Subject: Garbage collection and Memory
Posted by hevans[2] on Thu, 02 Jun 1994 15:55:55 GMT
View Forum Message <> Reply to Message

Hello,

I am using PV~Wave v4.01 (VAX/VMS) and after loading in large data sets and manipulating them, PV~Wave runs out of memory (core). This is despite creating more variables.

The question I have is: Is there any command that invokes a garbage collector to clean up the memory used? Or do I just have to save the session, exit and restart the session?

Regards,
--
Hugh Evans
European Space Research and Technology Centre - Noorwijk, Netherlands
Internet: hevans@wm.estec.esa.nl SPAN: ESTWM2::hevans

But she stopped herself. You didn't juggle matches in a firework factory.
(Terry Pratchett, Witches Abroad)

Subject: Re: Garbage collection and Memory
Posted by gumley on Thu, 02 Jun 1994 17:22:01 GMT
View Forum Message <> Reply to Message

In article <1994Jun2.155555@estwm0.wm.estec.esa.nl>,
hevans@estwm0.wm.estec.esa.nl (Hugh Evans) wrote:

> I am using PV~Wave v4.01 (VAX/VMS) and after loading in large data sets and
> manipulating them, PV~Wave runs out of memory (core). This is despite creating
> more variables.
>
> The question I have is: Is there any command that invokes a garbage collector
> to clean up the memory used? Or do I just have to save the session, exit and
> restart the session?

If you have a large variable array which is longer needed, just redefine it
as a single value variable, e.g.

```
array = fltarr(10000,10000)
(processing steps)
array = 0
```

See page 12-7 of the IDL User's Guide (v3.5).

--
Liam E. Gumley
NASA/GSFC Climate and Radiation Branch
Greenbelt MD, USA

Subject: Re: Garbage collection and Memory
Posted by pjclinch on Fri, 03 Jun 1994 09:59:28 GMT
View Forum Message <> Reply to Message

Liam E. Gumley (gumley@climate.gsfc.nasa.gov) wrote:
: In article <1994Jun2.155555@estwm0.wm.estec.esa.nl>,
: hevans@estwm0.wm.estec.esa.nl (Hugh Evans) wrote:

: > The question I have is: Is there any command that invokes a garbage collector
: > to clean up the memory used? Or do I just have to save the session, exit and
: > restart the session?

: If you have a large variable array which is longer needed, just redefine it
: as a single value variable, e.g.

: array = fltarr(10000,10000)
: (processing steps)
: array = 0

Or use the delvar command to trash it completely:
array=fltarr(10000,10000)
(processing steps)
delvar, array

Pete.
--
Peter Clinch University of Dundee
voice 44 382 60111 x 2604 Department of Medical Physics
fax 44 382 640177 Ninewells Hospital
email pjclinch@dux.dundee.ac.uk Dundee, DD1 9SY, Scotland, UK

Subject: Re: Garbage collection and Memory
Posted by hevans[2] on Fri, 03 Jun 1994 13:58:27 GMT
View Forum Message <> Reply to Message

In article <gumley-020694132201@macfeng.gsfc.nasa.gov>, gumley@climate.gsfc.nasa.gov
(Liam E. Gumley) writes:

|>In article <1994Jun2.155555@estwm0.wm.estec.esa.nl>,
|>hevans@estwm0.wm.estec.esa.nl (Hugh Evans) wrote:

|>
|>> I am using PV~Wave v4.01 (VAX/VMS) and after loading in large data sets and
|>> manipulating them, PV~Wave runs out of memory (core). This is despite creating
|>> more variables.
|>>
|>> The question I have is: Is there any command that invokes a garbage collector
|>> to clean up the memory used? Or do I just have to save the session, exit and
|>> restart the session?
|>
|>If you have a large variable array which is longer needed, just redefine it
|>as a single value variable, e.g.
|>
|>array = fltarr(10000,10000)
|>(processing steps)
|>array = 0
|>
|>See page 12-7 of the IDL User's Guide (v3.5).
|>

Perhaps I phrased my question badly.

I have discovered that after using Wave for an extended period that it slowly grabs more and more memory, even if new variables are not created, until finally it runs out of core memory. Whereas by saving the session and restarting it, the previous operation that crashed on a memory allocation problem will complete successfully.

This, by the way, is being done on VAX/VMS.

--
Hugh Evans
European Space Research and Technology Centre - Noorwijk, Netherlands
Internet: hevans@wm.estec.esa.nl SPAN: ESTWM2::hevans

It's a nice touch, when you're a houseguest, to make your bed.
It's a particularly nice touch to make it a place of delight
for your host's teen-age daughters.
P.J. O'Rourke - Modern Manners

Subject: Re: Garbage collection and Memory
Posted by landers on Fri, 03 Jun 1994 18:08:33 GMT
View Forum Message <> Reply to Message

In article <1994Jun2.155555@estwm0.wm.estec.esa.nl>, hevans@estwm0.wm.estec.esa.nl (Hugh Evans) writes:
|> Hello,
|>

|> I am using PV~Wave v4.01 (VAX/VMS) and after loading in large data sets and
|> manipulating them, PV~Wave runs out of memory (core). This is despite creating
|> more variables.
|>
|> The question I have is: Is there any command that invokes a garbage collector
|> to clean up the memory used? Or do I just have to save the session, exit and
|> restart the session?

That's one way. Or you can DELVAR all the unused variables. Or SAVE what you want to keep, then .RNEW a something.pro (this cleans all variables - unfortunately there's no DELVAR,/All), and then RESTORE. This will re-pack your variables into memory, and free up the unused stuff.

Another thing to do is be more careful with memory allocation. Be sure to free memory when you're done with it. Things like this are helpful:

```
a = 0   ; free up any memory associated with a (assuming it's an array)
a = some_large_array_expression
; some code...
a = 0   ; done with a, so free it up
```

When you do an assignment (like the 2nd expression above), WAVE has to allocate memory for the intermediate answer (the right hand side), and then do the assignment (point the left hand side variable at the result). If the left hand side is already defined, then it's memory is not free'd until just before the assignment. So you can free up most of the memory by doing an 'a = 0' so there's memory available for the RHS evaluation.

Of course this doesn't always cure everything, because WAVE has to be able to find enough unbroken memory for your results. Eventually, (depending on your code), everything can get fragmented (just like VMS does to your disks).

A really good way to frag up your memory is by 'growing' arrays. Like this:

```
while ( condition ) do begin
  x = calcualtion()
  a = [ a, x ]
end
```

Each time this loops thru, it needs a piece of memory a little bigger than the last time. It probably has to keep allocating a new block of memory for each loop, except for a few times when you'll get lucky. You use something like (1 + 2 + 3 + ... + N) hunks of memory, rather than just (N).

To fix this, allocate the array before the loop, and keep track of a pointer to the last used piece. Then trim the array when you're done. Either allocate the array to as much as you'll need (or more), or put some code in to grow it by a large ammount (like double it) when necessary.

```
a = fltarr( largest_possible )
i = 0
while condition do begin
   x = calc()
   a(i) = x
   i = i + n_elements(x)
end
a = a(0:i-1)
```

Remember that once you start a WAVE session, you can never 'give memory back'
to the operating system. It just marks it as unused, and will reuse it. this
is an artifact of C's malloc() and free() procedures.

|> Regards,
|> --
|> Hugh Evans
|> European Space Research and Technology Centre - Noorwijk, Netherlands
|> Internet: hevans@wm.estec.esa.nl SPAN: ESTWM2::hevans
|>
|> But she stopped herself. You didn't juggle matches in a firework factory.
|> (Terry Pratchett, Witches Abroad)

;Dave

Subject: Re: Garbage collection and Memory
Posted by thompson on Sat, 04 Jun 1994 15:49:06 GMT
View Forum Message <> Reply to Message

hevans@estwm0.wm.estec.esa.nl (Hugh Evans) writes:

> I have discovered that after using Wave for an extended period that it slowly
> grabs more and more memory, even if new variables are not created, until
> finally it runs out of core memory. Whereas by saving the session and
> restarting it, the previous operation that crashed on a memory allocation
> problem will complete successfully.

> This, by the way, is being done on VAX/VMS.

The problem is that the memory is getting fragmented. It would be nice if IDL
(and apparently PV-Wave) were more sophisticated about handling this--perhaps a
dot command to defragment the memory.

Bill Thompson

Subject: Re: Garbage collection and Memory
Posted by thompson on Sat, 04 Jun 1994 15:52:44 GMT
View Forum Message <> Reply to Message

hevans@estwm0.wm.estec.esa.nl (Hugh Evans) writes:

> I have discovered that after using Wave for an extended period that it slowly
> grabs more and more memory, even if new variables are not created, until
> finally it runs out of core memory. Whereas by saving the session and
> restarting it, the previous operation that crashed on a memory allocation
> problem will complete successfully.

> This, by the way, is being done on VAX/VMS.

It also strikes me that you could save the session, use .RNEW to clear out all
the memory, and restore it.

Bill Thompson

Subject: Re: Garbage collection and Memory
Posted by eharold on Thu, 09 Jun 1994 22:00:14 GMT
View Forum Message <> Reply to Message

In article <thompson.770745164@serts.gsfc.nasa.gov>, thompson@serts.gsfc.nasa.gov (William Thompson) writes:
|> hevans@estwm0.wm.estec.esa.nl (Hugh Evans) writes:
|>
|> >I have discovered that after using Wave for an extended period that it slowly
|> >grabs more and more memory, even if new variables are not created, until
|> >finally it runs out of core memory. Whereas by saving the session and
|> >restarting it, the previous operation that crashed on a memory allocation
|> >problem will complete successfully.
|>
|>
|> It also strikes me that you could save the session, use .RNEW to clear out all
|> the memory, and restore it.
|>

 But will this allow you to start up in the middle of a program?
i.e. can I Control-C a program; save,/all; save ,/routines; .RNEW;
and then restore everything and .continue from where I left off?

This is not an idle question. After a day carefully breaking up some
matrix calculations into 1-2 MB pieces that wouldn't stretch the memory of my
machine, a few hours into the run the sysadmin dropped by to warn me that
my process was taking up 30 megabytes! This really makes me wonder if there's
any way to deal with data sets that are larger than available memory.

Would it help if I cleared temporary variables and arrays every pass through
my main loops?

--

Elliotte Rusty Harold National Solar Observatory
eharold@sunspot.noao.edu Sunspot NM 88349

Subject: Re: Garbage collection and Memory
Posted by geomagic on Fri, 10 Jun 1994 00:33:54 GMT
View Forum Message <> Reply to Message

In article <1994Jun9.220014.28022@noao.edu> eharold@corona.sunspot.noao.edu (Elliotte Harold) writes:

In article <thompson.770745164@serts.gsfc.nasa.gov>, thompson@serts.gsfc.nasa.gov (William Thompson) writes:
|> hevans@estwm0.wm.estec.esa.nl (Hugh Evans) writes:
|>
|> >I have discovered that after using Wave for an extended period that it slowly
|> >grabs more and more memory, even if new variables are not created, until
|> >finally it runs out of core memory. Whereas by saving the session and
|> >restarting it, the previous operation that crashed on a memory allocation
|> >problem will complete successfully.
|>
|>
|> It also strikes me that you could save the session, use .RNEW to clear out all
|> the memory, and restore it.
|>

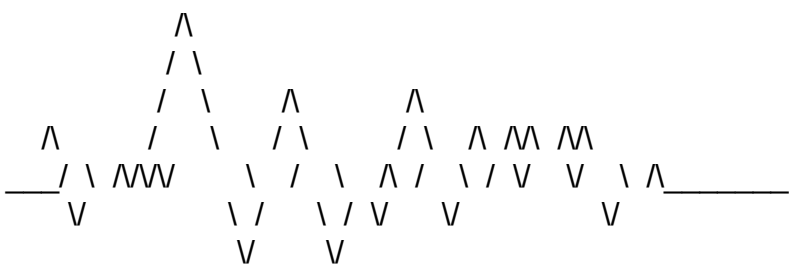
> But will this allow you to start up in the middle of a program? i.e.
> can I Control-C a program; save,/all; save ,/routines; .RNEW; and then
> restore everything and .continue from where I left off?
>
> This is not an idle question. After a day carefully breaking up some
> matrix calculations into 1-2 MB pieces that wouldn't stretch the
> memory of my machine, a few hours into the run the sysadmin dropped by
> to warn me that my process was taking up 30 megabytes! This really
> makes me wonder if there's any way to deal with data sets that are
> larger than available memory. Would it help if I cleared temporary
> variables and arrays every pass through my main loops?

Buy more memory. Seriously, if you look at your staff time costs of
dinking around trying to contort your software to fit a big problem
into a small amount of memory, it's not cost effective to NOT buy
more memory. With 32 MB of memory for most workstations costing between
$1200-$1500, it's not worth wasting lots of time playing games with

exotic memory tricks.

It might help to allocate the biggest chuck of memory that any
intermediate calculate requires, instead of creating and deleting
variables. That might reduce memory fragmentation.

Dan O'Connell
geomagic@seismo.do.usbr.gov
Seismotectonics Group, U.S. Bureau of Reclamation
Denver Federal Center, P.O. Box 25007 D-3611, Denver, CO 80225
"We do custom earthquakes (for food)"
or
"Just more roadkill on the information superhighway"



--
Dan O'Connell
geomagic@seismo.do.usbr.gov
Seismotectonics Group, U.S. Bureau of Reclamation

Subject: Re: Garbage collection and Memory
Posted by eharold on Fri, 10 Jun 1994 14:17:57 GMT
View Forum Message <> Reply to Message

In article <GEOMAGIC.94Jun9173354@moe.seismo.do.usbr.gov>,
geomagic@seismo.do.usbr.gov (Dan O-Connell) writes:
|> Buy more memory. Seriously, if you look at your staff time costs of
|> dinking around trying to contort your software to fit a big problem
|> into a small amount of memory, it's not cost effective to NOT buy
|> more memory. With 32 MB of memory for most workstations costing between
|> $1200-$1500, it's not worth wasting lots of time playing games with
|> exotic memory tricks.
|>

The problem is that in my field (astronomy)
it's always been and probably always will be VERY easy to produce data
sets that overwhelm available memory. The data we collect seems to grow
much faster than the memory capacity of our computers. The current
project I'm working on would really like 512 MB, about the maximum you
can shove in a Sparc. Buying that (which is not totally out of the

question) would cost around $20,000. This is the same order of magnitude as my annual salary so it's not all that cost-ineffective for me to spend a few days playing tricks with memory for data sets of this size.

It isn't too much trouble to fit the code into a 128 MB machine. I actually have access to a Sparc with 128 MB but this machine is shared among multiple astronomers, all of whom want to run their own 128 MB (or larger) jobs. Thus as a grad student I'm one of the first to get shoved off the machine when load is high. Therefore it becomes very important to me to fit my code into as little actual memory as possible.

Of course different analyses may apply if the professionals working on your code are paid more than the average grad student, or if the data sets are not astronomically large and don't quadruple every time someone makes a better CCD. I'm not particularly familiar with geological seismology. How fast does your data grow?

--

Elliotte Rusty Harold National Solar Observatory
eharold@sunspot.noao.edu Sunspot NM 88349

Subject: Re: Garbage collection and Memory
Posted by eharold on Fri, 10 Jun 1994 14:44:29 GMT
View Forum Message <> Reply to Message

|> In article <thompson.770745164@serts.gsfc.nasa.gov>, thompson@serts.gsfc.nasa.gov (William Thompson) writes:
|> |> hevans@estwm0.wm.estec.esa.nl (Hugh Evans) writes:
|> |>
|> |> >I have discovered that after using Wave for an extended period that it slowly
|> |> >grabs more and more memory, even if new variables are not created, until
|> |> >finally it runs out of core memory. Whereas by saving the session and
|> |> >restarting it, the previous operation that crashed on a memory allocation
|> |> >problem will complete successfully.
|> |>

I think I've bumped across this one myself and although the problem is documented, it's still nasty. Consider the following sequence

a=indgen(really_big_number)
a=findgen(some_other_number)

The allocation from the first statement isn't freed when the second assignment to a is made, even though, near as I can tell, there's no way to get at the values stored in a from the first assignment. On the other

hand the sequence of statements

a=indgen(really_big_number)
a=0
a=findgen(some_other_number)

will free the first allocation. Why does assignment to a scalar free the memory allocated to a but assignment to an array just leaves it hanging? I have no idea, but this is how IDL works and it's quite annoying. Since after the a=findgen(some_other_number) there's no way to get at the original contents of a I see no reason not to automatically free a before making the assignment, but currently that's not what happens. Consequently unless you're careful to manully free arrays by setting them to scalars before reassigning them, you'll pile up a lot of allocated but unused memory.

--

Elliotte Rusty Harold National Solar Observatory
eharold@sunspot.noao.edu Sunspot NM 88349

Subject: Re: Garbage collection and Memory
Posted by geomagic on Fri, 10 Jun 1994 19:37:07 GMT
View Forum Message <> Reply to Message

In article <1994Jun10.141757.23823@noao.edu> eharold@corona.sunspot.noao.edu (Elliotte Harold) writes:

|> Buy more memory. Seriously, if you look at your staff time costs of
|> dinking around trying to contort your software to fit a big problem
|> into a small amount of memory, it's not cost effective to NOT buy
|> more memory. With 32 MB of memory for most workstations costing between
|> $1200-$1500, it's not worth wasting lots of time playing games with
|> exotic memory tricks.
|>

> The problem is that in my field (astronomy)
> it's always been and probably always will be VERY easy to produce data
> sets that overwhelm available memory. The data we collect seems to grow
> much faster than the memory capacity of our computers. The current
> project I'm working on would really like 512 MB, about the maximum you
> can shove in a Sparc. Buying that (which is not totally out of the
> question) would cost around $20,000. This is the same order of magnitude
> as my annual salary so it's not all that cost-ineffective for me to spend
> a few days playing tricks with memory for data sets of this size.
>
> It isn't too much trouble to fit the code into a 128 MB

> machine. I actually have access to a Sparc with 128 MB but this
> machine is shared among multiple astronomers, all of whom want to
> run their own 128 MB (or larger) jobs. Thus as a grad student I'm
> one of the first to get shoved off the machine when load is high.
> Therefore it becomes very important to me to fit my code into as little
> actual memory as possible.

Have someone throw party for a weekend and the run your stuff on the
machine then (that's how I finished my dissertation) ;).
>
> Of course different analyses may apply if the professionals
> working on your code are paid more than the average grad student,
> or if the data sets are not astronomically large and don't quadruple
> every time someone makes a better CCD. I'm not particularly familiar with
> geological seismology. How fast does your data grow?

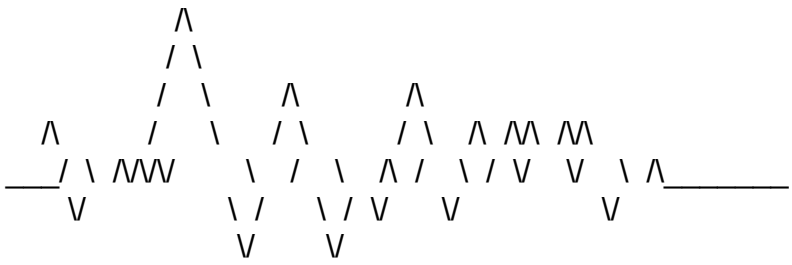
In earthquake seismology, the rate of data growth has increased substantially
as more broadband seismographic stations have been installed. What some
people would like to do with the available data is restricted by
maximum memory and processing speed limits (3d whole-earth global
data set broadband waveform inversions, 3d calculations of strong motion
produced by complex fault ruptures, etc.). So most people work with
the resources available or find special configurations (big memory,
parallel processors, etc.) to solve "big" problems.

We could use machines with ~20 gigabytes of memory and teraflop
throughputs to investigate interesting and important aspects of
three-dimensional wave propagation related to things like earthquake
ground motions and related hazards.

Does using delvar deallocate memory effectively? Something like
a=findgen(n)
...
delvar,a
a=fltarr(m)
...

Also, can't you allocate the maximum size arrays you'll need and just
use them throughout your code with appropriate subscript ranges? That
would cut down on allocation/deallocation fragmentation.

Dan O'Connell
geomagic@seismo.do.usbr.gov
Seismotectonics Group, U.S. Bureau of Reclamation
Denver Federal Center, P.O. Box 25007 D-3611, Denver, CO 80225
"We do custom earthquakes (for food)"
or
"Just more roadkill on the information superhighway"



--
Dan O'Connell
geomagic@seismo.do.usbr.gov
Seismotectonics Group, U.S. Bureau of Reclamation

Subject: Re: Garbage collection and Memory
Posted by landers on Fri, 10 Jun 1994 22:34:42 GMT
View Forum Message <> Reply to Message

In article <1994Jun9.220014.28022@noao.edu>, eharold@corona.sunspot.noao.edu (Elliotte Harold) writes:
|> In article <thompson.770745164@serts.gsfc.nasa.gov>, thompson@serts.gsfc.nasa.gov (William Thompson) writes:
|> |> hevans@estwm0.wm.estec.esa.nl (Hugh Evans) writes:
|> |>
|> |> [snip]
|> |>
|> |> It also strikes me that you could save the session, use .RNEW to clear out all
|> |> the memory, and restore it.
|> |>
|>
|> But will this allow you to start up in the middle of a program?
|> i.e. can I Control-C a program; save,/all; save ,/routines; .RNEW;
|> and then restore everything and .continue from where I left off?

No, you can't .RNEW from anywhere but $MAIN$
You could DELVAR each and every variable....
Also, note that the SAVE will only save local variables in whatever subroutine you happen to interrupt - may not be where you want. So you may not be able to repack the proper variables.

|> [snip]

|> Would it help if I cleared temporary variables and arrays every pass through
|> my main loops?

Definately (usually). Clear (by setting = 0) everything when you're done with it. Especially arrays. Especially large arrays.

If you have a temp array that you reuse each pass thru a loop, it will eat memory.

Example:

```
for i = 0, 1000 do begin
  a = something()
  ... etc...
endfor
```

In this loop, when the a = something line is encountered, WAVE/IDL mallocs memory for the result of something(). A is still using memory. Then, memory for A is free()d, and a is pointed to the new result.

Assuming a is an array (and something() doesn't use it in the calcualtion), doing this:

```
  a = 0 ; free memory
  a = something()
```

forces WAVE/IDL to free most of a's memory before it goes looking for memory to use for something().

WAVE/IDL can't (in general) free a before it does something(), because a might be needed in the calculation.... Only you know if that's so.

;Dave

Subject: Re: Garbage collection and Memory
Posted by pendleton on Sun, 12 Jun 1994 20:28:09 GMT

View Forum Message <> Reply to Message

In article <1994Jun10.223442.13293@mksol.dseg.ti.com>, landers@tsunami.dseg.ti.com (David Landers) writes:
> In article <1994Jun9.220014.28022@noao.edu>, eharold@corona.sunspot.noao.edu (Elliotte Harold) writes:
> |> In article <thompson.770745164@serts.gsfc.nasa.gov>, thompson@serts.gsfc.nasa.gov (William Thompson) writes:
> |> |> hevans@estwm0.wm.estec.esa.nl (Hugh Evans) writes:
> |> |>
> |> |> [snip]
> |> |>
> |> |> It also strikes me that you could save the session, use .RNEW to clear out all
> |> |> the memory, and restore it.
> |> |>
> |>
> |> But will this allow you to start up in the middle of a program?
> |> i.e. can I Control-C a program; save,/all; save ,/routines; .RNEW;

> |> and then restore everything and .continue from where I left off?
>
> |> Would it help if I cleared temporary variables and arrays every pass through
> |> my main loops?
>
> Definately (usually). Clear (by setting = 0) everything when you're done
> with it. Especially arrays. Especially large arrays.
>
> If you have a temp array that you reuse each pass thru a loop, it will eat
> memory.
>
Here's another hint for using memory efficiently, this time with respect to large temporary arrays in frequently called procedures.

If you are using a temp array of a fixed number of elements each and every time you pass a called procedure, you might find it most efficient to put the variable into a COMMON block. This way, the memory is allocated once and it's always there. Use an N_ELEMENTS statement, for example, to determine if the variable has already been defined.

```
Common Temp_Block, A
If (N_Elements(A) eq 0) then A = FltArr(100000)
A(*) = 0.
```

If you want to delete the variable at a later time, simply call up your COMMON block from another procedure and set the variable to 0B.

Jim Pendleton, Programmer Analyst/Technical Services Specialist
GRO/OSSE, Dept. Physics & Astronomy
Northwestern University
j-pendleton@nwu.edu (708) 491-2748 (708) 491-3135 [FAX]
http://www.astro.nwu.edu/astro/pendleton