Laboratorio di Programmazione a.a. 2024-25

Problema 6

18 e 19 Novembre 2024

Descrizione

Il comando `diff`, disponibile nelle shell dei sistemi operativi basati su Linux/Unix[1], consente di confrontare il contenuto di due file per rilevarne le differenze. È comunemente utilizzato per risalire alle modifiche apportate in versioni successive di documenti testuali, in particolare di programmi. Per esempio, nel caso dei due file associati a questo problema, contenenti due versioni diverse di un programma in Scheme per il calcolo della *sottosequenza comune più lunga* (LCS), l'esecuzione "da terminale" del comando di shell

```
diff lcs_v1.rkt lcs_v2.rkt
```

produce il seguente output (*):

```
11c11
<             (string (string-ref u 0)) (lcs (substring u 1) (substring v 1))))
---
>             (substring u 0 1) (lcs (substring u 1) (substring v 1))))
15a16,17
> ;;  Stringa piu' lunga
>
19,21c21,24
<       (if (< m n)
<           v
<           u))
---
>       (cond ((< m n) v)
>             ((> m n) u)
>             ((= (random 2) 0) v)
>             (else u)))
```

La strategia per identificare le differenze rielabora opportunamente (e perfeziona) la soluzione del problema della sottosequenza comune più lunga, restituendoci una concatenazione di tali differenze, che in genere sono abbastanza circoscritte e più informative, anziché la parte comune. In questo contesto, i testi oggetto del confronto sono visti come sequenze di righe, dove ciascuna riga è considerata elemento atomico indivisibile e svolge pertanto lo stesso ruolo di un unico simbolo in relazione alla procedura discussa a lezione. Di conseguenza, la "misura" delle differenze, misura che vogliamo rendere minima, è determinata dal *numero* di righe che non sono poste in corrispondenza nei due testi, indipendentemente dalla *lunghezza* delle righe in termini dei caratteri di cui sono costituite.

Nell'output del comando le differenze sono descritte da indici numerici con interposta una lettera, seguiti dalle righe di testo coinvolte. Gli indici rappresentano le posizioni nel testo di righe singole o di gruppi di righe contigue — nel qual caso sono riportate le posizioni estreme separate da una virgola. L'indice o la coppia di indici di posizione a sinistra della lettera si riferiscono al primo testo, la corrispondente informazione a destra della lettera riguarda invece il secondo testo. Le lettere denotano tre "azioni" per passare dal primo testo al secondo: `d` (delete) = cancellazione di una o più righe; `a` (add) = inserimento di una o più righe; `c` (change) = sostituzione di una o più righe con righe diverse. Le righe introdotte da < (sinistra) provengono dal primo testo, mentre quelle precedute da > (destra) dal secondo. Nel caso di sostituzione (`c`), i gruppi di righe riportate dai due testi sono separati da "`---`". Si può osservare che la numerazione degli indici delle righe parte da 1 (prima riga del testo), e non da 0 come in altri casi che potrebbero essere asimilati; nel caso di cancellazione (`d`) da oppure inserimento (`a`) in un testo, l'indice relativo all'altro testo si riferisce alla posizione dell'ultima riga la cui elaborazione è stata completata trovando una corrispondenza o assumendone la cancellazione.

L'obiettivo di questa esercitazione è realizzare in Scheme un modello della funzionalità del comando `diff`. A tal fine, il contenuto di un ipotetico file di testo può essere rappresentato da una lista di stringhe, dove ciascuna stringa corrisponde a una riga di testo (e viceversa), rispettandone l'ordine. Inoltre, il risultato dell'elaborazione (l'output) sarà rappresentato da una lista di differenze, rappresentate a loro volta da liste di quattro elementi:

- un indice di posizione relativa al primo testo,
- una lettera per denotare l'azione,
- un indice di posizione relativa al secondo testo,
- la riga coinvolta.

Per semplicità, considera solo le due azioni di inserimento (`a`) e cancellazione (`d`), visto che la sostituzione (`c`) è sostanzialmente descritta da sequenze di cancellazioni da un testo e inserimenti nell'altro.

Definisci quindi un programma per realizzare in Scheme la procedura `diff` che, dati due "testi" (cioè due liste di stringhe), restituisce la lista delle differenze in accordo con le specifiche riportate sopra. In particolare, l'applicazione di `diff` codificata nel file `input.txt`, associato a questo problema, restituirà una lista di differenze organizzata come segue, a meno di possibili piccoli cambiamenti dell'ordine di alcune azioni (**):

```
(list
 (list 11 'd 10 "            (string (string-ref u 0)) (lcs (substring u 1) (substring v 1))))")
 (list 11 'a 11 "            (substring u 0 1) (lcs (substring u 1) (substring v 1))))")
 (list 15 'a 16 ";;  Stringa piu' lunga")
 (list 15 'a 17 "")
 (list 19 'd 20 "       (if (< m n)")
 (list 20 'd 20 "           v")
 (list 21 'd 20 "           u))")
 (list 21 'a 21 "       (cond ((< m n) v)")
 (list 21 'a 22 "             ((> m n) u)")
 (list 21 'a 23 "             ((= (random 2) 0) v)")
 (list 21 'a 24 "             (else u)))"))
```

Le relazioni fra l'output (*) del comando di shell `diff` e la lista di diffrenze (**) restituita dall'omonima procedura Scheme possono essere illustrate più chiaramente attraverso l'uso di colori per enfatizzare le parti corrispondenti:

```
11 'd                11c11
                     < (string (string-ref u 0)) (lcs (substring u 1) (substring v 1))))
                     ---
    'a 11            > (substring u 0 1) (lcs (substring u 1) (substring v 1))))
                     15a16,17
    'a 16            > ;;  Stringa piu' lunga
    'a 17            >
                     19,21c21,24
19 'd                <       (if (< m n)
20 'd                <           v
21 'd                <           u))
                     ---
    'a 21            >       (cond ((< m n) v)
    'a 22            >             ((> m n) u)
    'a 23            >             ((= (random 2) 0) v)
    'a 24            >             (else u)))
```

```
(list
 (list 11 'd 10 "            (string (string-ref u 0)) (lcs (substring u 1) (substring v 1))))")
 (list 11 'a 11 "            (substring u 0 1) (lcs (substring u 1) (substring v 1))))")
 (list 15 'a 16 ";;  Stringa piu' lunga")
 (list 15 'a 17 "")
 (list 19 'd 20 "       (if (< m n)")
 (list 20 'd 20 "           v")
 (list 21 'd 20 "           u))")
 (list 21 'a 21 "       (cond ((< m n) v)")
 (list 21 'a 22 "             ((> m n) u)")
 (list 21 'a 23 "             ((= (random 2) 0) v)")
 (list 21 'a 24 "             (else u)))"))
```

Per progettare la soluzione puoi ispirarti alla struttura dell'esempio `xlcs` discusso a lezione (sorgente associato al problema). Sperimentando l'applicazione della procedura `diff` a piccoli testi, come quelli codificati nel file `input.txt`, ti accorgerai che la soluzione adottata, ancorché logicamente corretta, ha prestazioni molto scadenti in termini di tempi di calcolo. Le prestazioni di questo e di altri programmi analoghi possono comunque essere migliorate significativamente con delle tecniche di trasformazione che saranno oggetto di studio nella seconda parte del corso.

---

1 "In computing, the utility `diff` is a data comparison tool that computes and displays the differences between the contents of files. [...] `diff` is line-oriented rather than character-oriented, [... and] it tries to determine the smallest set of deletions and insertions to create one file from the other" (fonte: *wikipedia.org*, Novembre 2024). L'utility `diff` è stata realizzata all'inizio degli anni 1970, ad opera di D. McIlroy e J. Hunt, per il sistema operativo *Unix* in corso di sviluppo presso i Bell Labs a Murray Hill (NJ, USA). L'algoritmo, pubblicato in un articolo del 1976, si basa su perfezionamenti dello schema di soluzione LCS.