Subject: Re: Accelerating "for "loops
Posted by jeanh on Thu, 14 Oct 2010 13:59:40 GMT
View Forum Message <> Reply to Message

On 14/10/2010 9:31 AM, Regine wrote:
> Hello,
> I need to know if there's a way to make this code go faster as I am
> using very large arrays.
> the code is :
> nTab=15296820
> n_Tag1=5925470
> n_Tag2= 2478581
> n_Tag3=6892766
> FinalTable=make_array(nTab)
>
> ;Indice_tag1, Indice_tag2 and Indice_Tag3 are arrays of size n_Tag1,
> n_Tag2 and n_Tag3 respectively
>
> for i=0L,nTab-1 do begin
> for j=0L,n_Tag1-1 do begin
> for k=0L,n_tag2-1 do begin
> for l=0L,n_tag3-1 do begin
> if i eq Indice_Tag1[j] then begin
> FinalTable[i]=Tag1[j]
> Endif
> if i eq Indice_tag2[k] then begin
> FinalTable[i]=tag2[k]
> endif
> if i eq Indice_tag3[l] then begin
> FinalTable[i]=tag3[l]
> endif
> endfor
> endfor
> endfor
> endfor
>
> END
>
> Thank you in advance for any help that you can provide,
>
> Cheers,

Hi,

Here are three ways, faster and faster....
First, from your code, you are checking tag1[j] k*l times too often,
tag2[k] l times too often... Also, in your code, only the last occurence

of i=indiceTag[j] (and for k and l) are considered... is this what you
want? if the first occurence is ok, then break the loop when found!)

```
for i=0L,nTab-1 do begin
   for j=0L,n_Tag1-1 do begin

     if i eq Indice_Tag1[j] then $
       FinalTable[i]=Tag1[j]

     for k=0L,n_tag2-1 do begin

       if i eq Indice_tag2[k] then $
 FinalTable[i]=tag2[k]

        for l=0L,n_tag3-1 do begin
          if i eq Indice_tag3[l] then $
            FinalTable[i]=tag3[l]

       endfor
     endfor
    endfor
endfor
```

---

now, this is inefficient... you can easily remove loops j,k and l

```
for i=0L,nTab-1 do begin

    J_idx = where(indice_tag1 eq i, countJ)
    if countJ gt 0 then finalTable[i] = tag1[countJ-1]
    J_idx = 0B

    K_idx = where(indice_tag2 eq i, countK)
    if countK gt 0 then finalTable[i] = tag2[countK-1]
    K_idx = 0B

     L_idx = where(indice_tag3 eq i, countL)
    if countL gt 0 then finalTable[i] = tag3[countL-1]
    L_idx = 0B

endfor
```

---

and the best option (if you have enough memory), is to use histograms!
I don't have IDL in front of me so I will leave the implementation

details to you

basically:

histo = histogram(indice_tag1, min=0, binSize=0, reverse_indices = ri)
then look when there is one or more match (histo>0) and use the last
indice provided in ri for the bin.

Do the same with tag2 etc and assign the values to your final array

Jean
