Pcd-Assigment03 - part 3 Report

Rock-Paper-Scissors in Go

Bedeschi Federica
0001175655 federica.bedeschi4@studio.unibo.it

Pracucci Filippo
0001183555 filippo.pracucci@studio.unibo.it

Anno accademico 2024-2025

Indice

Analisi

1.1 Descrizione del problema

Rock-Paper-Scissors è un gioco a due giocatori in cui lo scopo è sconfiggere l'avversario scegliendo un segno in grado di battere quello dell'altro, secondo le seguenti regole:

- il sasso spezza le forbici (vince il sasso);
- le forbici tagliano la carta (vincono le forbici);
- la carta avvolge il sasso (vince la carta).

Se i due giocatori scelgono lo stesso segno, si ha un pareggio.
Nella nostra versione del gioco partecipano due tipi di entità:

- **referee**: chiede le mosse ai giocatori e decreta il vincitore del turno, comunicandolo ai giocatori;
- **giocatore**: sceglie la propria mossa casualmente e la comunica al referee; una volta ricevuto il risultato del turno, lo mostra insieme al proprio punteggio, che incrementa in caso di vittoria.

1.2 Visione concorrente

Seguendo un approccio concorrente abbiamo rispettato i seguenti requisiti:

- i due giocatori devono scegliere il proprio segno concorrentemente;
- il referee deve coordinare i due giocatori scandendo la partita seguendo una gestione a turni;
- i giocatori ricevono il risultato del turno ed effettuano le azioni richieste in maniera concorrente.

Design e architettura

Abbiamo utilizzato il paradigma a **scambio di messaggi sincrono**, sfruttando il linguaggio **Go**. In particolare abbiamo suddiviso l'esecuzione in tre **goroutines** (2 per i giocatori ed 1 per il referee) seguendo la logica mostrata in Figura 2.1:

- referee: utilizza 4 canali, 2 per ricevere le mosse dei giocatori e 2 per inviare i risultati;
- giocatore: utilizza un canale per inviare la propria mossa al referee ed uno per ricevere da quest'ultimo il risultato; in caso di vittoria incrementa il proprio punteggio.

I canali creati sono quindi di 2 tipi:

- uno trasmette messaggi di tipo `Move`, che rappresenta le 3 possibili mosse dei giocatori (rock, paper e scissors);
- uno trasmette messaggi di tipo `Result`, che rappresenta i 3 possibili risultati di un turno (win, lose e draw).

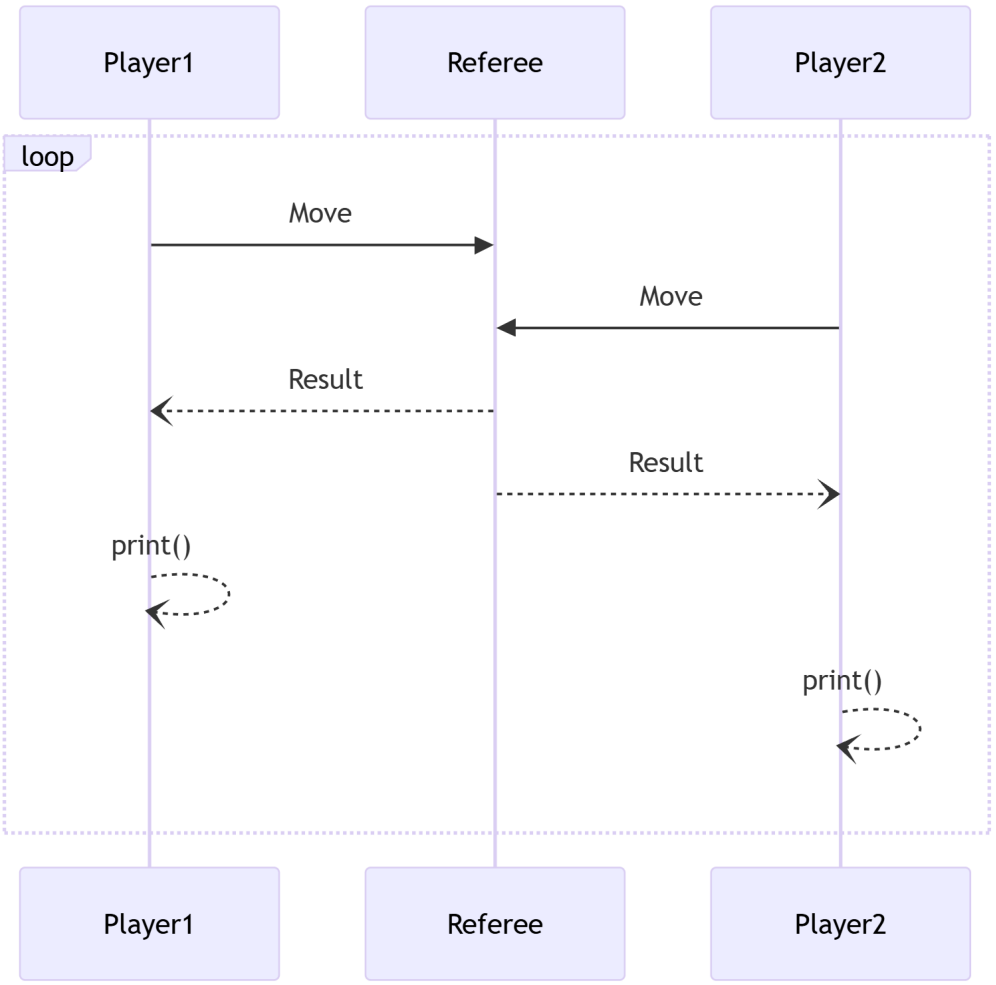

Figura 2.1: Diagramma di sequenza

Conclusioni

L'utilizzo di **Go** e soprattutto delle **goroutines** ha facilitato la gestione concorrente del sistema, infatti queste ultime sono molto leggere rispetto ai thread e di semplice utilizzo. Il paradigma a scambio di messaggi ci ha consentito di separare logicamente le diverse comunicazioni tra le entità, tramite la creazione di canali per il tipo specifico. La sincronizzazione è facilitata dal fatto che lo scambio di messaggi, ovvero `send` e `receive`, avviene in maniera sincrona.