Subject: Re: Improving a piece of code with arrays and for-loops
Posted by Vince Hradil on Thu, 08 Feb 2007 17:53:40 GMT
View Forum Message <> Reply to Message

On Feb 8, 7:23 am, "Halfdan" <halfd...@gmail.com> wrote:
> Hello
>
> I have been looking at my same piece of IDL-code for quite a while now
> and I have yet not found any good method to improve it. I want to
> improve the speed and rid the code of the nested for-loops. Maybe
> there is someone here who has good ideas and is willing to point me in
> the right direction?
>
> The problematic code is below. It is a part of a method to estimate
> wind gusts in output from an atmospheric model. The i and j
> dimensions are the x- and y-locations of the model-points in the
> horizontal and s are the model level heights in the vertical (starting
> from the model top and growing towards the surface).
>
> The code works in the vertical, starting from the surface (largest
> value of s) and works upwards (towards smaller s) to where the value
> of the variable tke is less than tkelvl or tke_diff is less than a
> very small number. The code has to do the following three things:
>
> 1. Choose the greatest value of wsp (windspeed) where the value of
> int_diff exceeds 0.
> 2. Choose the greatest value of wsp where the value of int_diffver
> exceeds 0.
> 3. Choose the greatest value of wsp.
>
> This has to be repeated for every grid-point in the horizontal (I have
> to assume that I have very little a priori knowledge of the behaviour
> of any of my variables at any gridpoint and model height).
>
> Any ideas on improving this?
>
> Thanks in advance,
> Halfdan
>
> ps. The problematic code:
>
> for i=1,ni-2 do begin
> for j=1, nj-2 do begin
> s = ns-1
> REPEAT BEGIN
> if int_diff(i,j,s) GE 0. AND wsp(i,j,s) GT fgtmp(i,j,
> 0) then $
> fgtmp(i,j,0) = wsp(i,j,s)

```
>                 if wsp(i,j,s) GT fgtmp(i,j,1) then $
>                   fgtmp(i,j,1) = wsp(i,j,s)
>                 if int_diffver(i,j,s) GE 0. AND wsp(i,j,s) GT
>  fgtmp(i,j,2) then $
>                   fgtmp(i,j,2) = wsp(i,j,s)
>                 s=s-1
>               ENDREP UNTIL tke(i,j,s) LT tkelvl(i,j) OR
>  tke_diff(i,j,s) LT eps
>           endfor
>       endfor
```

First thing that comes to mind is the invert the loop order. Granted
I've spent about 30 s looking at this, though.

Subject: Re: Improving a piece of code with arrays and for-loops
Posted by Vince Hradil on Thu, 08 Feb 2007 17:59:11 GMT
View Forum Message <> Reply to Message

On Feb 8, 11:53 am, "hradilv" <hrad...@yahoo.com> wrote:
> On Feb 8, 7:23 am, "Halfdan" <halfd...@gmail.com> wrote:
>
>
>
>> Hello
>
>> I have been looking at my same piece of IDL-code for quite a while now
>> and I have yet not found any good method to improve it. I want to
>> improve the speed and rid the code of the nested for-loops. Maybe
>> there is someone here who has good ideas and is willing to point me in
>> the right direction?
>
>> The problematic code is below. It is a part of a method to estimate
>> wind gusts in output from an atmospheric model. The i and j
>> dimensions are the x- and y-locations of the model-points in the
>> horizontal and s are the model level heights in the vertical (starting
>> from the model top and growing towards the surface).
>
>> The code works in the vertical, starting from the surface (largest
>> value of s) and works upwards (towards smaller s) to where the value
>> of the variable tke is less than tkelvl or tke_diff is less than a
>> very small number. The code has to do the following three things:
>
>> 1. Choose the greatest value of wsp (windspeed) where the value of
>> int_diff exceeds 0.
>> 2. Choose the greatest value of wsp where the value of int_diffver
>> exceeds 0.
>> 3. Choose the greatest value of wsp.

>
>> This has to be repeated for every grid-point in the horizontal (I have
>> to assume that I have very little a priori knowledge of the behaviour
>> of any of my variables at any gridpoint and model height).
>
>> Any ideas on improving this?
>
>> Thanks in advance,
>> Halfdan
>
>> ps. The problematic code:
>
>>   for i=1,ni-2 do begin
>>   for j=1, nj-2 do begin
>>   s = ns-1
>>   REPEAT BEGIN
>>   if int_diff(i,j,s) GE 0. AND wsp(i,j,s) GT fgtmp(i,j,
>> 0) then $
>>   fgtmp(i,j,0) = wsp(i,j,s)
>>   if wsp(i,j,s) GT fgtmp(i,j,1) then $
>>   fgtmp(i,j,1) = wsp(i,j,s)
>>   if int_diffver(i,j,s) GE 0. AND wsp(i,j,s) GT
>> fgtmp(i,j,2) then $
>>   fgtmp(i,j,2) = wsp(i,j,s)
>>   s=s-1
>>   ENDREP UNTIL tke(i,j,s) LT tkelvl(i,j) OR
>> tke_diff(i,j,s) LT eps
>>   endfor
>>   endfor
>
> First thing that comes to mind is the invert the loop order. Granted
> I've spent about 30 s looking at this, though.

Another thing that comes to mind is trying to make a "mask" of the of
the layers and applying that mask:

instead of if int_diff[i,j,s] gt 0, use id_mask = int_diff[*,*,s] > 0
then idx = where(id_mask gt 0, count) - then change only those idx
points.

Subject: Re: Improving a piece of code with arrays and for-loops
Posted by Halfdan on Fri, 09 Feb 2007 12:01:42 GMT
View Forum Message <> Reply to Message

Thank you for the advice.

I always manage to mix up the order of the indices in

the for loops. No matter how often I write them.

I will try the "mask"-method. I had thought of something in this direction but never managed to get the method and the code right

ha
