
Subject: Windows XP memory limitation?

Posted by [dcw_yip](#) on Thu, 04 Dec 2003 23:01:27 GMT

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

The code that I'm working on is dying with a "Unable to allocate memory: to make array." in MESH_SURFACEAREA. It dies when the machine has allocated a little over 800MB of memory. Why does it do that? I've tried it on two different windows machines with 4GB and 2GB of RAM respectively. So there should be plenty of RAM available. It doesn't run into this problem under Linux or on a Mac. Is there some sort of RAM limitation with IDL under Windows?

thanks,

David

Subject: Re: Windows XP memory limitation?

Posted by [dcw_yip](#) on Wed, 10 Dec 2003 00:11:36 GMT

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

Thanks for the response. I actually read that thread before posting. I even tried editbase but it didn't help.

I understand your point about MFC vs console apps due to the loading of DLLs. Though in the scenario I'm describing, I can allocate 10 times the amount of memory in a C program than under IDL while IDL is in it's crash state. If IDL can't get 120MB of contiguous memory from the OS then why can a C program get 1200MB under the same conditions? I might not have made it clear that I leave IDLDE up after it crashes out with the error. I then go to another window and run the C program to allocate memory.

David

"Karl Schultz" <kschultz_no_spam@rsinc.com> wrote in message news:<vtbtkgsbkl0580@corp.supernews.com>...

> "David Yip" <dcw_yip@yahoo.com> wrote in message

> news:201431cc.0312081038.47b3503e@posting.google.com...

>> Thanks everyone for the responses. Unfortunately none of them worked.

>> Contrary to what RSI says, there must be a built in memory limitation

>> or bug in IDL. I'm running 6.0 by the way. Once IDL crashes out with

>> the memory error, if I type in "BYTARR(120000000)" in the command

>> window I get "Unable to allocate memory: to make array." Even though

>> I still should have about 2GB of RAM available. I'm using the /3GB

>> flag in XP Pro. But if I try to allocate the same amount of memory in

>> C using "malloc(120000000)" it works just fine. This is while IDL is

>> in it's crash state. So there is that much available memory available

>> in the system. In fact if I use "malloc(1200000000)" in C it still

>> works. That's 10 times the amount of memory that fails under IDL
>> under the same conditions.
>
> There's still a big difference in the largest contiguous block of memory
> that you can allocate from a stand-alone C program, a Win32 application, and
> a Win32 application with MFC. If you build your C test program as a Win32
> app with MFC, I doubt that it will be able to allocate a contiguous block as
> big as a simple console app can.
>
> You may also want to read the thread "Memory Headaches" posted to this
> newsgroup starting Aug 1, 2002. There is a lot more detail in the thread
> and some mention of some tools you can use to determine what is fragmenting
> your memory space.
>
> IDL has no self-imposed memory limitations that might be responsible for
> your observations.
>
> Karl

Subject: Re: Windows XP memory limitation?
Posted by [Rick Towler](#) on Wed, 10 Dec 2003 01:13:50 GMT
[View Forum Message](#) <> [Reply to Message](#)

"David Yip" wrote in message...

> I understand your point about MFC vs console apps due to the loading
> of DLLs. Though in the scenario I'm describing, I can allocate 10
> times the amount of memory in a C program than under IDL while IDL is
> in it's crash state. If IDL can't get 120MB of contiguous memory from
> the OS then why can a C program get 1200MB under the same conditions?
> I might not have made it clear that I leave IDLDE up after it crashes
> out with the error. I then go to another window and run the C program
> to allocate memory.

It isn't the fact that a MFC app is running but that there are limits on the amount of contiguous RAM a MFC app can allocate. IDL is built as a win32 app with MFC. It runs into these limits. Your simple C program isn't. It doesn't run into these limits. Like Karl said, compile your simple C program as a MFC application and then try it.

-Rick

>
> David
>
> "Karl Schultz" wrote in message ...

>> "David Yip" wrote in message...
>>> Thanks everyone for the responses. Unfortunately none of them worked.
>>> Contrary to what RSI says, there must be a built in memory limitation
>>> or bug in IDL. I'm running 6.0 by the way. Once IDL crashes out with
>>> the memory error, if I type in "BYTARR(1200000000)" in the command
>>> window I get "Unable to allocate memory: to make array." Even though
>>> I still should have about 2GB of RAM available. I'm using the /3GB
>>> flag in XP Pro. But if I try to allocate the same amount of memory in
>>> C using "malloc(1200000000)" it works just fine. This is while IDL is
>>> in it's crash state. So there is that much available memory available
>>> in the system. In fact if I use "malloc(1200000000)" in C it still
>>> works. That's 10 times the amount of memory that fails under IDL
>>> under the same conditions.
>>
>> There's still a big difference in the largest contiguous block of memory
>> that you can allocate from a stand-alone C program, a Win32 application,
>> and
>> a Win32 application with MFC. If you build your C test program as a
>> Win32
>> app with MFC, I doubt that it will be able to allocate a contiguous
>> block as
>> big as a simple console app can.
>>
>> You may also want to read the thread "Memory Headaches" posted to this
>> newsgroup starting Aug 1, 2002. There is a lot more detail in the
>> thread
>> and some mention of some tools you can use to determine what is
>> fragmenting
>> your memory space.
>>
>> IDL has no self-imposed memory limitations that might be responsible for
>> your observations.
>>
>> Karl

Subject: Re: Windows XP memory limitation?
Posted by [Karl Schultz](#) on Wed, 10 Dec 2003 02:03:45 GMT
[View Forum Message](#) <> [Reply to Message](#)

"Rick Towler" <rtowler@u.washington.edu> wrote in message
news:br5s51\$leu\$1@nntp6.u.washington.edu...

>

> "David Yip" wrote in message...

>

>> I understand your point about MFC vs console apps due to the loading
>> of DLLs. Though in the scenario I'm describing, I can allocate 10
>> times the amount of memory in a C program than under IDL while IDL is

>> in it's crash state. If IDL can't get 120MB of contiguous memory from
>> the OS then why can a C program get 1200MB under the same conditions?
>> I might not have made it clear that I leave IDLDE up after it crashes
>> out with the error. I then go to another window and run the C program
>> to allocate memory.
>
> It isn't the fact that a MFC app is running but that there are limits on
the
> amount of contiguous RAM a MFC app can allocate. IDL is built as a win32
> app with MFC. It runs into these limits. Your simple C program isn't.
It
> doesn't run into these limits. Like Karl said, compile your simple C
> program as a MFC application and then try it.
>
> -Rick

It's really contiguous virtual address space that is needed.

David:

Each process has its own virtual address space and not all system-loaded
DLL's are necessarily mapped into every process. IDL has a bunch of DLL's
mapped into its process address space that a standalone C program does not.

So, if you have IDL running (or in crash state as you say) in one process,
that has no effect on the process you are running with the standalone C
program. The C program doesn't need all the DLL's IDL needs, and so can
find a bigger block of contiguous virtual address space. Your assertion of
"under the same conditions" really doesn't apply here, as the virtual
address spaces have much different utilizations.

I Really Think that some DLL's are getting loaded into your address space
when IDL runs in such a way that it's breaking a big piece of VM up into
small parts. A tool like VMMAP would tell you for sure. And it may not
even be a DLL that is used directly by IDL. I was surprised to see a nVidia
desktop applet DLL of some sort loaded into every windowed app's VM. It was
loaded into VM so that it fragmented VM a little bit. I guess there's some
hook in the OS where an applet such as this can get a DLL loaded for every
windowed app on the system. Maybe the graphics drivers need to monitor all
the windows or this DLL supports some sort of fancy GUI gizmo that lets you
tweak the card parameters. I don't really know much more than that. But it
was there, and it fraged my VM. You can try rebasing that DLL or disabling
the offending applet or control component.

Karl

>
>
>>

>> David
>>
>> "Karl Schultz" wrote in message ...
>>> "David Yip" wrote in message...
>>>> Thanks everyone for the responses. Unfortunately none of them
worked.
>>>> Contrary to what RSI says, there must be a built in memory
limitation
>>>> or bug in IDL. I'm running 6.0 by the way. Once IDL crashes out
with
>>>> the memory error, if I type in "BYTARR(1200000000)" in the command
>>>> window I get "Unable to allocate memory: to make array." Even
though
>>>> I still should have about 2GB of RAM available. I'm using the /3GB
>>>> flag in XP Pro. But if I try to allocate the same amount of memory
in
>>>> C using "malloc(1200000000)" it works just fine. This is while IDL
is
>>>> in it's crash state. So there is that much available memory
available
>>>> in the system. In fact if I use "malloc(1200000000)" in C it still
>>>> works. That's 10 times the amount of memory that fails under IDL
>>>> under the same conditions.
>>>
>>> There's still a big difference in the largest contiguous block of
memory
>>> that you can allocate from a stand-alone C program, a Win32
application,
> and
>>> a Win32 application with MFC. If you build your C test program as a
> Win32
>>> app with MFC, I doubt that it will be able to allocate a contiguous
> block as
>>> big as a simple console app can.
>>>
>>> You may also want to read the thread "Memory Headaches" posted to this
>>> newsgroup starting Aug 1, 2002. There is a lot more detail in the
> thread
>>> and some mention of some tools you can use to determine what is
> fragmenting
>>> your memory space.
>>>
>>> IDL has no self-imposed memory limitations that might be responsible
for
>>> your observations.
>>>
>>> Karl
>

>

Subject: Re: Windows XP memory limitation?
Posted by [dcw_yip](#) on Wed, 10 Dec 2003 19:39:22 GMT
[View Forum Message](#) <> [Reply to Message](#)

Just as an experiment I tried the same thing with a Win32 MFC App to allocate 1200MB instead of the console app I was using before. It worked as well. So it appears that the memory allocation issues aren't really generic to MFC Apps under Windows but something specific to IDL running under Windows. Some very good points have been made that IDL most likely uses DLLs that cause this to happen. Even just after starting a fresh instance of IDLDE I can't allocate 1200MB on the command line. Does anyone know where I can download "VMMAP"? The only references to it I can find are to the CDROM that comes with the book. If not, I guess I can write my own using virtualqueryex.

David

"Karl Schultz" <kschultz_no_spam@rsinc.com> wrote in message news:<vtcvkgov7oul77@corp.supernews.com>...

> "Rick Towler" <rtowler@u.washington.edu> wrote in message
> news:br5s51\$leu\$1@nntp6.u.washington.edu...

>>

>> "David Yip" wrote in message...

>>

>>> I understand your point about MFC vs console apps due to the loading
>>> of DLLs. Though in the scenario I'm describing, I can allocate 10
>>> times the amount of memory in a C program than under IDL while IDL is
>>> in it's crash state. If IDL can't get 120MB of contiguous memory from
>>> the OS then why can a C program get 1200MB under the same conditions?
>>> I might not have made it clear that I leave IDLDE up after it crashes
>>> out with the error. I then go to another window and run the C program
>>> to allocate memory.

>>

>> It isn't the fact that a MFC app is running but that there are limits on
> the

>> amount of contiguous RAM a MFC app can allocate. IDL is built as a win32
>> app with MFC. It runs into these limits. Your simple C program isn't.
> It

>> doesn't run into these limits. Like Karl said, compile your simple C
>> program as a MFC application and then try it.

>>

>> -Rick

>

> It's really contiguous virtual address space that is needed.

>

> David:

> Each process has its own virtual address space and not all system-loaded
 > DLL's are necessarily mapped into every process. IDL has a bunch of DLL's
 > mapped into its process address space that a standalone C program does not.
 >

> So, if you have IDL running (or in crash state as you say) in one process,
 > that has no effect on the process you are running with the standalone C
 > program. The C program doesn't need all the DLL's IDL needs, and so can
 > find a bigger block of contiguous virtual address space. Your assertion of
 > "under the same conditions" really doesn't apply here, as the virtual
 > address spaces have much different utilizations.
 >

> I Really Think that some DLL's are getting loaded into your address space
 > when IDL runs in such a way that it's breaking a big piece of VM up into
 > small parts. A tool like VMMAP would tell you for sure. And it may not
 > even be a DLL that is used directly by IDL. I was surprised to see a nVidia
 > desktop applet DLL of some sort loaded into every windowed app's VM. It was
 > loaded into VM so that it fragmented VM a little bit. I guess there's some
 > hook in the OS where an applet such as this can get a DLL loaded for every
 > windowed app on the system. Maybe the graphics drivers need to monitor all
 > the windows or this DLL supports some sort of fancy GUI gizmo that lets you
 > tweak the card parameters. I don't really know much more than that. But it
 > was there, and it fraged my VM. You can try rebasing that DLL or disabling
 > the offending applet or control component.
 >

> Karl

>

>>

>>

>>>

>>> David

>>>

>>> "Karl Schultz" wrote in message ...

>>>> "David Yip" wrote...

>>>> > Thanks everyone for the responses. Unfortunately none of them
 > worked.

>>>> > Contrary to what RSI says, there must be a built in memory
 > limitation

>>>> > or bug in IDL. I'm running 6.0 by the way. Once IDL crashes out
 > with

>>>> > the memory error, if I type in "BYTARR(1200000000)" in the command
 >>>> > window I get "Unable to allocate memory: to make array." Even
 > though

>>>> > I still should have about 2GB of RAM available. I'm using the /3GB
 >>>> > flag in XP Pro. But if I try to allocate the same amount of memory
 > in

>>>> > C using "malloc(1200000000)" it works just fine. This is while IDL
 > is

>>>> > in it's crash state. So there is that much available memory
> available
>>>> > in the system. In fact if I use "malloc(1200000000)" in C it still
>>>> > works. That's 10 times the amount of memory that fails under IDL
>>>> > under the same conditions.
>>>>
>>>> There's still a big difference in the largest contiguous block of
> memory
>>>> that you can allocate from a stand-alone C program, a Win32
> application,
> and
>>>> a Win32 application with MFC. If you build your C test program as a
> Win32
>>>> app with MFC, I doubt that it will be able to allocate a contiguous
> block as
>>>> big as a simple console app can.
>>>>
>>>> You may also want to read the thread "Memory Headaches" posted to this
>>>> newsgroup starting Aug 1, 2002. There is a lot more detail in the
> thread
>>>> and some mention of some tools you can use to determine what is
> fragmenting
>>>> your memory space.
>>>>
>>>> IDL has no self-imposed memory limitations that might be responsible
> for
>>>> your observations.
>>>>
>>>> Karl
>>
>>
