Versione Definitiva — 10 gennaio 2018

# Progetto Unix 2017/18: descrizione

Enrico Bini Daniele Radicioni Claudio Schifanella

10 gennaio 2018

**Introduzione** Si intende realizzare la simulazione di una "società" di individui. In questa simulazione è presente una "popolazione" di processi di due tipi. Sulla base di certe regole, illustrate di seguito nel dettaglio, i processi generano dei figli che entrano a far parte della popolazione.

**Processi** Sono presenti processi di tre tipi:

1. "Gestore", che gestisce la simulazione e i vari eventi, e genera gli individui;
2. processi di tipo "A";
3. processi di tipo "B".

Ogni processo individuo è caratterizzato da:

- un *tipo* ("A" o "B");
- un *nome* codificato da una stringa di caratteri;
- un *genoma* codificato da un `unsigned long`.

**Attività del gestore** All'inizio, il gestore crea un numero `init_people`$\geq 2$ (per esempio 20) di individui. Per ogni individuo creato, vengono determinati casualmente:

- il suo tipo: "A" o "B";
- il suo nome: un carattere maiuscolo (da "A" a "Z");
- il suo genoma: un `unsigned long` casuale da `2` a `2+genes`.

La creazione degli individui avviene con `fork` e poi una `execve` del figlio.

Non appena creati, i processi "individuo" devono attendere la creazione di tutti gli altri, prima di iniziare il proprio ciclo di vita.

Quindi, il gestore aspetta la terminazione dei propri figli e, come vedremo, genera nuovi individui una volta che altri sono terminati. Difatti, la terminazione di una coppia di processi di tipo diverso determina la creazione di una nuova coppia di processi con caratteristiche "ereditate" dai due processi che si sono accoppiati e che sono poi terminati.

Ogni `birth_death` secondi il gestore:

1. termina un processo con un segnale (attenzione: la terminazione istantanea del processo selezionato con `SIGTERM` potrebbe determinare uno stato inconsistente delle strutture dati. Esempio: cosa succede se viene ucciso un processo "B" la cui proposta viene accettata "troppo tardi"? Cosa succede se viene ucciso un processo "A" appena dopo che un processo "B" lo ha contattato?). La scelta del processo da terminare è a scelta del progettista. Di seguito alcuni esempi:
   - casuale;
   - il processo più vecchio;
   - il processo con genoma più piccolo;
   - il processo con il nome più breve;
   - altro;
2. crea un nuovo processo con le stesse modalità della creazione iniziale e lo immette nella popolazione;
3. aggiorna l'utente sullo stato della simulazione.

Dopo `sim_time` dalla creazione iniziale, la simulazione termina. Il gestore informa tutti i processi presenti che devono terminare e fa in modo che non rimangano in stato zombie. Il gestore rilascia tutte le risorse eventualmente utilizzate. Dopo la terminazione, il processo gestore stampa alcune statistiche della simulazione che includono:

- il numero di processi di ciascun tipo vissuti durante tutta la simulazione;
- tutte le caratteristiche del processo con il nome più lungo (in seguito si vedrà che il nome di un processo potrà anche essere più lungo di un solo carattere);
- tutte le caratteristiche del processo con il genoma (che è un intero) più grande;
- altre informazioni ritenute utili.

Le quantità `init_people`, `genes`, `birth_death` e `sim_time` sono lette dall'utente a run time (da `stdin`, da un file testo di configurazione, da riga di comando, da variabili di ambiente, etc. a scelta).

**Vita dei processi "A"** Dopo la creazione dei processi "A" vengono pubblicate le loro informazioni (genoma, PID e/o altre informazioni ritenute utili). Tale pubblicazione avviene su apposita struttura dati accessibile anche ai processi "B". Dopo la pubblicazione delle proprie informazioni, i processi "A" rimangono in attesa di essere contattati da processi "B".

Quando un processo "B" contatta un processo "A", gli comunica le proprie informazioni (genoma, PID e/o altro). Il processo "A" decide se accoppiarsi con

il processo "B", sulla base di criteri che "rafforzino i propri discendenti" ovvero che diano ai propri discendenti un genoma alto (si leggano sotto le caratteristiche dei figli generati). Se il processo "B" ha un genoma multiplo del genoma del processo "A", il processo "A" acconsente sempre. Altrimenti, effettua la propria decisione per massimizzare il massimo comun divisore (MCD) fra il proprio genoma e quello del processo "B" che ha contattato. In ogni caso, il processo "A" informa il processo "B" della decisione effettuata.

- Se il processo "A" rifiuta l'offerta del processo "B"
  1. informa il processo "B" del rifuito
  2. si rimette in attesa di essere contattato da un nuovo processo "B".
- Se il processo "A" accetta l'offerta del processo "B"
  1. informa il processo "B" dell'accettazione
  2. comunica al processo gestore (con modalità a scelta del progettista) il PID del processo "B" accettato
  3. termina

**Vita degli individui "B"** Dopo la creazione, un processo "B" scorre le informazioni dei processi "A" presenti e contatta quello che ritiene gli possa garantire dei figli con un valore di genoma alto.

Una volta che ha contattato un processo "A" attende la sua risposta:

- se il processo "A" rifiuta, allora il processo "B" si mette alla ricerca di un nuovo processo "A"
- se il processo "A" accetta, allora il processo "B"
  1. comunica al processo gestore (con modalità a scelta del progettista) il PID del processo "A" che lo ha accettato
  2. termina.

**Caratteristiche dei processi generati da una coppia** In condizioni normali, un processo individuo termina quando ha trovato un processo di tipo diverso con cui accoppiarsi. In questo caso, il gestore è a conoscenza sia del PID del processo terminato (valore di ritorno della `wait()` o `waitpid()`) che di quello del processo con cui si intende accoppiare (comunicato dal processo prima di terminare).

A seguito della terminazione di una coppia di processi, il gestore provvede a immettere nella popolazione un'altra coppia di processi con le seguenti caratteristiche:

- tipo: casuale

- nome: i nomi dei due processi generati sono uguali al nome di ciascuno dei due genitori a cui viene attaccato in fondo (append) un nuovo carattere da "A" a "Z" casuale
- genoma: sia `x` il massimo comun divisore (MCD) fra i genomi dei due genitori (noti al gestore attraverso i loro PID). Il genoma del figlio è un numero casuale compreso fra `x` e `x+genes`

Si ricorda che a seguito di un accoppiamento entrambi processi "genitore" terminano e quindi vengono creati dal gestore due figli.

**Commenti** Seguono ulteriori commenti e charificazioni

**Diversità** All'atto della creazione e a seguito della creazione di nuovi processi, il gestore deve assicurare che la popolazione non sia costituita interamente da individui dello stesso tipo.

**Adattamento** Si richiede che il comportamento dei processi di tipo "A" cambi al variare dell'evoluzione della simulazione. Quando un processo "A" non è stato in grado di generare figli con alcun processo "B" presente (eventualità che si può scoprire andando a tenere traccia dei processi "B" che hanno richiesto un contatto), il processo "A" deve necessariamente abbassare il proprio target in modo da permettere una più probabile generazione di figli.

**Compilazione e grado di finitura del software** Le funzioni fondamentali devono essere *documentate con chiarezza* (quelle *non* commentate dovrebbero essere auto-esplicative): occorre specificare che input prendono, qual la loro funzione, se ci sono delle precondizioni, forme di dipendenza, di che tipo, *etc.*.

È necessario costruire un Makefile per la compilazione dei file costituenti il progetto; i file devono essere compilati con le opzioni `-Wall -pedantic`; deve essere possibile compilare i moduli principali insieme (opzione di default) o separatamente.

**Documentazione** L'elaborato finale deve contenere anche una documentazione in cui si illustrano i principali problemi di comunicazione fra processi presenti nell'esercitazione e istruzioni su come eseguire il codice della presente esercitazione.