

Realtime
publishers

"Leading the Conversation"

The Shortcut Guide[™] To



Managing Disk Fragmentation

sponsored by



Mike Danseglio

Chapter 3: Solving Disk Fragmentation Issues.....	32
Performance	33
Backup and Restore	34
Stability.....	35
Addressing the Disk Fragmentation Problem.....	36
Evaluating a Defragmentation Solution.....	36
Cost.....	37
Defragmentation Engine Operation	37
Deployment.....	39
Operational Autonomy.....	40
User Experience	41
Reporting.....	44
Defragmentation Approaches	45
Automatic Defragmentation.....	45
Manual Defragmentation	46
How to Make Your Decision	46
Preselection.....	47
Test.....	47
Purchase	48
Deployment.....	49
Summary	50

Copyright Statement

© 2008 Realtimedpublishers.com, Inc. All rights reserved. This site contains materials that have been created, developed, or commissioned by, and published with the permission of, Realtimedpublishers.com, Inc. (the "Materials") and this site and any such Materials are protected by international copyright and trademark laws.

THE MATERIALS ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. The Materials are subject to change without notice and do not represent a commitment on the part of Realtimedpublishers.com, Inc or its web site sponsors. In no event shall Realtimedpublishers.com, Inc. or its web site sponsors be held liable for technical or editorial errors or omissions contained in the Materials, including without limitation, for any direct, indirect, incidental, special, exemplary or consequential damages whatsoever resulting from the use of any information contained in the Materials.

The Materials (including but not limited to the text, images, audio, and/or video) may not be copied, reproduced, republished, uploaded, posted, transmitted, or distributed in any way, in whole or in part, except that one copy may be downloaded for your personal, non-commercial use on a single computer. In connection with such use, you may not modify or obscure any copyright or other proprietary notice.

The Materials may contain trademarks, services marks and logos that are the property of third parties. You are not permitted to use these trademarks, services marks or logos without prior written consent of such third parties.

Realtimedpublishers.com and the Realtimedpublishers logo are registered in the US Patent & Trademark Office. All other product or service names are the property of their respective owners.

If you have any questions about these terms, or if you would like information about licensing materials from Realtimedpublishers.com, please contact us via e-mail at info@realtimedpublishers.com.

[**Editor's Note:** This eBook was downloaded from Realtime Nexus—The Digital Library. All leading technology guides from Realtimepublishers can be found at <http://nexus.realtimepublishers.com>.]

Chapter 3: Solving Disk Fragmentation Issues

Chapter 1 explored how disks work. They were designed as efficient long-term data storage devices, and they've lived up to that design criteria well. The first disks were large, clunky, fragile, and had very limited storage capacity. Over time, disks have significantly evolved. A disk today might fit on a postage stamp, draw virtually no power, have a lifetime measured in decades, and have the capacity to store the entire Library of Congress. Performance has also come a long way with today's disk throughput being orders of magnitude more than even a decade ago.

Cost has always been a concern about disks. In Chapter 1, we learned that disks used to be extremely expensive and hence very rare. Today they're virtually commodity items. You can buy a reliable, high-capacity disk drive at any office supply store for less than the cost of a nice chair. Overall the disk storage market has boomed and products are keeping up with demand. As an example of the drastic evolution in the market, at the time of this writing, a fully redundant disk array that provides one terabyte of storage can be implemented for less than \$1000 using off-the-shelf hardware and does not require specialized knowledge or extensive consulting. Such disk arrays were scarcely available to consumers and small businesses even 5 years ago and, when available, required extensive consulting with storage experts, specialized hardware implementations, and cost tens of thousands of dollars or more. In short, disk-based storage is getting cheaper, easier, and more commonplace.

Disk operation is not all paradise, though. There are many issues to consider when operating disks. None of them should prevent you from using disk storage. However, they should be taken into account when implementing and operating any disk-reliant system. These issues can include:

- Disk lifetime—How long will each disk drive work before it fails?
- Throughput—How quickly is data getting from the storage system to the computer?
- Redundancy—Is the system truly redundant and fault tolerant?
- Fragmentation—Is the disk system operating at optimum efficiency?

Then in Chapter 2 we explored one problem, disk fragmentation, in great detail. In a nutshell, bits of a file get scattered all over a disk. This makes reading the file more difficult for the hardware to accomplish, decreasing the efficiency of disks and slowing down disk throughput. When critical files or often used files become fragmented the impact can be profound.

Fragmentation is a problem that can actually result from a number of different causes. Unfortunately these causes include normal daily operation of a disk. Over time, disks will become fragmented. There are preventative measures that we can take, and many that are designed right into our modern operating and file systems. But these measures only delay the inevitable.

The problems caused by fragmentation can generally be broken up into three categories: performance, backup and restore, and stability.

Performance

Most modern computer systems identify the disk storage system as a performance bottleneck. The CPU, memory, and data bus speeds increase almost as fast as customers can adopt them. But disk speeds have traditionally been slower to increase. This is mostly due to the nature of disk storage construction being based on moving parts including the rotating spindle and the read-write heads.

Disk throughput is serialized. The data must be transmitted to or received from the computer in a very specific order, because that's the order that the data makes sense. This can cause some performance issues when the disk cannot accept all of the data at once. The computer must wait for the disk to write the first part of the data before sending the next part. When the system must wait for the disk to become free for either read or write operations, the system's performance is often noticeably affected. This is illustrated in the following figure. Note how the file must be broken up. Each data segment requires its own operation, which naturally slows it down.

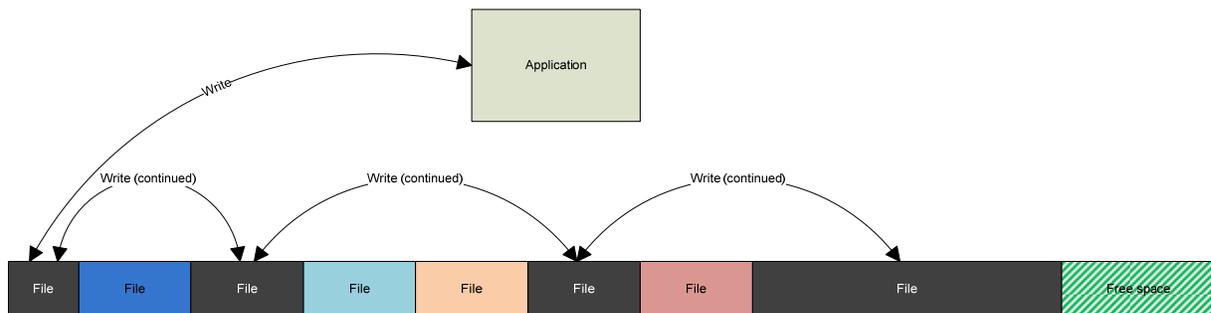


Figure 3.1: Writing a fragmented file. This file requires a minimum of four write operations just for the data.

When either reading or writing to disk, efficiency is critical to a quick and effective operation. Ideally all of the desired data is in one spot when reading from a disk, and a large enough piece of free disk space is available when writing to disk that the entire file can fit in one spot. Files where all the data is located in one spot are called *contiguous*. Contiguous files are the fastest to read and write because they optimize the disk's efficiency. Files where the data is in more than one location on the disk are *discontiguous* or *fragmented*. Depending on the level of fragmentation, these files can be very inefficient to operate on. The result can be read or write delays. The delays increase as the amount of fragmentation increases.

Backup and Restore

Companies that place value on their data go to great lengths to protect it against loss or compromise. They cannot lose the central focus of their business when a hard disk fails or a power supply blows up. These companies invest in a variety of data protection methods to minimize loss and downtime. One of the most basic and most effective methods is data backup.

In short, data backup is the process of copying data from a main source to a secondary source; for example, burning a DVD with a copy of a customer database stored on a server's hard drive. Normally, the data backup is carefully stored in a secure location so that when the original source is compromised, it is likely that the backup will be unaffected.

Data backup is often a slow process. There are several factors that contribute to data backup being slow:

- Volume of data can be enormous
- Inability to take the data offline, requiring a complex backup scheme to copy the data while it's being changed
- Scarce system resources available on data host
- Fragmented state of data source

Most standard data backup practices have built-in mitigations to these factors. They include scheduling backups during periods of system inactivity, purging unwanted data before the backup begins, and (less frequently) scheduling system downtime to coincide with data backup. However, many organizations ignore data fragmentation as a component of data backup. It's simply not part of their thought process. This is an incorrect oversight.

Data fragmentation can significantly impact a backup process. As we've already seen, fragmentation leads to delays in reading data from the hard disk. Data backups rely on reading data as quickly as possible for two reasons: to speed the backup process and to efficiently supply data to continuous-write devices such as DVD drives. Heavily fragmented data will take longer to read from the disk. Thus, at best, the backup process takes longer to complete. The worst case is that the backup will fail due to the delay in supplying data to the continuous-write data backup device.

Similarly, data restore is the opposite of data backup. It is the process used to take the information from a data backup and place it back in a usable state. This is most often a result of the primary data source becoming damaged or failing. For example, when a disk drive fails, the backup of the data from that drive is restored to another drive for continued use.

Usually data that is stored on a backup device is restored through normal file write operations. These operations rely on fragmentation avoidance techniques built in to the operating and file systems. However, the state of the disk at the time of restore plays a significant role on how the files are written. If the disk is crowded and there are few contiguous free spaces to write new data, the files will most likely be fragmented as they are written. This fragmentation will continue as the available free space becomes scarcer and even more fragmented. This issue is illustrated in the following diagram.

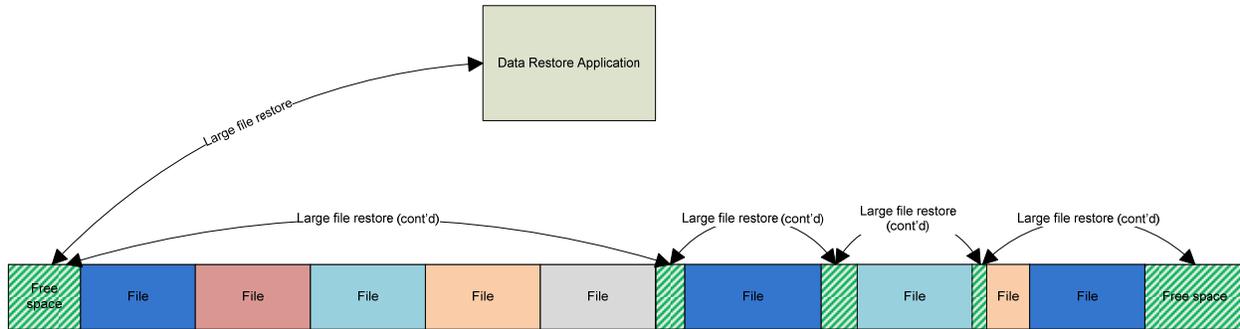


Figure 3.2: Restoring data to a disk with little or fragmented free space results in a fragmented file.

Stability

It is even possible that a fragmented system can go beyond slowdown to failure. Although rare, it is a possibility. The reason is that during OS boot, key system files must be loaded in a timely manner. The most likely cause for this scenario can be traced back to a heavily fragmented master file table (MFT) on a Windows computer running the NTFS file system. The MFT holds key information about all other files on the hard disk. Fragmentation of this file has a cascade effect, causing all other disk input/output to slow down while the MFT is read. Windows attempts to keep the MFT in a single extent but often fails to do so, especially when using a small or nearly full hard disk. Although other key Windows files can cause boot failures if they're fragmented, the MFT usually has the biggest impact.

If the key OS files are not loaded within a predetermined time limit, the OS may think that it has become compromised or corrupted. In that case, it will display an error message and stop the boot sequence in order to protect system integrity. This results in system downtime and could potentially require reinstallation of the OS to fix (unless you have a solution in place to defragment offline systems).

Addressing the Disk Fragmentation Problem

Now that we clearly understand how disk fragmentation works and why it is a problem, we can work on addressing it. Today, the most effective way to resolve fragmentation issues is to use a software-based disk defragmenter. These software packages can be highly effective and, over time, can not only eliminate fragmentation but also prevent it from reoccurring.

The remainder of this chapter serves as a guide to help you determine the best method to eliminate fragmentation in your environment. It is broken up into two sections:

- **Evaluating a Defragmentation Solution.** This section examines the various features of a defragmentation solution that should be considered when deciding which solution to choose. The most important areas are called out and decision criteria are provided to help you make an informed choice. No preference is given to any specific defragmentation solution. The decision is yours. This section merely helps you tell the different in features to help you make that decision.
- **Defragmentation Approaches.** There are two general approaches to running defragmentation software, automatic and manual. Each has benefits and drawbacks. This section calls out those benefits and drawbacks so you can decide which is best for your environment.

Evaluating a Defragmentation Solution

We understand that solving the fragmentation problem requires some type of software solution. But which one should we use? Luckily this decision isn't as hard as it seems.

The defragmentation software market isn't nearly as scattered or difficult to navigate as, say, the email server market. There are a couple of major players in the defragmentation market, several smaller niche players, and a solution built into Windows itself. Each of these solutions, even the most basic, has some benefits and drawbacks when compared to the other available options.

Our approach in this section is to examine the most common decision making criteria when evaluating a software product for wide scale deployment. For deployments of just five or ten computers, this type of exhaustive research and decision process may be overkill. It would probably more cost effective to just select a well-known software package and go with it. But if you've got hundreds or thousands of computers that need a defragmentation solution, it is best to perform a careful analysis before purchasing or deploying anything. This will help ensure that the software meets your expectations and solves the right problem in the right way.

 For more information on general software evaluation processes, consider reading about the [Software Selection Process](#). This process is presented by Technology Evaluation Center (<http://www.technologyevaluation.com/>) and is applicable to virtually any software evaluation and purchase decision. This web site has a number of tools and techniques that you can use to help simplify and improve your software selection process.

Cost

When we get right down to it, cost is always a consideration. Software licensing is never inexpensive, no matter what the software is. Some companies take “liberties” with their software licensing. The purpose of this paper isn’t to explore what a valid license agreement or use is. You need to determine that for yourself. But for the sake of this paper let’s assume that you’re going to purchase a license for each installation of the defragmentation software.

Most companies require a [cost-benefit analysis](#) before completing any type of large scale IT purchase. In the case of defragmentation software it should be no different. Throughout this paper we’ve examined how fragmentation can negatively impact an organization. The benefits to deploying the defragmentation software include increased productivity from higher performing computer systems without the need for hardware upgrades, stability of systems, and a higher degree of data integrity. All of these have direct value to an organization, and in most cases outweigh the cost of purchasing licenses.

When planning to purchase software, you should also take into account the indirect costs such as the costs of testing, deploying, and supporting the software. Although this is not a direct and specific dollar amount, it can be considerable. For example, you might evaluate two defragmentation solutions. One costs \$35 per installation, the other \$60 per installation. They both allow automatic background defragmentation and your research shows that they produce similar on-disk results. However, the \$35 solution requires interactive installation and must be manually updated when a newer version is available. The \$60 version provides extensive deployment automation and maintains itself over time. So which one is a better investment? The answer is that it depends on your environment and needs. Both are viable options. But you might not fully appreciate the subtle long-term differences between the options until you fully research and test both.

☞ Check both the initial and long-term costs, because you’re really investing in a solution for today and the future.

Defragmentation Engine Operation

When it comes right down to it, the defragmentation software needs to find all files that are fragmented and defragment them. This includes the operating system files, file system metadata, and all data files. Few files, if any, should be excluded, though some are harder to defragment because of how the operating system works or because of the limited impact of fragmentation on them. For example, the Windows pagefile is difficult to defragment because Windows itself puts a lock on the file whenever the operating system is running. So a different approach is required, such as defragmenting the file during boot time before Windows puts a lock on it.

Interestingly, virtually all commercial defragmentation software does this work. The results are about the same. Small features here and there might be different. Some defragment files slightly more efficiently or faster than their peers. Others claim to defragment in such a way as to leave bigger chunks of free space for future files to avoid fragmentation. But ultimately when all of the packages say that they will defragment the file system, in most cases they do it with roughly equal results.

How the engines get the files to a defragmented state, however, is an interesting area to explore. Consider that in Chapter 2 we learned that to defragment a file the software will read the entire file, locate contiguous free space where the file will fit, write the file there, and then delete the original file. This takes system resources. Disk usage obviously goes up when this is occurring. But CPU and memory are consumed to some extent as well. While the disk throughput can be slightly changed based on how the engine works, there's not a lot of wiggle room there. The significant difference between engines can come in CPU and memory utilization.

Obviously we want less CPU and memory used while the system is in use. For systems running 24 hours a day with consistent load, the only solution is really to simply ensure that the engine operates slowly and in the background, never interfering with the system's intended use. For computers such as desktops or workgroup-based servers, the engine should behave very differently. If all your users work from 9AM to 6PM, the engine should either be configurable to not use any resources that the system or other applications need within that time block or do so automatically. Once the users go home, it doesn't matter if the engine consumes 100% of system resources because the user will not be impacted as long as the process is completed by the time the user returns. You should look for software that allows you this flexibility.

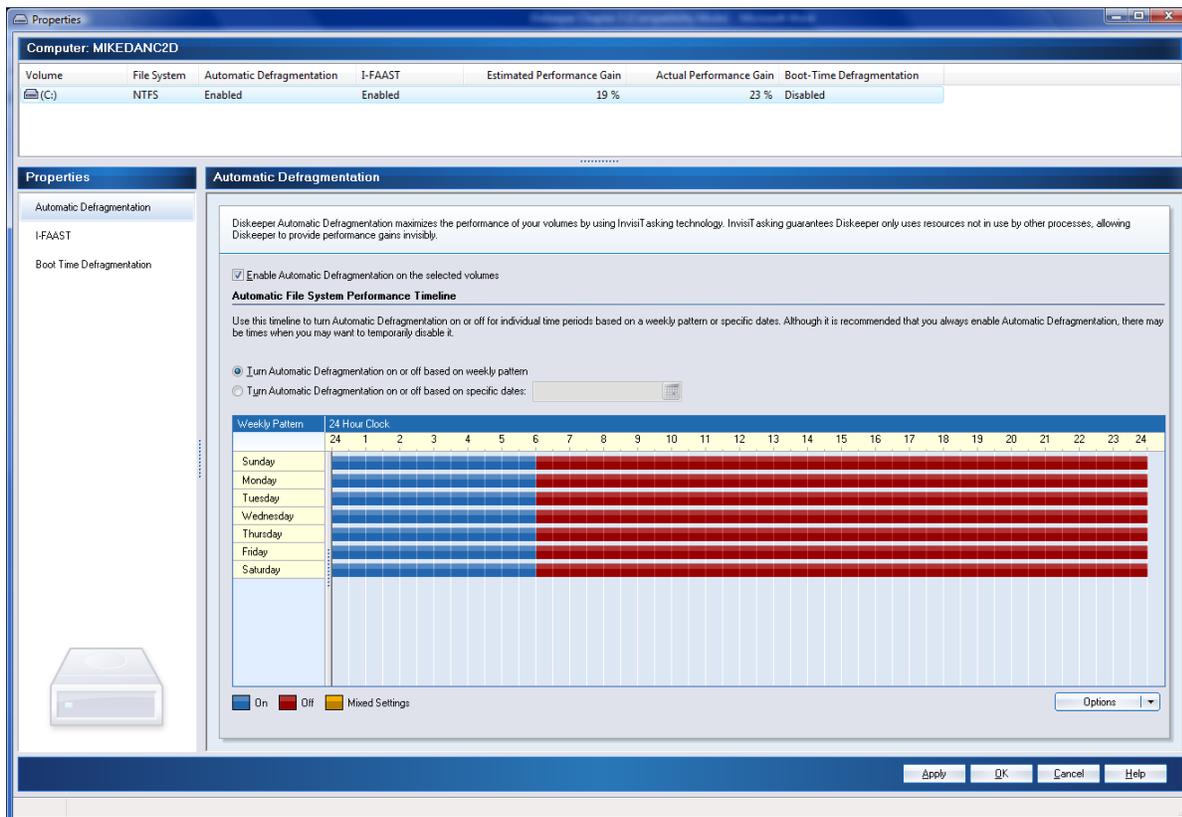


Figure 3.3: An example of how one software defragmentation package allows the user to configure what times defragmentation will and will not run. Note that in this case defragmentation is not forced to run, but is permitted to run if the volume requires it.

☞ Most of the defragmentation engines produce the same result in many cases. However, the way they get there can be different. Ensure the software doesn't interfere with the operation of the system.

One feature that most high end defragmentation engines tout is the ability to prevent future fragmentation by optimizing the disk layout. The optimization method varies by software, and is often unique to that package. Although this can be useful for systems with high file throughput (e.g. file servers or systems that create a number of temporary files), its benefit could be somewhat limited. If the entire file system is frequently defragmented before fragmentation ever affects the user, it can be a far more efficient solution for optimized performance. In addition, the result of an optimized layout is somewhat theoretical because in a high-throughput scenario, and how the particular file system allocates the writing of new data, the operating and file systems will quickly refragment the system regardless of the free space layout.

Deployment

You'll need to get the software out to the client and server computers in your enterprise. How do you plan to do that? Walking up to each computer and installing from CD or USB memory doesn't scale past a small workgroup because of the labor and the potential inconsistencies when deploying in such a one-off manner. You must automate the deployment of your disk defragmentation solution to have any hope of a successful deployment.

Some software packages lend themselves to simple automated deployments by coming prepackaged and ready for distribution through mechanisms such as Microsoft Systems Management Server or Windows Group Policy. Usually this software is delivered in the form of a single Microsoft Installer (MSI) file. An administrator can simply take the MSI file, point their desired deployment software at it, and tell the software where to deploy it. Most deployment solutions are also good enough to provide scheduling and status updates to help the administrator know exactly how the deployment is going and whether there are any problems.

Digitally Signed Software

Digital signatures are becoming more common for Windows based software. The signatures are usually found on the installed software, but the need for digitally signed installer packages is quickly becoming important. Windows, by default, resists installing software that's not digitally signed because malware is very likely to be unsigned. To help ensure the smoothest installation experience and help support your company's security policy, you should seek out software solutions that provide both a digitally signed application and a signed installation package.

An automated deployment is the simplest and easiest way to get the software out to the clients. It is also the most consistent. Software-based deployments won't forget a desk because it's in the corner or miss Larry's computer because he has it in the back corner. There are a number of other reasons to recommend this method of deployment. The time and money savings, combined with the reasonable consistency and completeness of an automated deployment, makes it the preferred method.

Undeployment

We understand the need for some method of automated deployment. But what happens if the software doesn't produce its intended result or doesn't get funded long term? Can you take the software back from the systems you deployed it to? This can become important, especially in cases of licensing where you no longer have permission to use the software. Failure to undeploy the software consistently may result in liability or a destabilized environment. You should ensure that the software will come out just as easily as it comes in. This can be proven during the testing process, described later in this document.

The only exception to an automated deployment should be disconnected or remote computers. Usually this means laptops and edge servers. These computers usually are not joined to the domain and frequently have no automated maintenance methods available. However, they are just as in need of the defragmentation solution as the rest of the computers, if not more so. In these cases, consider using a network map or list of computer assets to ensure that all computers receive the proper software installation.

☞ Deploy your defragmentation solution through automated software installation wherever possible. Examples include Microsoft's System Center Configuration Manager (formerly SMS) and Windows Group Policy. For computers that can't receive automated software delivery, do it the old fashioned way – by hand.

Operational Autonomy

Once the software is deployed, it begins to operate. How well does it do on its own? Most software is good about using default values to begin operating properly. Even if the software cannot get additional setting changes from a central location like Microsoft's Group Policy or a software-specific administration server, it should be intelligent enough to begin operating without any additional information.

Over time the use of a system may change. Perhaps a disk is added to the system. Does the software recognize that and adapt itself? The more effective software solutions do. If specific setting changes are required whenever a system configuration is modified, the cost of administering that software solution goes way up. This should be something you consider long term.

☞ If the software can operate itself with little or no administrative interaction, that's usually a good thing. Self updating and configuring with intelligent defaults are the two biggest features in this category.

User Experience

As we found earlier, for the most part the defragmentation engines do the same thing. They reassemble bits of fragmented files into one big unfragmented file. Users care about this indirectly because they want higher performing, more reliable computers. But what users and administrators truly care about is the simplicity of using the software. They care about the user experience.

Defragmentation software has made tremendous inroads in the user experience area over the last decade. Some of the earliest solutions were nothing more than a command-line entry of

```
C:\> Defrag.exe
```

This resulted in a prompt telling the user to wait until defragmentation was complete. This is an effective if unpleasant and cold experience. Through the years the experience changed, most notably with the disk space “kaleidoscope” where data and files were represented by different colors – red for fragmented and green for contiguous files, for example.

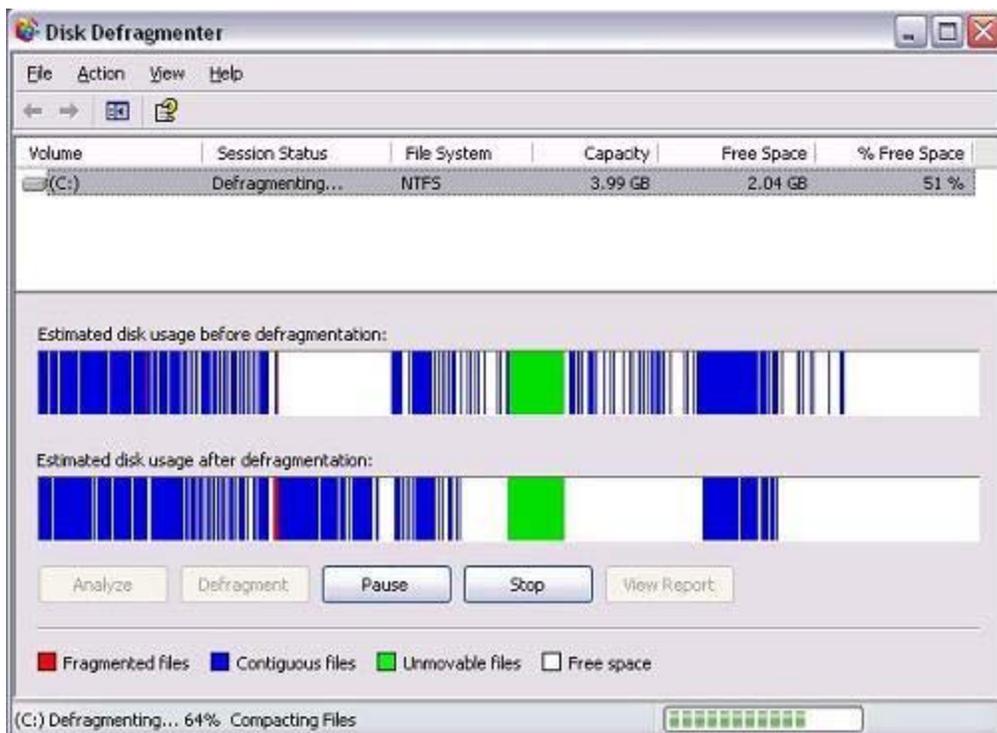


Figure 3.4: Windows XP's built in Disk Defragmenter tool with its basic “kaleidoscope” display.

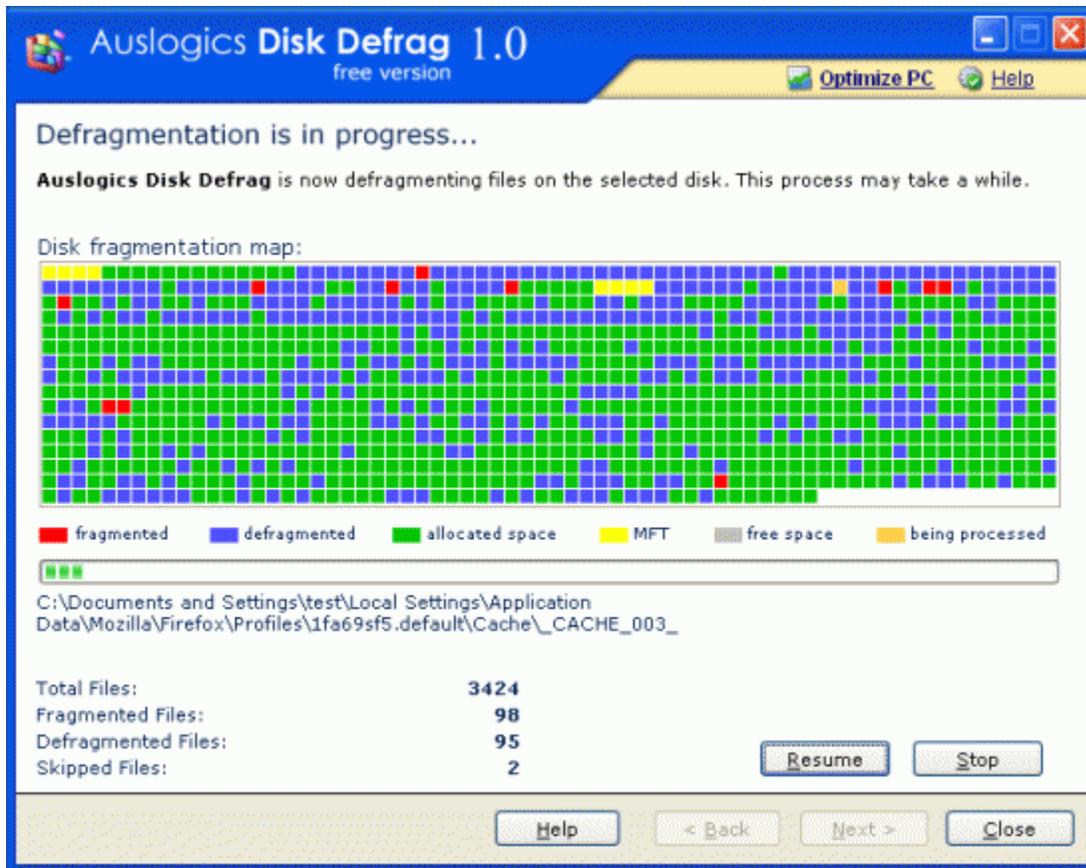


Figure 3.5: A typical defragmentation software package. Note that the display is almost identical to the Windows XP built in utility except the colors are small boxes instead of tall lines.

Today, the user experience for most defragmentation software is an effective balance between pleasant graphics and task-oriented objects to help the user understand status and make decisions. For example, this is what a typical screen looks like from the Diskeeper defragmentation solution:

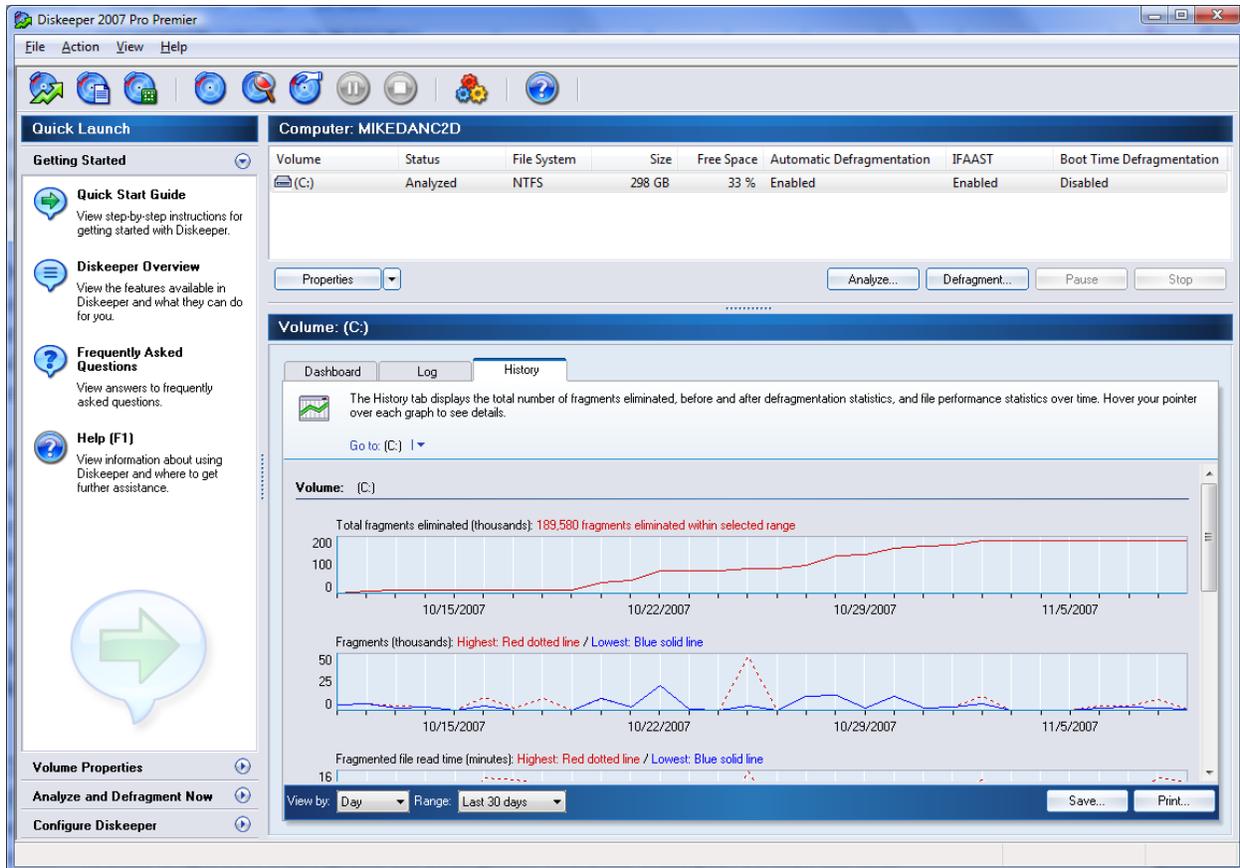


Figure 3.6: Diskeeper 2007 Pro Premier. Far different than the command-line defragmentation software of a decade ago.

Usability isn't everything though. A pretty shell with no substance behind it isn't very effective at improving system uptime or performance. Luckily most defragmentation software concentrates on the engine before polishing the shell. But when in doubt, the usability should be considered less important than the operation of the system itself.

☞ The user might never see the defragmentation software depending on how you deploy it and how it operates. So the usability feature might be chiefly for administrators. Ensure that the person that needs to operate the software understands it.

Reporting

Once we've decided on our preferred defragmentation solution, tested it, and deployed it, are we done? For some administrators, the answer is yes. But for most of us, the answer is a resounding "No!" We need to ensure the solution is working properly, both immediately after and long-term as a sustained operation. This is where reporting comes in.

There are two main functions for reporting on disk defragmentation solutions. One is to verify that the software is, in fact, installed and functioning properly on the desired computers. Getting a daily report that shows that the defragmentation software operated properly is very useful. If the computer reports an error or fails to report, an administrator can address it before the computer is impacted by excessive fragmentation.

The other reason we gather information on defragmentation results is to see whether the software is making a difference. Whenever we spend money on software solutions, we want to be able to measure the impact that this software had. Simply telling the CIO that "The computer was defragmented and that's a good thing" doesn't really support an ongoing business model. However, telling her that the defragmentation solution has removed over 19,000 fragments per week and that resulted in a 7% increase in throughput on a central server is a very significant statement and can easily justify the software investment that you've made.

If your organization wants to provide ongoing justification for your defragmentation investment, or you want to ensure that the software is installed and operating properly over time, consider obtaining a software package that provides rich reporting features. Some packages simply put an entry in the Windows Event Log that contains little more information than "Defragmentation job ran." While this might be enough for you, other packages contain much more information. Some data you might want to gather, depending on your specific organizational needs, could include:

- Defragmentation start and stop time
- Number of file fragments defragmented
- Condition of file system (e.g. NTFS metadata)
- Version of software running

All of this information should, optimally, be compiled into a report. Although there are any number of great reporting software packages available, the better defragmentation packages already have most of that functionality built right in. They can provide detailed analysis about performance impact and, from that information, you can clearly show the benefit and justify the cost of the investment.

 Reporting is key for initial installation verification and for ongoing maintenance. If you plan to track the defragmentation software over time, ensure that the software has the capability to report on itself. Also check to see if it requires other software to coalesce reports, as this might be an expensive prerequisite.

Defragmentation Approaches

Once you've determined the best defragmentation software solution for your environment, you're ready to decide on an approach. There are two categories of defragmentation approach, manual and automatic. We'll explore both of these in this document.

You can actually explore both approaches and test them in your pilot or test environments to help you make a decision. Although it's easy to decide on one approach or the other, most of the time you'll need to have one as a default approach and make exceptions where the other is most appropriate or the only viable option.

Automatic Defragmentation

As its name implies, automatic defragmentation just happens. On a regular basis (usually in the middle of the night), the defragmentation software wakes up, scans the drive, and if necessary performs a defragmentation and clean-up operation. Usually the system reports the results of the operation to a central server (see Reporting in this document).

This approach seems to be the obvious choice. And for most environments and applications it is the best way. The benefits to using automatic defragmentation include:

- No user or administrator interaction required, making the experience easier and more likely to succeed
- Predictable operation
- Customized run time to ensure little interference with normal system operation

Almost all defragmentation solutions default to some type of automatic configuration. A few older or more limited solutions require you to configure them as batch jobs or via scripts to run automatically. These are not recommended because they're more likely to fail and will almost always lack any reporting features. And although this isn't a direct correlation, most applications that have to be executed in this way consume enormous system resources as they believe that the user is interactively executing the defragmentation process. This could result in resource issues if the process is triggered at the wrong time or runs too long.

There may be situations where you do not want to use automatic defragmentation. You might want to completely control when and how the defragmentation software operates. In those cases, you'll need to use the manual defragmentation approach.

Manual Defragmentation

Automatic defragmentation works in the majority of situations. The more automated the solution, the less we have to worry about mistakes or errors causing a breakdown in the system. And for most systems, regular and predictable defragmentation is the preferred solution. But, depending on the product, there may be times where an automatic solution is just not going to work. Let's look at an example.

You are in charge of a small web server cluster that consists of two web servers and one database server. This cluster has very specific performance requirements that it is just barely meeting. You run backups and maintenance whenever the load is low. Unfortunately, due to the nature of the traffic and the users that hit this cluster, you never know when a spike or sag in traffic will occur. In this case, an automatic solution might kick off a resource-intensive defragmentation pass in the middle of a usage spike. That could easily push the performance outside the minimum requirements. If this is a possibility with software, the better method would be to manually kick off the defragmentation pass during a lull and, if traffic picks up before the conclusion of the pass, stop the defragmentation and resume it later.

Some defragmentation solutions include logic that will automatically perform defragmentation when the system reaches an idle state, and stop the process when the system begins being used. This works well for some applications such as desktop and laptop computers, and depending on the technology, for servers as well.

How to Make Your Decision

You've looked at the available defragmentation solutions. You've decided on a default defragmentation method and potential exceptions. You have an idea of how many computers will receive the software and how it will be deployed. Now you need to make your purchase.

The remaining phases of the purchasing process are fairly straightforward. These are common to any software evaluation decision and include:

- Preselection
- Test
- Purchase
- Deployment

Let's take a brief look at each of these phases.

 One phase of the process not mentioned here is operations, which is sustaining the software in the environment over time. Because the sustained operation of defragmentation software solutions is so similar to other software solutions, it is not included here.

Preselection

Now that you've identified the needs of your organization, take a look at the solutions available. There are a number of ways that you can find out information about the features of the software packages. These include:

- Download a demonstration or limited copy of the software
- Read whitepapers from the software developer
- Check software reviews from other corporate users
- Visit the company's web site
- Ask the manufacturer to have a sales representative contact you
- Network to find others who use the same software and ask them their opinions

The desired result of this work is that you'll have one or two solutions that you believe will work best for your needs. There may be a long list of potential solutions, but using these methods against the decision criteria we developed earlier should help bubble the best candidate to the top. Once that happens, we can examine the best candidate through testing.

Test

If you asked a hundred of your peers whether they would test a preselected software package before deploying it in their environment, I'm sure 99 of them would say yes. In fact, you're probably wondering why this section is more than one simple sentence, "Test before you buy." The reality of the situation is that, while the statement is true, there is a bit more to it.

There are a number of things you should look at during your test suite. Hitting these will help ensure that you make the right purchase decision and that future issues with the software are minimized and well understood.

Some basic test methodology and items to look for during the process include:

- Setup an isolated test environment to minimize impact on production resources
- Ensure the test environment is representative of the entire production environment
- Ensure the test deployment mirrors the intended production deployment
- Test normal use cases, such as a user running Microsoft Word while the system defragments a minimally used drive
- Test edge cases, such as a system under 100% load while the defragmentation process engages on a near capacity and heavily fragmented volume
- Verify that the reporting component provides desired data
- Verify that the engine updates itself if necessary
- Document your findings
- Consult the software manufacturer for assistance with unexpected or undesired results
- Ensure the software can be undeployed gracefully

This may sound like a lot. But with virtualization and a small hardware investment this can be accomplished on one or two computers with just a few days of work. Once this set of tasks is complete and you are satisfied with the results, consider performing a pilot deployment. Choose a small number of users and computers that are representative of your organization and deploy the software to only them. You can then measure the impact in production without the potential for widespread impact to the organization.

Deployment Guide as a Result of Test

Most organizations overlook one key element of testing that often justifies the entire process. During test, you have to deploy and redeploy a number of times. And you're documenting the process as you go. A natural result of this work should be a Deployment Guide for the software that you can use in production. This detailed guide will be fully tested and verified before the end of the test process. It is an invaluable document for your deployment staff because they can understand exactly what steps to perform, what results to expect, and how to handle any variances that may occur. And if you're doing a thorough job of testing, this document should require virtually no additional effort.

Once you've completed both your isolated test suite and your pilot deployment, you should have enough information to decide whether to proceed with the purchase and widespread deployment of the software. But do not be surprised at this point if the project takes a different direction. The results of applied testing sometimes help us draw different conclusions than we had previously thought. For example, you might find that your preselected and tested defragmentation solution conflicts with a disk management application that you use on 25% of your computers. In that case, you would be unable to proceed with the deployment, at least to those affected computers. You might decide not to deploy any defragmentation solution, to those computers, to use two defragmentation solutions, or to test your second choice to see if it also has the same issue. But obviously it's better to find this out before you've purchased licenses and begun your widespread deployment.

Purchase

The purchase process will be different for every customer and every software vendor. Virtually every purchase is going to vary to some degree. So providing specific details here isn't really useful. Some software vendors will negotiate bulk pricing, while others will not. Some will accept purchase orders or incremental purchases at the same discount, others will not. You might receive your software funding over time or all at once. The possibilities are endless.

The one element that you should concern yourself about is what you're buying. Is the software copy protected? If so, will you receive one license key for your entire purchase, or a different key for every seat? This can dramatically affect your deployment, so make sure to check with the software vendor. You do not want to have to walk to each computer and type in a unique 35 alphanumeric character key! That would be more painful than a visit to the dentist's chair and far less productive.

You should also ensure that you receive some number of retail shrink-wrapped packages. These are effective as known good, clean copies for building system images and performing test installations from local media instead of the network. Most vendors are happy to supply a handful of these, which should be kept under lock and key per your company's software retention policy. You should ensure that you have enough on hand in case of problems like the loss of a software deployment server or having to install the software to an isolated or offline system. Some software companies offer alternative methods for software acquisition and storage, such as online libraries or the option to burn their software to CD/DVD on demand. Use whatever method you're comfortable with, as long as you have access to a backup of the software in case of emergency.

Deployment

Great! You've analyzed the market, selected a software package, tested it thoroughly enough to know it works for you, and purchased enough licenses to begin your deployment. Now let's get going!

The section on deployment earlier in this document covered the majority of deployment considerations and decision criteria that you make. But by the time you get to this stage you almost certainly have a very specific deployment strategy, plan, and documentation. Now it is time to execute on your well thought out and documented strategy.

In a perfect world, the deployment is the easiest part of the process. But in reality, issues will arise. Conflicts will come to the surface that weren't detected during testing. Your deployment software might hiccup and miss a hundred users. The new software might conflict with another application that's only deployed on a small number of computers so you missed it during testing. Regardless of how well you planned, remain flexible and deal with the snags as they arise.

Consider Standardizing and Automating Your Deployments

If your organization does not currently have a standardized deployment strategy, you should consider investigating this option. The benefits to having one are almost too numerous to list but include ensuring IT consistency across the organization, more effectively managing software licenses, and reducing the total cost of ownership of systems by reducing the deployment time of a computer from hours or days down to minutes with little or no administrator interaction. Consider reviewing Microsoft's [Business Desktop Deployment 2007](#), which includes both guidance and automated tools and is available for free download.

At the end of the deployment phase you have your solution installed and running on all of the intended computers with the software verified and reporting its status. But deployment isn't really ever complete. New computers come into the environment and require one-off deployments. Old computers require undeployment or reconfiguration. This is part of the ongoing software operation lifecycle, but it is the same as any other piece of software.

Summary

Selecting a defragmentation solution can be a difficult process. There are a number of factors to consider. Beyond the basic surface considerations such as cost and features, there are things such as operational autonomy and ease of deployment that will contribute to the long term cost of such a solution. Having a list of things to look at and a process to follow helps us along this process considerably.

In Chapter 4, we will examine the business side of defragmentation. We'll talk extensively about the return on investment strategies and justifications for using defragmentation in the enterprise. We'll talk about cost-benefit analysis in some depth. Chapter 4 is written primarily for those in a decision-making role for enterprises to help with the cost justification of the purchase.

Download Additional eBooks from Realtime Nexus!

Realtime Nexus—The Digital Library provides world-class expert resources that IT professionals depend on to learn about the newest technologies. If you found this eBook to be informative, we encourage you to download more of our industry-leading technology eBooks and video guides at Realtime Nexus. Please visit <http://nexus.realtimepublishers.com>.