Fondamenti di Informatica, A.A. 2013-2014

04/09/2014

Prova Pratica

Si scriva una funzione Matlab/Octave che dato in ingresso un testo rappresentato con un vettore contenente caratteri, individui le parole (stringhe di caratteri alfabetici) e ne conti il numero e la lunghezza media.

Si ricorda che per verificare se un carattere è una lettera dell'alfabeto è disponibile sia in Matlab che in Octave la funzione `isletter`.

Bonus: individuare le parole ripetute, e determinare la loro frequenza.

Soluzione

Per quanto riguarda la soluzione dell'esercizio base, occorre per prima cosa stabilire cosa intendiamo per "parola". Al riguardo adottiamo la definizione:

> Una parola è una sequenza di uno o più caratteri alfabetici consecutivi.

Da questa definizione possiamo ricavare un criterio operativo:

> Per contare il numero di parole in una stringa, possiamo contare quante volte un carattere alfabetico è seguito immediatamente da uno non alfabetico.

Il criterio appena detto si realizza facilmente usando la funzione `isletter` per trasformare un array di caratteri in un array di valori logici; questo è utile anche per contare il numero totale di caratteri alfabetici, totale che poi diviso per il numero di parole ci darà la lunghezza media. L'unico punto che occorre considerare è che il metodo proposto non considera correttamente una parola nel caso in cui l'array termini con un carattere alfabetico; pertanto questo caso va contato a parte

```
function [count,avg] = wordcount(wtext)

 letters=isletter(wtext);
 n=sum(letters);
 count=0;
 for i=2:length(letters)
     if (letters(i-1)&&(~(letters(i))))
        count = count + 1;
     end
 end
 count = count + letters(end);
 if (nargout>1)
    avg = n/count;
 end
end
```

Lo stesso risultato si può ottenere usando esclusivamente gli operatori su array:

```
function [count,avg] = wordcountv(wtext)

 letters=isletter(wtext);
 n=sum(letters);
 count=sum(letters(1:end-1)&(~letters(2:end)))+letters(end);
 if (nargout>1)
    avg = n/count;
 end

end
```

Per contare la frequenza delle parole è invece necessario avere uno schema più complesso, in cui bisogna tenere memoria delle parole trovate e capire se erano già presenti nell'input oppure no. Questo può essere realizzato con un array di strutture, in cui ciascuna struttra contiene tre campi, la parola, il numero di lettere, ed il numero di volte con cui la parola occorre nel testo dato. Il programma proposto utilizza una semplice ricerca lineare per verificare se la parola sia nuova ovvero già presente nei dati; ovviamente la ricerca lineare non sarebbe efficiente per grandi numeri, ed occorrerebbero schemi nettamente più sofisticati.

```
function [count,avg,freq] = wordfrq (wtext)

  state=0; words=[];
  for i=1:length(wtext)
    switch (state)
      % State = 0: we were out of a word
      case 0
        if (isletter(wtext(i)))
          state=1;
          k=i;
        end
      % State = 1: we were inside a word
      case 1
        if (~isletter(wtext(i)))
          % Current word is in pos. k:i-1
          wrd = makewrec(i-k,wtext(k:i-1));
          % Is it a new word?
          % Count the occurrences
          pos=linsearch(words,wrd);
          if (pos <= length(words))
             words(pos).cnt= words(pos).cnt+1;
          else
             words=[words wrd];
          end
          state=0;
        end
    end
  end
  if (state==1)
```

```
      % Current word is in pos. k:end
      i=length(wtext);
      wrd = makewrec(i-k+1,wtext(k:i));
      pos=linsearch(words,wrd);
      if (pos <= length(words))
        words(pos).cnt= words(pos).cnt+1;
      else
        words=[words wrd];
      end
    end
    count=sum([words.cnt]);
    if (nargout>1)
      avg = sum([words.len])/count;
    end
    if (nargout>2)
      freq=[words.cnt]/count;
    end
end

function pos=linsearch(vect,wrec)
    for k=1:length(vect)
      if (strcmp(wrec.word,vect(k).word))
            pos=k;
            return
        end
    end
    pos=length(vect)+1;
end

function wrec=makewrec(len,wrd)
      wrec.word=wrd;
      wrec.len=len;
      wrec.cnt=1;
end
```