

Re: High-performance IO

Source:

<http://www.tech-archive.net/Archive/Development/microsoft.public.win32.programmer.kernel/2006-08/msg00397.htm>

- *From:* "anton bassov" <soviet_bloke@xxxxxxxxxxx>
 - *Date:* 24 Jul 2006 18:23:13 -0700
-

Hi mate

As I can see, things are not as seamless as I thought – it looks like you are bound by some logical constraints of your program, so that you need asynchronous IO and multiple threads. In such case the only thing you can optimize is memory buffer

That's why I want to use AWE -- physically contiguous and non paged.

First of all, memory does not have to be physically contiguous – the only thing you need is to make it resident and locked

But I don't know whether the disk drivers are supposed to support 64-bit (36, in fact) physical addressing.

Drivers are not concerned about things like that – after all, they deal only with linear addresses. Furthermore, AWE can utilize up to 4 G of memory even without PAE. In such case, as far as your program is concerned, memory that has been mapped with `MapUserPhysicalPages()` is just non-paged one, with couple of limitation that AWE imposes (i.e. non-sharable and always read/write). The only problem is that AWE requires `SeLockMemory` privilege

Anton Bassov

Piotr Wyderski wrote:

Thank for your answers,

Re: High-performance IO

anton bassov wrote:

First of all, "one file-one thread" strategy is not the best one in your situation.

Obviously, but it is the existing architecture and I cannot change it.

These files may be physically located in totally different parts of the disk. Therefore, every time context switch occur, the system has to update the position of the head, i.e. you have to search the disk all the time.

Doesn't Windows sort queued request to minimize seek time (the elevator algoritm etc.)? Anyway, it is the reason I want to make IO transfers in large chunks. Reasonably bounded latency does not matter, throughput does.

I think the best thing to do is to process files sequentially, i.e. one by one

Cannot be done, the files contain only partial information. Only a specified constelation of them provide a complete set of data one can call record.

Second, as I have already mentioned in my previous message, you should try to avoid page faults.

That's why I want to use AWE — physically contiguous and non paged. But I don't know whether the disk drivers are supposed to support 64-bit (36, in fact) physical addressing.

To summarize, the fact that you are about to access files only once and do so sequentially gives you a HUGE potential for optimization in terms of minimizing the number of disk accesses and disk searching.

Yes, I want to make use of this scenario.

Best regards
Piotr Wyderski

Re: High-performance IO