Wolfgang Reisig in [9], la costruzione $\mathcal{P}$ basata su permutazioni di marche è stata proposta originariamente da Pierpaolo Degano, José Meseguer e Ugo Montanari [7] assieme alla nozione di processo concatenabile, e successivamente assiomatizzata da Vladimiro Sassone [19]. José Meseguer, Ugo Montanari e Vladimiro Sassone hanno poi introdotto la nozione di processo fortemente concatenabile e hanno definito la costruzione pseudo-funtoriale $\mathcal{Q}$ [15, 20].

Le pre-reti sono state introdotte da Roberto Bruni, José Meseguer, Ugo Montanari e Vladimiro Sassone in [4] assieme all'aggiunto $\mathcal{Z}$. I processi fortemente concatenabili delle pre-reti sono stati formalizzati esplicitamente e relazionati con la semantica per "srotolamento" in [1].

Ringraziamenti. Ringraziamo caldamente i colleghi Paolo Baldan, Fabio Gadducci e Alberto Lluch-Lafuente per i loro preziosi commenti su una versione preliminare di questo saggio e i curatori del volume per averci invitato e per la paziente attesa del nostro contributo.

Riferimenti bibliografici

[1] P. Baldan, R. Bruni, and U. Montanari. Pre-nets, read arcs and unfolding: a functorial presentation. In M. Wirsing, D. Pattinson, and R. Hennicker, editors, *Proceedings of WADT 2002, 16th International Workshop on Algebraic Development Techniques*, volume 2755 of *LNCS*, pages 145–164. Springer Verlag, 2003.

[2] E. Best and R. Devillers. Sequential and concurrent behaviour in Petri net theory. *Theoret. Comput. Sci.*, 55(1):87–136, 1987.

[3] G. Boudol. *Computational semantics of term rewriting systems*, pages 169–236. Cambridge University Press, New York, NY, USA, 1986.

[4] R. Bruni, J. Meseguer, U. Montanari, and V. Sassone. Functorial models for Petri nets. *Inform. and Comput.*, 170(2):207–236, 2001.

[5] A. Corradini, F. Gadducci, and U. Montanari. Relating two categorial models of term rewriting. In J. Hsiang, editor, *RTA*, volume 914 of *Lecture Notes in Computer Science*, pages 225–240. Springer, 1995.

[6] A. Corradini and U. Montanari. An algebraic semantics for structured transition systems and its applications to logic programs. *Theor. Comput. Sci.*, 103(1):51–106, 1992.

[7] P. Degano, J. Meseguer, and U. Montanari. Axiomatizing the algebra of net computations and processes. *Acta Inform.*, 33(7):641–667, 1996.

[8] R. v. Glabbeek and G. Plotkin. Configuration structures. In D. Kozen, editor, *Proceedings of LICS'95, 10th Annual IEEE Symposium on Logics in Computer Science*, pages 199–209. IEEE Computer Society Press, 1995.

[9] U. Goltz and W. Reisig. The non-sequential behaviour of Petri nets. *Inform. and Comput.*, 57(2/3):125–147, 1983.

[10] C. Laneve and U. Montanari. Axiomatizing permutation equivalence. *Mathematical Structures in Computer Science*, 6(3):219–249, 1996.

[11] S. MacLane. *Categories for the Working Mathematician.* Graduate Texts in Mathematics 5. Springer Verlag, 2nd edition, 1998.

[12] J. Meseguer. Conditional rewriting logic as a unified model of concurrency. *Theor. Comput. Sci.*, 96(1):73–155, 1992.

[13] J. Meseguer and U. Montanari. Petri nets are monoids. *Inform. and Comput.*, 88(2):105–155, 1990.

[14] J. Meseguer, U. Montanari, and V. Sassone. On the semantics of Place/Transition Petri nets. *Math. Struct. in Comput. Sci.*, 7:359–397, 1997.

[15] J. Meseguer, U. Montanari, and V. Sassone. Representation theorems for Petri nets. In C. Freksa, M. Jantzen, and R. Valk, editors, *Foundations of Computer Science: Potential - Theory - Cognition, to Wilfried Brauer on the occasion of his sixtieth birthday*, volume 1337 of *Lect. Notes in Comput. Sci.*, pages 239–249. Springer Verlag, 1997.

[16] M. Nielsen, G. Plotkin, and G. Winskel. Petri Nets, Event Structures and Domains, Part 1. *Theoret. Comput. Sci.*, 13:85–108, 1981.

[17] C. Petri. *Kommunikation mit Automaten.* PhD thesis, Institut für Instrumentelle Mathematik, Bonn, 1962.

[18] W. Reisig. *Petri Nets: An Introduction.* EATCS Monographs on Theoretical Computer Science. Springer Verlag, 1985.

[19] V. Sassone. An axiomatization of the algebra of Petri net concatenable processes. *Theoret. Comput. Sci.*, 170(1-2):277–296, 1996.

[20] V. Sassone. An axiomatization of the category of Petri net computations. *Math. Struct. in Comput. Sci.*, 8(2):117–151, 1998.