
Subject: Re: dynamic array workaround?

Posted by [Robert Moss](#) on Mon, 15 Aug 2005 02:09:27 GMT

[View Forum Message](#) <> [Reply to Message](#)

David Fanning wrote:

```
>
>
> I'd probably use a pointer to do this:
>
> IF Ptr_Valid(ptr) THEN ptr = Ptr_New([mylist]) ELSE $
>   *ptr = [Temporary(*ptr), mynewList]
>
>   finalList = Temporary(*ptr)
>
>
```

You should be careful if your number of concatenations become large. Using the method David suggests is fine for small n (I use it frequently), but if (say) you wanted to do it 90,000 times, the overhead becomes prohibitive. For example, the test program at the end of this message produced the following results:

```
% Compiled module: DELETEME.
```

```
IDL> deleteme
```

```
method 1 time elapsed = 17.2340002060
```

```
method 2 time elapsed = 0.0159997940
```

Method 1 was using David's method (again, perfectly fine for small n), and method 2 simply guesses at a large number of elements and counts the ones being used. This is much, much faster. If you really have no idea how big to guess, you could use CATCH to extend the array by another large chunk if it fills up.

Robert Moss, Ph.D.

```
pro deleteme
  compile_opt idl2

  t0 = systime( /sec )
  for i = 0, 90000 do begin
    IF ~Ptr_Valid(ptr) THEN ptr = Ptr_New([ i ]) ELSE $
      *ptr = [ Temporary(*ptr), i ]
  endfor
  finalList = Temporary(*ptr)
```

```
    print, "method 1 time elapsed = ", systime( /sec ) - t0, format =  
'(%"%"s",D13.10)'  
    PTR_FREE, ptr  
  
    t1 = systime( /sec )  
  
    finalList = intarr( 200000, /nozero )  
    count = 0  
    for i = 0, 90000 do begin  
        finalList[ i ] = i  
        count++  
    endfor  
    finalList = finalList[ 0:count-1 ]  
    print, "method 2 time elapsed = ", systime( /sec ) - t1, format =  
'(%"%"s",D13.10)'  
  
end
```
