Virtual Memory Optimization Guide Rev. 4.0 - Final

Virtual Memory

Back in the 'good old days' of command prompts and 1.2MB floppy disks, programs needed very little RAM to run because the main (and almost universal) operating system was Microsoft DOS and its memory footprint was small. That was truly fortunate because RAM at that time was horrendously expensive. Although it may seem ludicrous, 4MB of RAM was considered then to be an incredible amount of memory.

However when Windows became more and more popular, 4MB was just not enough. Due to its GUI (Graphical User Interface), it had a larger memory footprint than DOS. Thus, more RAM was needed.

Unfortunately, RAM prices did not decrease as fast as RAM requirement had increased. This meant that Windows users had to either fork out a fortune for more RAM or run only simple programs. Neither were attractive options. An alternative method was needed to alleviate this problem.

The solution they came up with was to use some space on the hard disk as extra RAM. Although the hard disk is much slower than RAM, it is also much cheaper and users always have a lot more hard disk space than RAM. So, Windows was designed to create this pseudo-RAM or in Microsoft's terms - Virtual Memory, to make up for the shortfall in RAM when running memory-intensive programs.

How Does It Work?

Virtual memory is created using a special file called a swapfile or paging file.

Whenever the operating system has enough memory, it doesn't usually use virtual memory. But if it runs out of memory, the operating system will page out the least recently used data in the memory to the swapfile in the hard disk. This frees up some memory for your applications. The operating system will continuously do this as more and more data is loaded into the RAM.

However, when any data stored in the swapfile is needed, it is swapped with the least recently used data in the memory. This allows the swapfile to behave like RAM although programs cannot run directly off it. You will also note that because the operating system cannot directly run programs off the swapfile, some programs may not run even with a large swapfile if you have too little RAM.

Swapfile Vs. Paging File

We have all been using the terms swapfile and paging file interchangeably. Even Microsoft invariably refers to the paging file as the swapfile and vice versa. However, the swapfile and paging file are two different entities. Although both are used to create virtual memory, there are subtle differences between the two.

The main difference lies in their names. Swapfiles operate by swapping entire processes from system memory into the swapfile. This immediately frees up memory for other applications to use.

In contrast, paging files function by moving "pages" of a program from system memory into the paging file. These pages are 4KB in size. The entire program does not get swapped wholesale into the paging file.

While swapping occurs when there is heavy demand on the system memory, paging can occur preemptively. This means that the operating system can page out parts of a program when it is minimized or left idle for some time. The memory used by the paged-out portions are not immediately released for use by other applications. Instead, they are kept on standby.

If the paged-out application is reactivated, it can instantly access the paged-out parts (which are still stored in system memory). But if another application requests for the memory space, then the system memory held by the paged-out data is released for its use. As you can see, this is really quite different from the way a swapfile works.

Swapfiles were used in old iterations of Microsoft Windows, prior to Windows 95. From Windows 95 onwards, all Windows versions use only paging files. Therefore, the correct term for the file used to create virtual memory in current operating systems is paging file, not swapfile.

Because both swapfiles and paging files do the same thing - create virtual memory, people will always refer to swapfiles and paging files interchangeably. Let's just keep in mind their innate differences.

Do We Still Need A Paging File?

Even today, when the average home user's computer comes with at least 256MB of RAM, the paging file is still very important. While the large amount of RAM in the average user's computer makes the risk of memory shortage much less of a worry with single applications now than it was back then; the paging file is essential when multi-tasking.

Note that over the years, the emphasis has changed to multi-tasking. No longer will people be solely stuck to using one application at a time. In fact, it is common to have 10 or more applications running simultaneously!

For example, I normally have the following applications running at the same time :-

+ Microsoft Outlook
+ Internet browsers like Maxthon and Firefox
+ An FTP client
+ Instant messengers like Trillian and MSN Messenger
+ A download manager like FlashGet
+ Macromedia Dreamweaver
+ P2P clients like ShareAza
+ An antivirus software
+ Adobe Acrobat Reader with a few PDF documents opened

That's a total of 10-12 applications running simultaneously!

Even with 256MB of RAM, it would be impossible to load everything into memory. A paging file is needed to store the least used data in the memory so that I can open up all those applications I need. And let's not forget the disk cache.

Operating systems like Windows 98 and Windows XP allocate a sizeable portion of the RAM to the disk cache. This speeds up accesses to hard disk data by caching the most frequently used as well as data that are most likely to be accessed next by the computer. This cuts down on the amount of available RAM. So, without a paging file, you won't be able to open many applications even if your computer has 256MB of RAM.

Finally, some programs require the use of a paging file to function properly. It may be to store sensitive data on something less volatile than the RAM or to ensure the computer will have sufficient memory to run it. But whatever the reasons, a paging file is needed in order for these programs to run.

Why Optimize The Paging File?

Unless your computer is truly loaded with RAM, it will almost always use the paging file. As such, its performance affects the performance of the whole computer.

Now, using a paging file may sound like a really cheap way to run memory intensive programs without the expense of buying more RAM. However, even the fastest hard disk is more than an order of magnitude slower than the slowest RAM.

Even the fastest hard disk is currently over 70X slower than the dual-channel PC2700 DDR memory common in many computers. Let's not even start comparing the hard disk with faster RAM solutions like PC3200 DDR memory or PC2-4200 DDR2 memory.

So, paging file is only a stopgap solution for the lack of sufficient RAM. As long as you use the paging file, there will always be performance degradation. The ideal solution for insufficient RAM is always more RAM, not more Virtual Memory. But since we can't afford all the RAM we want, a paging file is necessary for us to run today's memory guzzling programs.

As you can tell, more isn't better for the paging file because more paging file space will only give you the ability to run more memory intensive programs at once. It will not speed up your system. But what we can do is to optimize the paging file so that the performance degradation when using it is minimized.

So How Do We Optimize The Paging File?

There have been many theories on how to optimize the paging file. The most important ones are listed below :-

+ Making the paging file contiguous.
+ Moving the paging file to the outer tracks of the hard disk.
+ Creating a huge paging file.
+ Moving the paging file to a different partition in the same hard disk.
+ Moving the paging file to a different hard disk.
+ Creating multiple paging files
+ Moving the paging file to a RAID array
+ Moving the paging file to a RAM drive
+ Reducing reliance on virtual memory

We will examine those methods and see what will work and what won't.

Virtual Memory Then And Now

Windows 3.1

Back in the good old days of DOS 6.22 and Windows 3.1, everyone knew that creating a permanent swapfile was the key to optimal swapfile performance. This was because Windows 3.1 only creates permanent swapfiles that are contiguous.

A contiguous swapfile is a swapfile that consists of an uninterrupted block of hard disk space. When a swapfile is contiguous, the read-write heads of the hard disk can read and write data on the swapfile in a continuous fashion.

In Windows 3.1, if the swapfile was set up as a temporary swapfile which is created every time Windows 3.1 boots up, it will end up at the end of the hard disk and fragmented too. So, every time the swapfile is read from or written to, the hard disk heads have to seek all over the platters to conduct those operations.

Needless to say, this greatly erodes the performance of the swapfile. That's why it was important to make the swapfile permanent in Windows 3.1 - so that the swapfile will become contiguous.

Windows 95 And Above

From Windows 95 onwards, Microsoft encouraged the use of its new dynamic virtual memory system. Of course, there is nothing new about the virtual memory part but the keyword in this new technique is dynamic.

The new dynamic virtual memory system no longer relies on a fixed-size swapfile but a paging file that dynamically resizes itself according to need. When the computer runs out of memory, more memory is created by increasing the size of the paging file. Once the virtual memory is freed up, theoretically the paging file diminishes in size.

Microsoft claims that while its dynamic virtual memory system will create a fragmented paging file, it is still faster than Windows 3.1's static virtual memory system. As a bonus, no hard disk space will be tied up in a permanent paging file.

However, this dynamic virtual memory system does have a big disadvantage - it cannot be moved to the outer tracks of the hard disk platters.

Dynamic Paging Files And Data Locality

There are people who assert that when left alone, Windows XP will actually create virtual memory pages in close proximity to frequently-used data in the hard disk, like documents. In other words, they claim that Windows XP monitors disk usage, maintains a database of frequently-used files in the hard disk and uses that information to create the paging file based on spatial locality.

With virtual memory pages created close to frequently-used data, this apparently allows shorter seeks between frequently-used data and the paging file. That is the premise behind their theory of letting Windows XP handle the paging file dynamically. However, I don't think this is true at all.

First of all, while Windows XP does monitor disk usage and maintain a database of frequently-used files, only disk defragmenting utilities use that database. The built-in Disk Defragmenter, as well as third-party disk defragmenting utilities, use this database to rearrange the hard disk so that frequently-used data.

But as far as I'm aware, the paging file does not arrange the location of the pages according to this database. From my observations, Windows XP simply uses the nearest available clusters for the dynamic paging file.

In fact, Microsoft states that if you create multiple paging files, Windows XP will favour the partition that is least active. This completely refutes the theory of virtual memory pages being allocated according to spatial locality. Here is a quote from Microsoft's Knowledge Base.

> By design, Windows uses the paging file on the less frequently accessed partition over the paging file on the more heavily accessed boot partition. An internal algorithm is used to determine which paging file to use for virtual memory management.

In any case, it doesn't make sense for Windows XP to create the paging file based on spatial locality to work files like your documents. Once opened, Windows keeps the working copy in the Temp folder, not your paging file.

In addition, let us remember that Windows pre-emptively pages out pageable parts of an application in system

memory. Windows does not directly load data from the hard disk into the paging file. Therefore, creating virtual memory pages close to frequently-used files will not help at all.

Fragmented Vs. Contiguous

Even though Microsoft asserts that the new dynamic virtual memory system does not benefit much from a contiguous paging file, the fact is maintaining a contiguous paging file will definitely improve the paging file's performance.

A contiguous paging file eliminates the need for the hard disk heads to seek all over the platters while accessing the paging file. The following pictures illustrate my points.

This shows a fragmented paging file (brown)

This shows a contiguous paging file (brown)

See how a contiguous paging file differs from a fragmented paging file? Instead of seeking and reading from a continuous block of hard disk space in the case of a contiguous paging file, the hard disk heads have to seek all over the platters to access the clusters allocated to a fragmented paging file.

As a result, a common operating pattern like the following may emerge :-

Fragmented : seek-read-read-seek-read-seek-read-read-read-seek-read-read-seek
Contiguous : seek-read-read-read-read-read-read-seek-read-read-read-read-read

Of course, the amount of time needed to do the seek operation is different from the time needed to read a block of data off the paging file but the logic remains.

A contiguous paging file allows data to be read with minimal amount of seeks. If the number of seeks can be reduced while accessing the paging file, then more data can be read in less time. This is the premise behind a contiguous paging file.

How Do We Create A Contiguous Paging File?

Now that we agree that making the paging file contiguous will greatly improve its performance, let's figure out how to make it contiguous.

Using A Permanent Paging File

Yes, I know. You are all thinking, "Simple! Just make the paging file permanent!"

True, creating a permanent paging file is usually the way to create a contiguous paging file. A permanent paging file ensures that the paging file will remain in one single block. However, creating a permanent paging file does not mean the paging file will automatically become contiguous.

That may have been true in Windows 3.1 but believe it or not, Windows XP does not force the creation of a contiguous paging file when you make the paging file permanent!

When you create a permanent paging file, Windows XP automatically uses the nearest (to the outer tracks) sectors to create the paging file. This creates a permanent but fragmented paging file. Naturally, this

reduces the performance of the paging file.

But that's not the end of the world. To avoid this problem, defragment your hard disk before creating the permanent paging file. That will create a contiguous area for Windows XP to create a permanent paging file.

Using A Dynamic Paging File

But making a permanent paging file is not the only way to create a contiguous paging file. You can also create a contiguous paging file that is also dynamic in nature!

All you need to do is create a separate partition for the paging file. This allows the paging file a contiguous space on the hard disk to freely expand according to usage.

At first glance, the benefits of this method seem obvious. It ensures the paging file is always contiguous and yet have the ability to expand when the need arises. However, this method is really not very desirable when you examine it closely.

The main reason for using a dynamic paging file is to save hard disk space by using it only when there is a need for more virtual memory. But creating a partition to allow the paging file to dynamically resize is really defeating the purpose.

The size of the partition limits the maximum size that the dynamic paging file can expand to and you cannot use the partition to store anything else because that would interfere with its contiguity. If you create a big partition, that wastes hard disk space. If you create a small partition, that limits how big the paging file can expand. Therefore, this method is self-defeating.

What About A Semi-Permanent Paging File?

Although everyone knows about dynamic and permanent paging files, there is a third type of paging file - a semi-permanent paging file.

A semi-permanent paging file theoretically allows you to receive the performance benefits of a contiguous permanent paging file without its main disadvantage - the need to predetermine an optimal size. But what is a semi-permanent paging file?

Well, a semi-permanent paging file is a combination of a permanent paging file and a dynamic paging file. It consists of a permanent part and a dynamic part. The permanent part of this paging file behaves exactly like a permanent paging file. It will not change in size and can thus be moved to the outer tracks of the hard disk.

The dynamic part, however, does not normally appear. In fact, it is only created when the permanent part of the semi-permanent paging file is unable to cope with increased memory requirements.

Once created, it dynamically resizes itself to suit the current paging file requirements. Just like the dynamic paging file, it will use any free space on the hard disk so it will be fragmented.

The Advantages Of A Semi-Permanent Paging File

The semi-permanent paging file offers the advantage of never running out of virtual memory. That means even if the permanent part cannot handle the memory load, the application won't halt with an "Out of memory" error

message. The dynamic part will come into action and provide the extra virtual memory required by the application.

With a permanent paging file, the application will just halt with the error message and you would have to close one or more applications to free up more memory. However, this is only true for older versions of Windows.

Newer iterations of Windows like Windows XP do not have a true permanent paging file. Even if you set a permanent paging file, Windows XP will automatically generate more virtual memory when it runs out of memory; by adding a dynamic component to the permanent paging file. In short, when you create a "permanent" paging file in Windows XP, you are actually creating a semi-permanent paging file.

The advantage of creating your own semi-permanent paging file, instead of a "permanent" paging file in Windows XP is that you get to avoid the warning message that appears whenever Windows XP runs out of memory and has to create more virtual memory by adding a dynamic component to the permanent paging file.

The Disdvantages Of A Semi-Permanent Paging File

Unfortunately, a semi-permanent paging file is a double-edged sword. With a dynamic component, it is inevitable that a dynamic paging file's disadvantages would also be applicable to it.

As mentioned earlier, the dynamic part will use any available space on the hard disk. This inevitably means the dynamic part of the semi-permanent paging file will always be fragmented. Naturally, the performance of the paging file deteriorates whenever the dynamic part comes into action.

But just how bad could be the deterioration be? Let's take a look at the disk map below which shows a semi-permanent paging file with both permanent and dynamic components in brown :-

This shows a semi-permanent paging file (brown)

You will notice that the paging file is split into two parts. The permanent part is at the outer tracks of the hard disk in one contiguous block. The lower, fragmented blocks of paging file are the dynamic part of the semi-permanent paging file. As the paging file requirement exceeds what the permanent part can provide, the dynamic part of the semi-permanent paging file will dynamically convert available hard disk space (which is usually on the inner tracks on the hard disk) into virtual memory.

Because the paging file's two components are at opposite ends of the hard disk, the hard disk heads will have to seek up and down the platters while servicing the paging file! Needless to say, that greatly degrades the performance of the paging file. The head seeks required to service a dynamic paging file are already bad enough. The amount of head seeks required to service both the permanent part and the fragmented dynamic part will definitely put a big dent on the paging file's performance.

Permanent Or Semi-Permanent?

Performance-wise, both a permanent and a semi-permanent paging file will perform equally, if the virtual memory requirement does not exceed what the permanent component of the semi-permanent paging file can provide. As the dynamic part comes into play, the semi-permanent paging file gradually loses its performance advantage over the dynamic paging file. Eventually, it may even become slower than a dynamic paging file.

The only way around this is to ensure that the permanent part of the semi-permanent paging file is enough to meet your usual virtual memory requirements. Do not look at the semi-permanent paging file as a way to save hard disk space. Instead, think of it as a permanent paging file with a backup capacity for dynamic expansion

in emergencies!

Hard disk space is no longer that much of a premium as it was back in the old days. With desktop hard disks approaching half a terabyte in size, allocating a few hundred megabytes or even a gigabyte or so for the paging file isn't going to break anyone's heart. The performance of the paging file, especially in systems with very little RAM or for people who multitask a lot, is definitely more important than saving a few hundred megabytes of hard disk space.

Is Writing And Rewriting To The Same Area Dangerous?

Creating a permanent or semi-permanent paging file inevitably causes numerous writes and rewrites of information in the same fixed area of the hard disk platters. Compared to other areas of the hard disk, the space allocated to the paging file will be the area where data is most often written, deleted and replaced with newer data.

Some users have expressed concern over this fact. Will the platter media in that area get worn out after continuous use? Like the magnetic cassettes that we used to record our favourite songs? Will bad sectors form in that area like the floppy disks that have been written to once too often?

Well, unlike magnetic cassettes or floppy disks, there is actually no contact between the hard disk read-write heads with the platters. The read-write heads actually fly over the platters on a thin cushion of air. In fact, at the high speed that the platters are spinning at, any contact between a read-write head with a platter would have resulted in a head crash, with disastrous consequences.

Therefore, friction isn't the concern here. What about the effect of changing the magnetic properties of the media during the write process? Will the magnetic properties of the media deteriorate after too many of such changes? Or in the context of this article, will creating a permanent paging file damage the drive in the long run and reduce its MTBF (Mean Time Between Failures)?

To obtain a definitive answer to these questions, I contacted IBM and Seagate. Let's see what their technical experts have to say.

Seagate

This should not hurt the drive at all. As you are aware, the heads are actually suspended above the platters on an air bearing, so there is no direct contact with the media. As far as the recording and re-recording of the same tracks, also no problems. What we are dealing with here in order to write the data is simply moving the magnetic domain one way or the other, no wear involved.

Regards,

Bob
Seagate Tech Support

IBM

Remember, the heads truly fly above the media. The wear and tear factor only becomes an issue for bearings (heat) and physical damage to the media if the drive is shocked during operation. Performance is best at the outer tracks of the drive, so any recurring access directed there will benefit you in performance. Writing and rewriting data to a drive is good in that it remagnetizes (refreshes) the area every time it is written.

To answer your question: Your swap file will not affect the MTBF of your drive.

Don Gardner
IBM Hard Disk Technical Support/SIT Lab

So, Are Multiple Writes To The Same Area Good?

Well, it appears to be so. From what Don Gardner said, I gather that the signal carried by the media weakens with time and rewriting it refreshes and strengthens the signal strength of the data carried by the media.

I guess that pretty much answers our questions. Creating a permanent or semi-permanent paging file won't harm your drive. In fact, it might even be good for your data!

Creating A Permanent Paging File In Windows 9x

Luckily, Microsoft gave us a relatively painless way to create a permanent paging file though the proper directions were not included. Fear not however. This is what guides like this are for.

First, open up System Properties, either through the Control Panel or by right-clicking on the My Computer icon and selecting Properties. Once in System Properties, click on the Performance tab and you'll see the following picture :-

Right at the bottom, you'll see a Virtual Memory... button. Click on it to get the following screen :-

By default, it is set to Let Windows manage my virtual memory settings. (Recommended). Ignore the Recommended label and select Let me specify my own virtual memory settings.

Now, you will be allowed to choose the partition you wish to place the paging file in. We will touch on this later.

Next up is the minimum and maximum values for the paging file. To create a permanent paging file, both values must be the same. You would think that Microsoft could at least post a notice about that.

Please note that Windows 95/98 will not automatically add a dynamic component to a permanent paging file. If you run out of memory with a permanent paging file, it will halt the application and generate the "Out of memory" error message.

Naturally, you will have to decide on a size for the paging file. We will be discussing this later in the guide but in this example, we will use an arbitrary value of 150MB. Once you set the two values, click on OK and then let Windows 95/98 reboot the system. A permanent paging file will be created on your hard disk.

For the curious, do not click on Disable virtual memory. (Not recommended) because that will force Windows 95/98 to use only physical RAM.

Creating A Semi-Permanent Paging File In Windows 9x

Creating a semi-permanent paging file is rather similar to creating a permanent paging file.

First, open up System Properties, either through the Control Panel or by right-clicking on the My Computer icon and selecting Properties.

Once in System Properties, click on the Performance tab and you'll see the following picture :-

Right at the bottom, you'll see a Virtual Memory... button. Click on it to get the following screen :-

By default, it is set to Let Windows manage my virtual memory settings. (Recommended). Ignore the Recommended label and select Let me specify my own virtual memory settings.

Now, you will be allowed to choose the partition you wish to place the paging file in. We will touch on this later.

To create a semi-permanent paging file, you will need to set both the minimum and maximum values. They must not be the same. If they are the same values, then the paging file becomes a permanent paging file.

The minimum value determines the size of the permanent component of the semi-permanent paging file. The maximum value determines the maximum size of the paging file (both permanent and dynamic components) and thus limits how much the dynamic component can expand.

In the example above, Windows 98 will create a permanent paging file of 150MB when it starts up. But if the paging file cannot meet the memory demands of the computer, it will dynamically expand the paging file, up to a maximum of 6692MB.

It is highly recommended that you create a large permanent component that will meet all of your usual memory needs. Use the dynamic component as a backup for emergencies.

Once you set the two values, click on OK and then let Windows 95/98 reboot the system. A permanent paging file will be created on your hard disk. Please note that the dynamic component of the paging file will only become active after the system's virtual memory requirements exceed the minimum value.

For the curious, do not click on Disable virtual memory. (Not recommended) because that will force Windows 95/98 to use only physical RAM.

Creating A Permanent Paging File In Windows 2000

In Windows 2000, it takes a little bit more digging to get where you want.

First, open up System Properties, either through the Control Panel or by right-clicking on the My Computer icon and selecting Properties.

Once in System Properties, click on the Advanced tab. There will be three options. Click on Performance Options... and you'll see the following picture :-

The second section you see is titled Virtual Memory. Under it, there's a Change... button. Click on it to get the following screen :-

By default, there won't be any values set for both the Initial size (MB) and the Maximum size (MB) options.

You can select the partition you wish to place the paging file in by clicking on the list of partitions shown on the screen. Again, the selection of partition will be discussed in detail later in this article.

To create a permanent paging file, both values for the Initial size and the Maximum size must be the same.

Please note that Windows 2000 will not automatically add a dynamic component to a permanent paging file. If you run out of memory with a permanent paging file, it will halt the application and generate the "Out of memory" error message.

Naturally, you will have to decide on a size for the paging file. We will be discussing this later in this article but for now, we will use an arbitrary value of 150MB. Once you set the two values, click on OK and then let Windows 2000 reboot the system. A permanent paging file will be created on your hard disk.

You will note that Windows 2000 does not allow a paging file size of less than 2MB.

Creating A Semi-Permanent Paging File In Windows 2000

Again, it's almost similar to creating a permanent paging file.

First, open up System Properties, either through the Control Panel or by right-clicking on the My Computer icon and selecting Properties.

Once in System Properties, click on the Advanced tab. There will be three options. Click on Performance Options... and you'll see the following picture :-

The second section you see is titled Virtual Memory. Under it, there's a Change... button. Click on it to get the following screen :-

By default, there won't be any values set for both the Initial size (MB) and the Maximum size (MB) options.

You can select the logical drive you wish to place the paging file in by clicking on the list of logical drives shown on the screen. Again, the selection of logical drives will be discussed in detail later in this article.

To create a semi-permanent paging file, you will need to set both the minimum and maximum values. They must not be the same. If they are the same values, then the paging file becomes a permanent paging file.

The minimum value determines the size of the permanent component of the semi-permanent paging file. The maximum value determines the maximum size of the paging file (both permanent and dynamic components) and thus limits how much the dynamic component can expand.

In the example above, Windows 2000 will create a permanent paging file of 150MB when it starts up. But if the paging file cannot meet the memory demands of the computer, it will dynamically expand the paging file, up to a maximum of 1422MB.

It is highly recommended that you create a large permanent component that will meet all of your usual memory needs. Use the dynamic component as a backup for emergencies.

Once you set the two values, click on OK and then let Windows 2000 reboot the system. A permanent paging file will be created on your hard disk. Please note that the dynamic component of the paging file will only become active after the system's virtual memory requirements exceed the minimum value.

You will note that Windows 2000 does not allow a paging file size of less than 2MB.

Creating A Permanent Paging File In Windows XP

Like in Windows 2000, it takes a little digging in Windows XP to get where you want.

First, open up System Properties, either through the Control Panel or by right-clicking on the My Computer icon and selecting Properties.

Once in System Properties, click on the Advanced tab. There will be three sections.

Click on Settings in the Performance section and the Performance Options screen will pop up. Click on the Advanced tab and you'll see the following picture :-

The second section you see is titled Virtual memory. Under it, there's a Change button. Click on it to get the following screen :-

You can select the logical drive you wish to place the paging file in by clicking on the list of logical drives shown on the screen. Again, the selection of logical drives will be discussed in detail later in this article.

To create a permanent paging file, both values for the Initial size and the Maximum size must be the same.

Please note that Windows XP will dynamically expand the paging file when you run out of memory, even if you create a permanent paging file. When this happens, you will get an error message telling you that Windows XP is trying to expand the paging file to create more virtual memory.

In this example, we are using an arbitrary value of 512MB. Once you set the two values, click on OK and then let Windows XP reboot the system. A permanent paging file will be created on your hard disk.

You will note that Windows XP does not allow a paging file size of less than 2MB.

Creating A Semi-Permanent Paging File In Windows XP

Creating a semi-permanent paging file is rather similar to creating a permanent paging file.

First, open up System Properties, either through the Control Panel or by right-clicking on the My Computer icon and selecting Properties.

Once in System Properties, click on the Advanced tab. There will be three sections.

Click on Settings in the Performance section and the Performance Options screen will pop up. Click on the Advanced tab and you'll see the following picture :-

The second section you see is titled Virtual memory. Under it, there's a Change button. Click on it to get the following screen :-

You can select the partition you wish to place the paging file in by clicking on the list of partitions shown on the screen. Again, the selection of partition will be discussed in detail later in this article.

To create a semi-permanent paging file, you will need to set both the minimum and maximum values. They must not be the same. If they are the same values, then the paging file becomes a permanent paging file.

The minimum value determines the size of the permanent component of the semi-permanent paging file. The maximum value determines the maximum size of the paging file (both permanent and dynamic components) and thus limits how much the dynamic component can expand.

In the example above, Windows XP will create a permanent paging file of 512MB when it starts up. But if the paging file cannot meet the memory demands of the computer, it will dynamically expand the paging file, up to a maximum of 768MB.

It is highly recommended that you create a large permanent component that will meet all of your usual memory needs. Use the dynamic component as a backup for emergencies.

Once you set the two values, click on OK and then let Windows XP reboot the system. A permanent paging file will be created on your hard disk. Please note that the dynamic component of the paging file will only become active after the system's virtual memory requirements exceed the minimum value.

You will note that Windows XP does not allow a paging file size of less than 2MB.

Making The Paging File Contiguous

After creating a permanent or semi-permanent paging file, check and make sure it is contiguous. You can ensure it is contiguous by defragmenting the hard disk before you creating the permanent or semi-permanent paging file. However, that does not always work.

In such cases, you will need to defragment the paging file after it is created. Unfortunately, Windows XP's Defrag utility does not have the ability to defragment the paging file. You will have to use a third-party defragmentation utility to do this. I will use Diskeeper as an example.

Windows NT, 2000 and XP does not allow the paging file to be defragmented while it is in used. Therefore, you must set Diskeeper to move the paging file during the next reboot.

Run Diskeeper and click on Change your settings to expand its menu. You will see the screen below.

Look for and click on Set a boot-time defragmentation. That will display this screen.

Now, select the partition where the paging file resides and tick the checkbox of Defragment the paging file option. The option will be grayed out if there is no paging file in that partition.

Then click OK and reboot the computer. Diskeeper will load up during the boot process and defragment the paging file.

Once Diskeeper has completed its operation, Windows XP will boot up and start using the newly optimized paging file that is contiguous.

Please note that Diskeeper requires a certain amount of free space to defragment the paging file. If you do not have the necessary amount of free space in that partition, then Diskeeper may not defragment the paging file.

Wanna Do It For Free?

You can easily do this for free. Just download a trial copy of Diskeeper!

Moving The Paging File To The Outer Tracks

Moving the paging file to the outer tracks is a powerful way of increasing paging file performance. In fact, it will give the paging file a bigger boost in performance than just making it contiguous. Why is that?

Check out this transfer rate graph of a hard disk :-

It shows pretty clearly the transfer rate of a hard disk is highest on the outer tracks and lowest on the inner tracks. In this case, the transfer rate of the inner tracks is only about half the transfer rate of the outer tracks.

The areal density of a hard disk's platters and its spin rate are constant. But the linear velocity at each point of the platter isn't constant. Therefore, the performance of the paging file depends on where it is located on the hard disk.

The time taken for the hard disk head to read from point A to point B is exactly the same as the time taken for the head to read from C to D. But because the areal density of the platter is constant, a lot more data can be read from the outer tracks than from the inner tracks, in the same amount of time.

Now that the outer tracks have been proven to be the fastest area on a hard disk, we can use that to our advantage. By moving the paging file to the outer tracks, we give the paging file a major boost in performance.

As you can see from the example above, the transfer rate at the outer tracks are about 59MB/s while the central and inner tracks have transfer rates of about 49MB/s and 30MB/s respectively. Moving the paging file from the inner tracks to the outer tracks will almost double its performance! Even moving the paging file from the central tracks to the outer tracks will give the paging file a transfer rate boost of 20%.

But please note that this method must be used in conjunction with a permanent paging file. This is because the paging file cannot be moved to the outer tracks of the hard disk unless it is a permanent paging file.

How Do We Move The Paging File To The Outer Tracks?

Before you can move the paging file to the outer tracks, you must first make the paging file permanent. Follow the steps outlined in the previous pages. Once you have a permanent paging file, you can use your favourite hard disk defragmentation utility to move the paging file to the outer tracks.

Unfortunately, Windows XP's Defrag utility does not have the ability to move paging file to the outer tracks. You will have to use a third-party defragmentation utility to move the paging file to the outer tracks. I will use Diskeeper as an example.

Windows NT, 2000 and XP does not allow the paging file to be moved while it is in used. Therefore, you must set Diskeeper to move the paging file during the next reboot.

Run Diskeeper and click on Change your settings to expand its menu. You will see the screen below.

Look for and click on Set a boot-time defragmentation. That will display this screen.

Now, select the partition where the paging file resides and tick the checkbox of Defragment the paging file option. The option will be grayed out if there is no paging file in that partition.

Then click OK and reboot the computer. Diskeeper will load up during the boot process and defragment the paging file. It will also move the paging file to the outer tracks.

Once Diskeeper has completed its operation, Windows XP will boot up and start using the newly optimized paging file that is not only contiguous but also located in the outer tracks of the hard disk! Your paging file will now show a marked boost in performance.

Please note that you cannot actually force Diskeeper to move the paging file right up to the outermost tracks. Diskeeper has an internal algorithm that determines which files are best placed in the outermost tracks for optimal performance.

In addition, Diskeeper requires a certain amount of free space to defragment and move the paging file. If you do not have the necessary amount of free space in that partition, then Diskeeper may not defragment the paging file or move it to the outer tracks.

Creating A Huge Paging File

Because games and applications often list a minimum paging file size, many people equate the size of the paging file with performance, just like they would with anatomy. But at least in the first case, that's not true.

What does a bigger paging file get you? Well, it gives you the ability to run more memory-intensive programs concurrently. But does a larger paging file make virtual memory faster or better? Unfortunately, the answer is no.

Why Not?

First of all, creating a large amount of virtual memory doesn't mean the operating system will use it all. Although Windows will pre-emptively page out parts of idle applications, there are limits to how much it can page out for each application. Therefore, creating an excessively large paging file will just waste hard disk space.

Second, if you ever move the paging file to the outer tracks of the hard disk, an excessively large paging file will take up outer track space that could have been used to store system or application files. Look at these two pictures :-

Hard disk with a 2GB paging file (brown)

Hard disk with a 600MB paging file (brown)

The first one has a huge 2GB paging file while the second has a smaller 600MB paging file. For many systems, 600MB of virtual memory is more than enough to multitask 7 or 8 applications at the same time or run the most memory-intensive 3D games out there. So, anything more is just taking space.

The extra space taken up by an excessively large paging file on the outer tracks could have been used to store system or application files for faster access. The amount of space regained from using a smaller page file can be seen as a red block in the second picture. You can bet on a faster loading time for Windows and other applications if you limit the size of your paging file.

Therefore, the trick here is to gauge the maximum size of the paging file that you will ever need. This way, you will not create an excessively large paging file that wastes hard disk space and takes up the precious space on the outer tracks away from the system and application files.

How Large Should The Paging File Be?

That's a question that has bugged many users. Since the good old days of DOS and Windows 3.1, many users have staunchly adhered to an old rule of the thumb that the swapfile should be 2.5 x the amount of RAM.

In fact, whenever I visit other forums, I still notice many people quoting this old "rule". The question is - is this rule still applicable for today's systems and operating systems? Unfortunately, it's a big NO!

Why Not 2.5 x RAM?

Back in the Windows 3.1 days, computers only came with 4MB or 8MB of RAM. 16MB of RAM was considered a luxury in those days. I remember running Windows 3.1 on an Intel i386SX-16 machine with just 4MB of RAM!

Because RAM in those days was horrendously expensive and only a limited amount of it was available in most systems, a relatively large swapfile was needed. A swapfile size of 2.5 times the system RAM wasn't a lot, considering the fact that most systems came with only 4MB or 8MB of RAM. That would only amount to a swapfile size of 10MB to 20MB, which enabled most systems to run Windows 3.1 applications comfortably.

But today, most computers come with at least 512MB of RAM and many have 1GB of RAM! If the 2.5X rule was applied, that would result in "optimal" paging file sizes of 1.28GB to 2.5GB! That doesn't make sense at all.

The purpose of buying more memory is to prevent the system from using the slower virtual memory. The more memory you buy, the less you need to use virtual memory. It doesn't make sense to increase the paging file size every time you increase the amount of RAM in your system!

Imagine if you have follow the rule when you upgrade to 2GB of RAM in the future... You would have to create a 5GB paging file! That's ridiculous.

The amount of hard disk space you dedicate to a paging file should depend on the amount of RAM you need to use, NOT the amount of RAM you have. The 2.5 x system RAM rule was flawed from the beginning and it is certainly not applicable today.

Do not use the 2.5 x system RAM rule to determine the size of your paging file. Instead, you should first gauge how much virtual memory is actually needed by the system during the heaviest memory load. Then use your finding to set the most appropriate paging file size for your system.

But You Require A Huge Paging File For A Memory Dump!

There are people who actually believe in increasing the size of the paging file following an increase in system memory. That certainly goes against what we have been recommending, doesn't it? The reason is simple.

Whenever Windows crashes, it first writes the memory contents to the paging file. After the computer is restarted, Windows will create a memory dump file using the memory contents stored in the paging file. This memory dump file is used to analyze the cause of the crash.

However, for a complete memory dump to created, the paging file size should be large enough to store all the contents of the system memory. That's why the paging file size has to meet this equation :-

Paging file size = Physical memory in the system + 1MB

So, if you have 1024MB of memory, the paging file size should be 1025MB in size for a complete memory dump to be created successfully.

However, this does not mean you should increase the size of your paging file according to the amount of system memory. Why not? Let's find out.

Why Not?

First of all, there is no need to create a complete memory dump. Windows supports three different kinds of memory dumps. Here is a summary of information from Microsoft's Knowledge Base.
Type Of Memory Dump

Description

Size
Small

* Small memory dump files contain the least information, but consume the least disk space, 64 kilobytes (KB).
* Unlike kernel and complete memory dump files; Windows XP stores small memory dump files in the systemroot\Minidump folder, instead of using the systemroot\Memory.dmp file name.
* Windows XP always create a small memory dump file when a Stop error occurs, even when you choose the kernel or complete memory dump file options.
* One of the services that use small memory dump files is the Error Reporting service. The Error Reporting service reads the contents of a small memory dump file to help diagnose problems that cause Stop errors.

64KB
Kernel

* This is an intermediate size dump file that records only kernel-level memory and can occupy several megabytes (MB) of disk space.
* When a Stop error occurs, Windows XP Professional saves a kernel memory dump file to a file named systemroot\Memory.dmp and create a small memory dump file in the systemroot\Minidump folder.
* You cannot exactly predict the size of a kernel memory dump file because this depends on the amount of kernel-mode memory allocated by the operating system and drivers present on the machine when the Stop error occurred.

About 1/3 of system memory
Complete

* A complete memory dump file contains the entire contents of physical memory when the Stop error occurred.
* The file size is equal to the amount of physical memory installed plus 1 MB.
* When a Stop error occurs, the operating system saves a complete memory dump file to a file named systemroot\Memory.dmp and creates a small memory dump file in the systemroot\Minidump folder.

System memory + 1MB

Although you may think that it is always good to create a complete dump file. However, that is not true. Even Microsoft recommends creating a kernel memory dump, instead of a complete memory dump. Why? I'll quote them :-

> For most purposes, a kernel memory dump file is the most useful kind of file for troubleshooting Stop messages. It contains more information than the small memory dump file and is significantly smaller than the complete memory dump file. It omits only those portions of memory that are unlikely to have been involved in the problem.

In addition, a kernel memory dump will require the paging file to be only about 1/3 of the system memory. It will also require the same amount of free hard disk space.

Even if you wish to create a complete memory dump, there is still no need to create a large paging file. Even if you restrict your paging file to, for example, 500MB; Windows XP will automatically expand the paging file to store the memory dump BEFORE it is written out to disk on the next reboot.

Therefore, I consider it to be a real waste of hard disk space if you have 2GB of memory and yet create a 2GB paging file, just so Windows XP can write an enormous memory dump the next time it crashes.

How Much Virtual Memory Do I Need?

No one can tell you how much hard disk space you need to allocate to a permanent paging file because every system is different and everyone uses his/her system differently.

If you create a permanent paging file that is too small, then Windows will continuously create more virtual memory via a dynamic extension to the permanent paging file. This reduces the paging file's performance

If you create a permanent paging file that is too large, you are only wasting hard disk space, especially on the outer tracks.

So, the best method would be to accurately gauge how much virtual memory you actually need. This allows you to create a permanent paging file with the appropriate size. To do that, you need to monitor your paging file usage. Let's see how you can do that.

Finding Out In Windows 9x

Give your system a clean boot and once you are in Windows 95/98, load System Monitor. You can get to it via Start Menu > Programs > Accessories > System Tools. You will see this screen :-

Go to the Edit menu and click on Add Item...

In the next screen, select the Memory Manager category and add Swapfile in use. Click OK and you will see this screen :-

Now, you can monitor the size of your paging file. Start up and run all the applications that you usually use at the same time. Load several documents and work files. Play around with them and check the peak value for the paging file.

Then play several of the most memory-intensive games you have. 3D games with large textures are good ones to test. At all times, record down the highest value for the paging file size that System Monitor reports.

Once you are done, select the highest value that has been recorded for the paging file size and round it up to the nearest 100MB. For example, if the biggest size your paging file ever went during the tests was 619MB, then 700MB is the ideal size for your paging file.

But always make sure you add at least 40-50MB as a cushion against future memory-guzzling applications or games. For example, if the largest size your paging file expanded to during your tests was 684MB, then 750MB would be an ideal size for your paging file.

How Much Virtual Memory Do I Need?

Finding Out In Windows XP

Finding your optimal paging file size in Windows XP is much easier.

Just give your system a clean boot. Once you are in Windows XP, run Task Manager . You can get to it by right-clicking on the taskbar and selecting Task Manager. You can also access it through the keyboard shortcut

of Ctrl-Alt-Del.

After you load Task Manager, click on the Performance tab. You will see this screen :-

Now, you can monitor the size of your paging file. Start up and run all the applications that you usually use at the same time. Load several documents and work files. Play around with them and check the peak value for the paging file.

Then play several of the most memory-intensive games you have. 3D games with large textures are good ones to test. At all times, record down the highest value for the paging file size that System Monitor reports.

Once you are done, select the highest value that has been recorded for the paging file size and round it up to the nearest 100MB. For example, if the biggest size your paging file ever went during the tests was 619MB, then 700MB is the ideal size for your paging file.

But always make sure you add at least 40-50MB as a cushion against future memory-guzzling applications or games. For example, if the largest size your paging file expanded to during your tests was 684MB, then 750MB would be an ideal size for your paging file.

Moving The Paging File To A Different Partition

Another popular technique proposed by many tweakers suggests moving a temporary paging file from the default first partition to a separate, dedicated partition.

The reasons for this technique are ostensibly two-fold :-

+ to reduce fragmentation of the first partition
+ to ensure that the paging file will remain contiguous even though it is a temporary paging file

This idea looks good because it enables users of temporary paging files to keep their primary partition neat and the paging file contiguous for a speed boost.

However, many users of this technique failed to take into account several things. Let's see what they are.

Cylinders And Partitions

First of all, let's take a look at a hard disk cylinder. A cylinder consists of the same tracks on all the platters in the hard disk.

The first cylinder, nominally called cylinder 0, is coloured in bright green. It is the outer most cylinder and consists of the first track of all the platters in the hard disk. Such groups of tracks have a cylindrical look, hence the name. Cylinder n (in red) is the last cylinder of the hard disk, where n can be any integer.

Partitions are constructed using full cylinders. The first one starts at cylinder 0 and go out to where you specify. The next one starts on the following cylinder, and so on. If you try to create a partition with an end that falls in the middle of the cylinder, FDISK or similar utilities will round it up so that the partition occupies the entire cylinder, instead of a partial cylinder.

Needless to say, the first partition will always start with the first track of every platter. In other words, the first partition will always be the fastest partition in the hard disk, followed by the second partition and so on. Therefore, if you create a second partition and dump the paging file there, you will actually be

moving it to a slower part of your hard disk!

As you can see, while the temporary paging file will be remain contiguous using this technique, the transfer of the paging file from the outer tracks to the inner tracks of the hard disk will inevitably reduce its performance.

Need More Reasons?

Creating a dedicated partition for the dynamic paging file also means tying up hard disk space and inviting inflexibility.

Users of FDISK will find it impossible to change the size of the paging file partition when they need to do so. In fact, they will have to remove at least two partitions to create a larger one. If they only have one primary and a secondary paging file partition, then they will have to remove both and recreate two new partitions.

Users of special utilities like Partition Magic will have an easier time as they can easily adjust the sizes of the partitions. But in the end, this method is counter-productive because for all your trouble, you have just slowed down your paging file and orphaned off a portion of your hard disk for the dedicated partition.

The main reason for using a temporary paging file is actually to save hard disk space. Users of a temporary paging file avoid tying up large amounts of hard disk space in a permanent swapfile.

However, this method actually requires you to set aside a large amount of hard disk space and worse, place cordon off this space in an inflexible partition. If you can afford to allocate space for this dedicated partition, you would be better off using the space for a permanent paging file.

In my opinion, this technique is a waste of time and needlessly endangers your data. Messing around with FDISK and partitions can be heartbreakingly exciting, if you catch my drift.

More Partitions = Data Parachute?

Some users advocate using multiple partitions for safety reasons. Their opinion is that in the event of a hard disk crash, corruption to the boot sector or FAT (File Allocation Tables), only the primary partition will be lost, leaving precious data safe in the other partitions.

Unfortunately, from my experience involving hard disk crashes, every partition was inevitably wiped out. When a hard disk head crashes with a platter, I seriously doubt it would politely avoid scoring through the media that has been allocated to other partitions.

Russ Johnson, a Product Support Engineer from Symantec Corporation has this to say, "It's not a substitute for a good backup, but it may save you from having to restore all of your data from a backup. However, if your first partition is taken out, more than likely the whole drive will be lost. The first partition is also the location of the Master Boot Record and the partition table."

Now, I agree that storing your data on a different partition is actually a good practice. It can save your data if the first partition gets corrupted due to a soft error. For example, even if the FAT of one of the partitions gets corrupted, data on the other partitions will still be safe.

So, if data integrity (as well as disk management) is important to you, you should consider using multiple partitions. However, this does not mean you should move the paging file to a different partition... oh no...

When the paging file is permanent, tweakers who advocate moving paging files around will tell you to move your partition to a second hard disk. Why?

As the theory goes, this allows your system to access both the paging file on the second hard disk and data on the first hard disk concurrently. This theoretically improves performance a lot! But does it really work?

Well, it depends.

Hard Disk, NOT Partition!

Many people get confused by drive letters. They assume that moving the paging file from drive C: to drive D: is the same as moving it to another hard disk. However, this is not true.

The operating system does not bother with physical drives. It is only interested in logical drives. By this, we mean properly-formatted partitions that can be accessed by the operating system.

To the operating system, partitions appear as physically-separate hard disks although they may reside on the same hard disk. If you partition your hard disk into three different partitions, your operating system will identify them as three logical drives (Drive C:, Drive D: and Drive E:). But they are still physically on the same hard disk!

Therefore, if you merely move the paging file to a different logical drive, you could be doing nothing more moving it to a different partition. So, please check and make sure you are moving it to a physically-separate hard disk. Preferably, it should be the first partition in that hard disk.

Parallel-ATA

Many tweakers forget one thing when they move their paging files to the second hard disk - only one PATA (Parallel-ATA) device can be active at any one time on the same IDE channel.

Most users slave the second hard disk to the first hard disk on the primary IDE channel and put the removable media drives (CD writers, DVD-ROM, etc.) on the secondary IDE channel. That is theoretically sound practice but it actually negates the purpose of moving the paging file off the primary hard disk!

Because both hard disks are on the same IDE channel, they can't be active at the same time. So, there is no way data can be read from both hard disks at the same time. In fact, because the secondary hard disk is often slower and smaller than the primary hard disk, the performance of the paging file on the second hard disk will actually be worse off.

So How Do We Make It Work?

The only way for this method to work is to put the first and second hard disks on separate IDE channels. That means the first hard disk gets hooked up to the primary IDE channel and second hard disk gets the secondary IDE channel. This allows both IDE channels can be active at the same time, delivering data from both hard disks concurrently.

In addition, the second hard disk needs to be at least half as fast as the primary hard disk. This allows the paging file on the second hard disk to be at least as fast as a paging file on the first hard disk. Otherwise, the performance advantage accessing the paging file concurrently on a second hard disk will be negated by the slower performance of the second hard disk.

Remember, if the first hard disk can serve data to and from both the application in use and the paging file faster than the second hard disk can access the paging file alone, then it is pointless to maintain a paging file on the second hard disk.

But if the second hard disk is more than half as fast as the first hard disk, then it would be advantageous to move the paging file there because the paging file can then be accessed concurrently with data on the first hard disk. In addition, the valuable outer tracks on the first hard disk will be freed up for use by the operating system.

Other Considerations

The trouble with such a setup is that most motherboards usually with only two IDE channels.

If you slave your DVD writer to the first hard disk (on the primary IDE channel), then you may have trouble writing data from the first hard disk to a DVD. This is because the IDE channel has to interleave its operations between the first hard disk and the DVD writer.

You won't have any trouble writing data from devices on the second IDE channel to the DVD writer though. This is because the DVD writer is on the first IDE channel and can thus be accessed concurrently with the devices on the second IDE channel.

However, if you slave your DVD writer to the second hard disk (on the secondary IDE channel), then you may have problems with games running off CDs or DVDs in that drive. Of course, this time you won't have any trouble writing data from devices on the first IDE channel to the DVD writer!

Either way, you will face performance compromises. It is a great idea but implementation is not quite as simple as you might think. The key to making this work is to be aware of such considerations and plan your setup accordingly.

But if your motherboard comes with enough IDE channels to give each device its own channel, then the way is clear - hook the second hard disk to a separate IDE channel and move the paging file there!

Serial-ATA

The beauty of Serial-ATA is that each device is given its own channel. This completely circumvents the issues that Parallel-ATA drives have with sharing the same IDE channel.

So, if you are only using Serial-ATA devices, you can immediately move the paging file to the second hard disk for a big speed boost. Just make sure you move it to a partition that is on the second hard disk, not the second partition in your first hard disk.

How Do I Move The Paging File In Windows 9x?

If you want to move your paging file to a different partition or drive, first open up System Properties, either through the Control Panel or by right-clicking on My Computer and selecting Properties.

Once in System Properties, click on the Performance tab and you will see the following picture :-

Right at the bottom, you'll see a Virtual Memory... button. Click on it to get the following screen :-

Select Let me specify my own virtual memory settings. This allows you to choose the logical drive in which you would like to place the paging file.

Click on the pull-down list. It will show you all the available partitions and hard disks in your system. Select the logical drive where you want the paging file to be.

Then set the minimum and maximum paging file sizes and click OK. After rebooting, your paging file will be established in the logical drive you selected.

Again, please remember that each logical drive represents a partition, not a physical drive. So, if you want to move your paging file to a separate hard disk, select a logical drive that resides on that hard disk. Preferably, it should be the first partition in the other hard disk (which should be on its own IDE channel).

How Do I Move The Paging File In Windows 2000?

First, open up System Properties, either through the Control Panel or by right-clicking on My Computer and selecting Properties.

Once in System Properties, click on the Advanced tab. There will be three options. Click on Performance Options... and you will see the following picture :-

The second section you see is titled Virtual Memory. Under it, there is a Change... button. Click on it to get the following screen :-

This is where you manage Windows 2000's paging file settings.

Just scroll through the selection of logical drives available. Click on the logical drive that you want to place the paging file. Then set the initial and maximum paging file sizes and click Set.

To remove the paging file from the default location in the first logical drive, select drive C: and set both initial and maximum sizes to 0 (zero). Then click Set.

After you are done, just click OK and allow Windows 2000 to reboot your computer. After rebooting, your paging file will be established in the logical drive you selected.

Again, please remember that each logical drive represents a partition, not a physical drive. So, if you want to move your paging file to a separate hard disk, select a logical drive that resides on that hard disk. Preferably, it should be the first partition in the other hard disk (which should be on its own IDE channel).

How Do I Move The Paging File In Windows XP?

First, open up System Properties, either through the Control Panel or by right-clicking on the My Computer icon and selecting Properties.

Once in System Properties, click on the Advanced tab. There will be three sections.

Click on Settings in the Performance section and the Performance Options screen will pop up. Click on the Advanced tab and you'll see the following picture :-

The second section you see is titled Virtual memory. Under it, there's a Change button. Click on it to get the following screen :-

This is where you manage Windows XP's paging file settings.

You can select the logical drive you wish to place the paging file in by clicking on the list of logical drives shown on the screen.

Just scroll through the selection of logical drives available. Click on the logical drive that you want to place the paging file. Then set the initial and maximum paging file sizes and click Set.

To remove the paging file from the default location in the first logical drive, select drive C: and select No paging file. Then click Set.

After you are done, just click OK and allow Windows XP to reboot your computer. After rebooting, your paging file will be established in the logical drive you selected.

Again, please remember that each logical drive represents a partition, not a physical drive. So, if you want to move your paging file to a separate hard disk, select a logical drive that resides on that hard disk. Preferably, it should be the first partition in the other hard disk (which should be on its own IDE channel).

Multiple Hard Disks

With hard disk prices dropping and multiple hard disks becoming common, this introduces two interesting possibilities :-

+ multiple paging files
+ moving the paging file to a RAID array

Both methods appear to offer better paging file performance. But do they really offer better performance? Let's find out...

Multiple Paging Files

With multiple hard disks in the same system, you can actually split the paging file into multiple paging files!

Instead of just moving the paging file from one hard disk to another, you can actually place a paging file on each and every hard disk in the system. And if each hard disk has its own IDE channel, having multiple paging files will greatly increase its performance.

Because each hard disk with its own channel can be accessed concurrently with the other hard disks in the same system, multiple paging files will allow the computer to access all of them simultaneously. Needless to say, this greatly increases its read and write performance.

However, it is still recommended that you do not place the paging file in the primary hard disk. Leave the outer tracks there for the operating system to use. This will also free up the first hard disk for the operating system's use, instead of sharing it with the paging file.

So, if you have four hard disks in your system, you should create only three paging files. One in each of the other hard disks, leaving the primary boot hard disk without a paging file.

Creating Multiple Paging Files In Windows XP

First, open up System Properties, either through the Control Panel or by right-clicking on the My Computer icon and selecting Properties.

Once in System Properties, click on the Advanced tab. There will be three sections.

Click on Settings in the Performance section and the Performance Options screen will pop up. Click on the

Advanced tab and you'll see the following picture :-

The second section you see is titled Virtual memory. Under it, there's a Change button. Click on it to get the following screen :-

This is where you manage Windows XP's paging file settings.

You can select the logical drive in you wish to create a paging file by clicking on the list of logical drives shown on the screen.

Just scroll through the selection of logical drives available. Click on the logical drive in which you want to create a paging file. Then set the initial and maximum paging file sizes and click Set. Do this for as many hard disks as you want in your system.

To remove the paging file from the default location in the first logical drive, select drive C: and select No paging file. Then click Set.

After you are done, just click OK and allow Windows XP to reboot your computer. After rebooting, multiple paging file will be created in the logical drives you selected.

Please remember that each logical drive represents a partition, not a physical drive. So, if you want to create a paging file in a separate hard disk, select a logical drive that resides on that hard disk. Preferably, it should be the first partition in the other hard disk (which should be on its own IDE channel).

Moving The Paging File To A RAID Array

Before proceeding, you should read our RAID Optimization Guide for a primer on the different RAID levels.

RAID 0

RAID 0 uses striping to achieve better performance. Putting the paging file on a RAID 0 array will greatly improve both its read and write performance because the paging file will be split up between the hard disks in the RAID 0 array.

Although RAID 0 does not offer any data redundancy, that is perfectly alright for the paging file since it is only used for the temporary storage of the system's memory contents.

RAID 1

A paging file on a RAID 1 array may benefit from a faster access time. But because the paging file has to be mirrored on a second hard disk, this greatly degrades the paging file's write performance. In addition, the paging file will not benefit from the additional data redundancy offered by RAID 1.

Therefore, it is recommended that you do not put your paging file in a RAID 1 array. You should place the paging file on a separate hard disk.

RAID 0+1

Although the paging file will benefit from the increased read performance in a RAID 0+1 array, its write performance will be as severely degraded as it would be in a RAID 1 array. In addition, the paging file will not benefit from the additional data redundancy offered by RAID 0+1.

Therefore, it is recommended that you do not put your paging file in a RAID 0+1 array. You should place the paging file on a separate hard disk.

Moving The Paging File To A RAID Array

RAID 5

RAID 5's distributed parity requires a lot of calculations. Moving the paging file to a RAID 5 array will greatly increase the amount of calculations the RAID controller has to do. This greatly reduces its performance. Again, the paging file will not benefit from the additional data redundancy offered by RAID 5.

Therefore, it is recommended that you do not put your paging file in a RAID 5 array. You should place the paging file on a separate hard disk.

JBOD

When the hard disks in a JBOD array, they act exactly like a single logical drive. Therefore, placing the paging file here will not incur any benefits at all. It will just be like putting it on the first hard disk.

Conclusion

Generally, placing the paging file in a RAID array is not recommended at all. The only situation where a RAID array can actually improve the paging file's performance is if the RAID array was created using RAID 0. Otherwise, avoid placing the paging file in a RAID array.

How To Place The Paging File In The RAID Array

To the operating system, the RAID array appears as an ordinary logical drive. Therefore, moving the paging file to the RAID array is as simple as moving it to another logical drive. Just follow the instructions for moving the paging file to a different logical drive.

Moving The Paging File To A RAM Drive

A RAM drive is nothing more than a logical drive created out of system memory.

To the operating system, a RAM appears as a normal logical drive, albeit a very fast one! Because it is created using system memory, a RAM drive is fast. VERY fast, in fact. As I mentioned earlier, even dual-channel PC2700 DDR memory is over 70X faster than the fastest hard disk.

That's why many people actually advocate moving the paging file to a RAM drive. Their reason is simple. Because a RAM drive is so fast, moving the paging file there will greatly improve its performance.

They are most certainly correct. Moving the paging file into a RAM drive will definitely give it an enormous boost in performance. However, that is really counter-productive. Let's see why.

From RAM To Hard Disk To RAM?

The purpose of a paging file is to create virtual memory for situations where there is not enough system memory. Virtual memory serves as an emergency source of additional memory, only to be used when there is not enough system memory.

The RAM drive, on the other hand, is used to create a very fast pool of temporary storage space in the system memory. It ties up system memory, so it is usually created when there is a lot of free system memory. Even then, it is usually kept small and only used to store temporary work files.

Therefore, does it make sense to use limited system memory to create a RAM drive that is used to service the paging file? No, it doesn't make sense at all.

Remember, most paging files are very large, a few hundred megabytes to a gigabyte in size. Most computers even come with enough memory to create such a large RAM drive, much less move the entire paging file to the RAM drive.

Even if you have a lot of system memory, creating a large RAM drive reduces the pool of available system memory. This increases the need for virtual memory which means more data will have to be paged out into the paging file. This increases the size of the paging file which will inevitably be much larger than the RAM drive.

At this point, Windows will automatically create more virtual memory via a dynamic paging file. Because a large portion of the system memory has already been taken up by the RAM drive, this will cause a lot of data to be paged out to a large dynamic paging file on the hard disk. That defeats the purpose of moving the paging file to the RAM drive - improved performance.

If your system has a lot of system memory, don't waste your time creating a RAM drive to service the paging file. If there is a lot of free system memory, Windows will not need to page out data to the paging file. That would produce the best results. Nothing is faster than running the programs directly in system memory.

Reducing Reliance On Virtual Memory

Windows can get too enthusiastic about paging data out to the paging file. This can lead to unnecessary paging, even when there is a lot of free system memory.

Luckily, we can do something about this :-

+ Enabling the Pagefile_Call_Async_Manager service (Windows 98, Me)
+ Stop NTExecutive from paging (Windows NT, 2000, XP and above)

We will take a look at each and see how they work.

Enabling The Pagefile_Call_Async_Manager Service

Microsoft added a Pagefile_Call_Async_Manager feature in Windows 98. This ostensibly forces Windows 98 to behave more like Windows 95 by asynchronously paging out data during periods of inactivity. According to Microsoft, this decreases performance. However, it is actually quite the opposite.

Enabling this feature actually forces Windows 98 to be more conservative about using the paging file. Windows 98 will reduce the amount of paging and keep more data in system memory. This improves performance by keeping more data in system memory than in the paging file.

Needless to say, it is recommended that you enable the Pagefile_Call_Async_Manager service. Just make sure your system has a good amount of system memory.

To enable the Pagefile_Call_Async_Manager service, you will have to edit the System.ini file (usually found in the drive:\\Windows\ folder.

Look for the [386Enh] section. In that section, add the following entry under the [386Enh] section :-

```
[386Enh]
ConservativeSwapfileUsage=1
```

Save the change you made to the System.ini file and reboot the computer. When Windows 98 boots up again, it will be using a more conservative approach to paging.

Incidentally, this method is said to work with Windows Me although I cannot confirm this. But please note that this method does not work in Windows NT, 2000 and XP.

Stop NTExecutive From Paging

In Windows NT, 2000 and XP, you can prevent pageable drivers and system code in the Windows NT Executive from being paged out to the paging file.

Normally, pageable drivers and system codes are paged out to the paging file to free up memory. Naturally, this reduces the performance of the operating system and affected drivers.

However, you can easily change that and force Windows to keep all drivers and system code in the system memory. But you will need to edit the registry.

Start up Registry Editor by running regedit.exe in the drive:\Windows\ folder or by going to Start Menu -> Run... -> regedit.exe.

Once you have opened up Registry Editor, go to the following subkey :-

HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Session Manager\Memory Management

You will see the following screen :-

Look for the DisablePagingExecutive option. By default, it is set to 0.

Double-click on it and change its value to 1. Then close Registry Editor and reboot your computer.

When Windows boots up again, the pageable drivers and system code will no longer be paged out to the paging file. Instead, they will be retained in system memory for maximum performance.

This method works with Windows NT, 2000 and XP. It does not work with Windows 98 or Windows Me.

Conclusion

Optimizing the paging file isn't a very hard thing to do. The main problem is evaluating and selecting the

best methods of optimization for your system.

The previous pages have discussed, at some length, the pros and cons of the different methods. By now, you should be able to see a pattern.

Evidently, creating a semi-permanent, contiguous paging file that is slightly larger than what you normally need and moving it to the outer tracks of the hard disk are generally the best ways to optimize the paging file. If you have multiple hard disks, creating multiple paging files will also greatly improve its performance. Needless to say, we should also force Windows to reduce its reliance on virtual memory.

But we should generally avoid placing the paging file in RAID arrays or a RAM drive. It also does not make sense to create a really massive paging file. Needless to say, it is counter-productive to simply move the paging file to another partition of the same hard disk.

I hope this guide has been of great help to you in optimizing the paging file. Let us know if you have any comments or perhaps new tips on further optimizing the virtual memory system!

Please feel free to take a look at our other guides and reviews or drop by our forums for a chat.

Don't forget to try your luck in our Hunt for the BOG Book Mega-Contest! There are USD 5000 worth of prizes to be given away! Don't miss out on this opportunity!