FARM2

1 Descrizione

Farm2 calcola un risultato numerico in base al contenuto di file regolari (v. ProgettoSOL_23-24.pdf).

1.1 Compilazione

```
$ make farm
```

1.2 Esecuzione

```
$ ./farm [files]
```

1.2.1 Opzioni

-n number of threads - numero di thread inizialmente creati.

-d directory - directory da cui prelevare i file regolari.

-q queue length - dimensione massima della coda concorrente condivisa.

-t delay (ms) - ritardo in millisecondi fra un inserimento di un file in coda e l'altro.

-h help - stampa le possibili opzioni da passare al programma.

1.3 Esecuzione test

```
$ make test
```

2 Struttura del codice

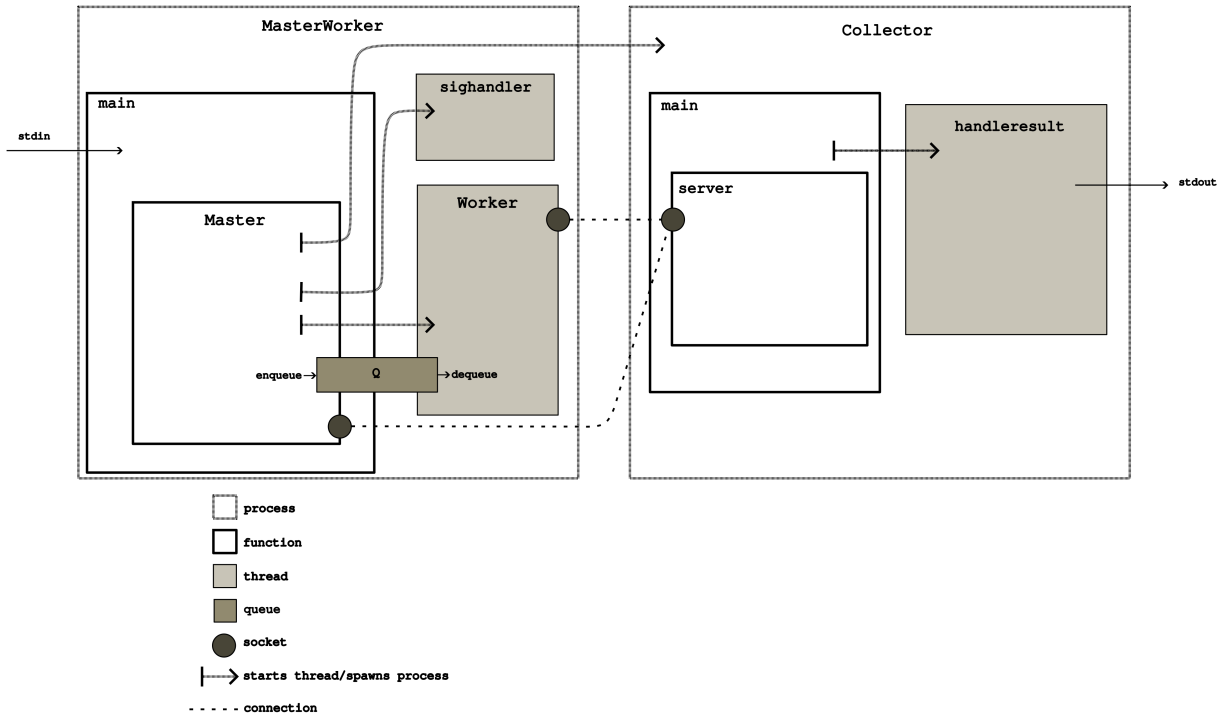

Figura 1: schema del codice

aux.c contiene funzioni, strutture e variabili usate dal programma. In particolare le funzioni is_reg e dir_rec vengono usate dal processo MasterWorker per cercare i file regolari fra quelli in input. La struttura res e' usata dai Worker e da Collector per scambiarsi i dati relativi ai risultati.

queue.c strumenti per la gestione della coda concorrente condivisa. La coda e' implementata come una linked list. Vengono mantenuti due puntatori: Q.first, che punta all'inizio della coda e viene modificato in fase di prelievo di un elemento (dequeue); e Q.last, che punta alla fine della coda e viene modificato in fase di aggiunta di un elemento (enqueue).

main.c riceve i dati forniti dall'utente, li passa a Master. Crea il processo Collector. Al termine della sua esecuzione, si connette a Collector tramite la funzione _collector_done, ed i nvia a Collector la stringa "DONE". Rimuove poi il file socket.

master.c controlla che i file passati da main siano regolari e li aggiunge alla coda condivisa (Q). Crea n thread Worker e il processo Collector. Crea il thread sighandler, che si occupa della gestione dei segnali. Scrive il numero di thread presenti all'uscita nel file worker_file.

worker.c preleva un file name dalla coda condivisa con Master, calcola il risultato, lo invia a Collector tramite connessione socket AF_UNIX.

collector.c chiama la funzione server, che apre una connessione socket AF_UNIX. si mette in attesa dei risultati, fino alla ricezione della stringa "DONE". I risultati vengono salvati in un vettore di res. il thread handleresult si occupa di ordinare e stampare i risultati contenuti nel vettore di res. Questa operazione viene eseguita dopo ogni secondo e prima della terminazione di Collector.

3 Sincronizzazione dei thread

La gestione della sincronizzazione fra i thread Worker e il thread Master e' interamente a carico di queue.c, ed e' gestita tramite condition variables e mutex.