Laboratorio di Reti e Sistemi Distribuiti

16: Introduzione agli Algoritmi Distribuiti

Roberto Marino, PhD[1]
roberto.marino@unime.it

[1]Dipartimento di Matematica, Informatica, Fisica e Scienze della Terra
Future Computing Research Laboratory
Università di Messina

Last Update: 13th May 2025

Indice

Definizione di algoritmo distribuito

"Gli Algoritmi Distribuiti sono algoritmi progettati per funzionare su molti processori interconessi fra di loro. Parti di un algoritmo distribuito operano simultaneamente e indipendentemente e ognuna ha a disposizione informazioni limitate. Gli algoritmi devono funzionare correttamente anche se i singoli processori e i canali di comunicazione operano a velocità diverse e sono soggetti a guasti." (Lynch, MIT)

Grafo di sistema

Una rete di agenti consiste nella collezione di "elementi di calcolo" localizzati ai nodi di un grafo orientato.
Tali elementi di calcolo si chiamano comunemente processori e si possono sia riferire ad un hardware che ad un software. I canali di comunicazione attraverso i quali i processori scambiano informazione sono rappresentati dagli archi del grafo.

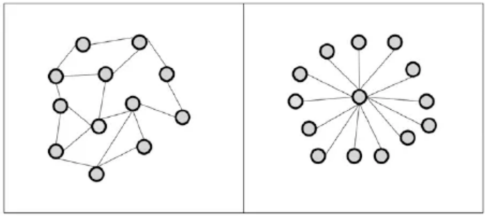


Figure: Sistema Distribuito vs Sistema Centralizzato

Modelli di Comunicazione

- `Shared Memory`: più processori/agenti scambiano informazione attraverso memoria condivisa (ad es. Thread)
- `Message Passing`: più processori/agenti comunicano tramite scambio di messaggi espliciti (socket, fifo, meccanismi pub/sub)

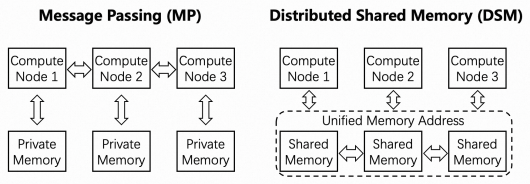


Figure: Shared Memory vs Message Passing

Tipi di parallelismo

- `Parallelismo dei Thread/Processi`: su processore multicore o multiprocessore
- `Sistema distribuito`: parallelismo su differenti macchine connesse via rete

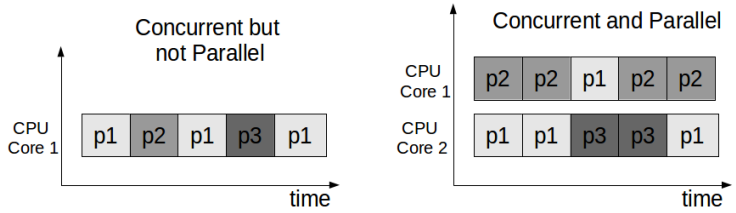


Figure: Differenza tra concorrenza e parallelismo

Sincronicità vs Asincronicità

- Un sistema distribuito è asincrono se ogni agente opera e scambia messaggio alla sua propria velocità, senza che vi sia alcuna relazione con la velocità degli agenti vicini.
- Un sistema distribuito è sincrono se tutti gli agenti calcolano, spediscono messaggi M e ricevono messaggi N sincronicamente cosicchè la configurazione C del sistema evolve in maniera sequenziale ed ordinata. Un sistema distribuito evolve in ROUND (es. da C_0 a C_1).

Esempio di evoluzione per sistema sincrono

$C_0, M_1, N_1, C_1, M_2, N_2, C_2$

Analisi di Complessità

- `Complessità temporale nel caso peggiore`: numero massimo di ROUND necessari affinchè tutti gli output vengano prodotti o tutti i processori si fermino (terminazione)
- `Complessità di comunicazione nel caso peggiore` : numero massimo di messaggi non-nulli trasmessi fino alla terminazione

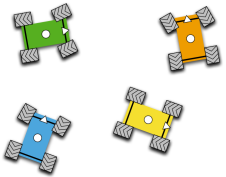
Distributed Control of Robotic Networks

A Mathematical Approach to Motion Coordination Algorithms

Chapter 1: An introduction to distributed algorithms

Francesco Bullo
Jorge Cortés
Sonia Martínez

May 20, 2009



PRINCETON UNIVERSITY PRESS
PRINCETON AND OXFORD

Network

Definition 1.38 (Network). The physical component of a *synchronous network* $\mathcal{S}$ is a digraph (I, E_{cmm}), where:

(i) $I = \{1, \ldots, n\}$ is called the *set of unique identifiers (UIDs)*; and

(ii) E_{cmm} is a set of directed edges over the vertices $\{1, \ldots, n\}$, called the communication links. •

Algoritmo Distribuito (DA)

Definition 1.39 (Distributed algorithm). A *distributed algorithm* $\mathcal{DA}$ for a network $\mathcal{S}$ consists of the sets

(i) $\mathbb{A}$, a set containing the `null` element, called the *communication alphabet*—elements of $\mathbb{A}$ are called *messages*;

(ii) $W^{[i]}$, $i \in I$, called the *processor state sets*; and

(iii) $W_0^{[i]} \subseteq W^{[i]}$, $i \in I$, sets of *allowable initial values*;

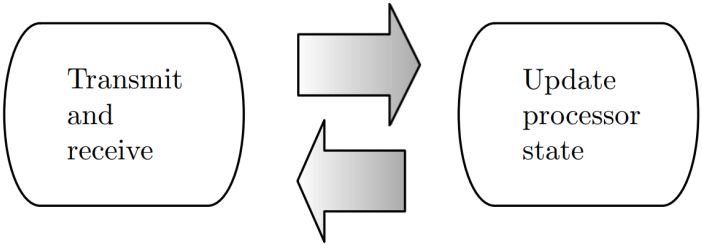
and of the maps

(i) $\text{msg}^{[i]} : W^{[i]} \times I \to \mathbb{A}$, $i \in I$, called *message-generation functions*; and

(ii) $\text{stf}^{[i]} : W^{[i]} \times \mathbb{A}^n \to W^{[i]}$, $i \in I$, called *state-transition functions*.

If $W^{[i]} = W$, $\text{msg}^{[i]} = \text{msg}$, and $\text{stf}^{[i]} = \text{stf}$ for all $i \in I$, then $\mathcal{DA}$ is said to be *uniform* and is described by a tuple $(\mathbb{A}, W, \{W_0^{[i]}\}_{i \in I}, \text{msg}, \text{stf})$. •

Modello di Comunicazione e Computazione



Ad ogni ROUND il primo step è la trasmissione del messaggio (`msg`) ed il secondo step è la computazione del nuovo stato interno dell'agente in funzione del proprio stato interno e dei messaggi ricevuti dai vicini (`stf`). I due step non possono essere invertiti altrimenti il formalismo non sarebbe consistente.

A^n è la tupla dei messaggi ricevuti dai vicini di ogni nodo (dagli in-neighbours)

Il problema della trasmissione broadcast

Problema

Data una rete (vedi definizione di Network 1.38), supponiamo che un agente v, chiamato sorgente, possieda un messaggio, chiamato token. L'obiettivo è trasmettere il token a tutti gli altri agenti nella rete.

Soluzione: algoritmo Flooding

L'algoritmo **Flooding** risolve il problema del broadcast e contemporaneamente **trova un albero BFS (Breadth-First Search Tree) con radice in** v (visita in ampiezza, albero dei cammini minimi).

Decrizione informale

La sorgente trasmette il token ai suoi vicini. In ogni ROUND, ogni nodo determina se ha ricevuto un messaggio non nullo da uno dei suoi vicini (in-neighbours). Quando viene ricevuto un messaggio non nullo, ossia quando il token è stato ricevuto, il nodo esegue due azioni: in primo luogo, il nodo **memorizza il token nella variabile data** (questo risolve il problema del Broadcast). In secondo luogo, il nodo **memorizza l'identità di uno dei vicini che hanno trasmesso il token nella variabile parent** (questo risolve il problema del calcolo dell'albero BFS). In particolare, se il messaggio viene ricevuto contemporaneamente da più vicini, il nodo nodo memorizza **la più piccola** tra le identità dei vicini trasmittenti. Nel successivo ciclo di comunicazione, il nodo trasmette il token ai suoi out-neighbors.

Flooding 1

Synchronous Network: $\mathcal{S} = (\{1, \ldots, n\}, E_{\text{cmm}})$

Distributed Algorithm: FLOODING

Alphabet: $\mathbb{A} = \{\alpha, \ldots, \omega\} \cup \texttt{null}$

Processor State: $w = (\texttt{parent}, \texttt{data}, \texttt{snd-flag})$, where

parent	$\in \{0, \ldots, n\}$,	initially:	$\texttt{parent}^{[1]} = 1$,
			$\texttt{parent}^{[j]} = 0$ for all $j \neq 1$
data	$\in \mathbb{A}$,	initially:	$\texttt{data}^{[1]} = \mu$,
			$\texttt{data}^{[j]} = \texttt{null}$ for all $j \neq 1$
snd-flag	$\in \{\texttt{false}, \texttt{true}\}$,	initially:	$\texttt{snd-flag}^{[1]} = \texttt{true}$,
			$\texttt{snd-flag}^{[j]} = \texttt{false}$ for $j \neq 1$

function $\text{msg}(w, i)$

1: **if** $(\texttt{parent} \neq i)$ AND $(\texttt{snd-flag} = \texttt{true})$ **then**
2: **return** data
3: **else**
4: **return** null

function $\text{stf}(w, y)$

Flooding 2

```
1:  case
2:    (data = null) AND (y contains only null messages):
        % The node has not yet received the token
3:        new-parent := null
4:        new-data := null
5:        new-snd-flag := false
6:    (data = null) AND (y contains a non-null message):
        % The node has just received the token
7:        new-parent := smallest UID among transmitting in-neighbors
8:        new-data := a non-null message
9:        new-snd-flag := true
10:   (data ≠ null):
        % If the node already has the token, then do not re-broadcast it
11:       new-parent := parent
12:       new-data := data
13:       new-snd-flag := false
14: return (new-parent, new-data, new-snd-flag)
```

Note importanti ai fini della comprensione

1. Si suppone che il vertice che possiede il token, cioè l'informazione da trasmettere, abbia UID pari ad 1.
2. `snd-flag` significa send flag
3. L'algoritmo è efficiente perchè evita ritrasmissioni, trasmette il messaggio solo una volta (vedi riga 10).

FLOODING: Analisi di Complessità

Complessità temporale

Data una rete S per un albero centrato in v la complessità è in $\Theta(radius(v, S))$, cioè la massima delle distanze tra v ed ogni altro vertice.

Complessità di comunicazione

Data una rete S la complessità è $\Theta(dim(E_{cmm}))$, cioè la cardinalità dell'insieme degli archi.