Making Everything Easier!™

2nd AMD Special Edition

High Performance Computing

FOR DUMMIES®

Learn to:

- **Pick out hardware and software**
- **Find the best vendor to work with**
- **Get your people up to speed on HPC**

Compliments of

AMD

Douglas Eadline, PhD

These materials are the copyright of Wiley Publishing, Inc. and any dissemination, distribution, or unauthorized use is strictly prohibited.

High Performance Computing FOR DUMMIES®

2ND AMD SPECIAL EDITION

by Douglas Eadline, PhD

Wiley Publishing, Inc.

These materials are the copyright of Wiley Publishing, Inc. and any dissemination, distribution, or unauthorized use is strictly prohibited.

High Performance Computing For Dummies®, 2nd AMD Special Edition

Published by
Wiley Publishing, Inc.
111 River Street
Hoboken, NJ 07030-5774

Copyright © 2011 by Wiley Publishing, Inc., Indianapolis, Indiana

Published by Wiley Publishing, Inc., Indianapolis, Indiana

No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning or otherwise, except as permitted under Sections 107 or 108 of the 1976 United States Copyright Act, without the prior written permission of the Publisher. Requests to the Publisher for permission should be addressed to the Permissions Department, John Wiley & Sons, Inc., 111 River Street, Hoboken, NJ 07030, (201) 748-6011, fax (201) 748-6008, or online at `http://www.wiley.com/go/permissions`.

Trademarks: Wiley, the Wiley Publishing logo, For Dummies, the Dummies Man logo, A Reference for the Rest of Us!, The Dummies Way, Dummies.com, Making Everything Easier, and related trade dress are trademarks or registered trademarks of John Wiley & Sons, Inc. and/or its affiliates in the United States and other countries, and may not be used without written permission. AMD, the AMD Arrow logo, AMD Opteron, AMD Virtualization, AMD-V, and combinations thereof are registered trademarks of Advanced Micro Devices, Inc. HyperTransport is a licensed trademark of the HyperTransport Technology Consortium. All other trademarks are the property of their respective owners. Wiley Publishing, Inc., is not associated with any product or vendor mentioned in this book.

LIMIT OF LIABILITY/DISCLAIMER OF WARRANTY: THE PUBLISHER AND THE AUTHOR MAKE NO REPRESENTATIONS OR WARRANTIES WITH RESPECT TO THE ACCURACY OR COMPLETENESS OF THE CONTENTS OF THIS WORK AND SPECIFICALLY DISCLAIM ALL WARRANTIES, INCLUDING WITHOUT LIMITATION WARRANTIES OF FITNESS FOR A PARTICULAR PURPOSE. NO WARRANTY MAY BE CREATED OR EXTENDED BY SALES OR PROMOTIONAL MATERIALS. THE ADVICE AND STRATEGIES CONTAINED HEREIN MAY NOT BE SUITABLE FOR EVERY SITUATION. THIS WORK IS SOLD WITH THE UNDERSTANDING THAT THE PUBLISHER IS NOT ENGAGED IN RENDERING LEGAL, ACCOUNTING, OR OTHER PROFESSIONAL SERVICES. IF PROFESSIONAL ASSISTANCE IS REQUIRED, THE SERVICES OF A COMPETENT PROFESSIONAL PERSON SHOULD BE SOUGHT. NEITHER THE PUBLISHER NOR THE AUTHOR SHALL BE LIABLE FOR DAMAGES ARISING HEREFROM. THE FACT THAT AN ORGANIZATION OR WEBSITE IS REFERRED TO IN THIS WORK AS A CITATION AND/OR A POTENTIAL SOURCE OF FURTHER INFORMATION DOES NOT MEAN THAT THE AUTHOR OR THE PUBLISHER ENDORSES THE INFORMATION THE ORGANIZATION OR WEBSITE MAY PROVIDE OR RECOMMENDATIONS IT MAY MAKE. FURTHER, READERS SHOULD BE AWARE THAT INTERNET WEBSITES LISTED IN THIS WORK MAY HAVE CHANGED OR DISAPPEARED BETWEEN WHEN THIS WORK WAS WRITTEN AND WHEN IT IS READ.

For general information on our other products and services, please contact our Customer Care Department within the U.S. at 877-762-2974, outside the U.S. at 317-572-3993, or fax 317-572-4002. For details on how to create a custom *For Dummies* book for your business or organization, contact `bizdev@wiley.com`. For information about licensing the *For Dummies* brand for products or services, contact `BrandedRights&Licenses@Wiley.com`.

ISBN: 978-1-118-01862-0

Manufactured in the United States of America

10 9 8 7 6 5 4 3 2 1

Publisher's Acknowledgments

Project Editor: Jennifer Bingham

Editorial Manager: Rev Mengle

AMD Contributors: Jeff Jones, Jeff Underhill, Scot Schultz, Jacob Liberman, Guy Ludden

These materials are the copyright of Wiley Publishing, Inc. and any dissemination, distribution, or unauthorized use is strictly prohibited.

Table of Contents

These materials are the copyright of Wiley Publishing, Inc. and any dissemination, distribution, or unauthorized use is strictly prohibited.

These materials are the copyright of Wiley Publishing, Inc. and any dissemination, distribution, or unauthorized use is strictly prohibited.

Introduction

The quest to capture and manipulate the world in digital form began when the first vacuum tube computers were used to calculate trajectories and crack codes. Viewed through today's technology, these problems seem trivial and could easily be solved on your cell phone and PDA. The quest to capture and compute continues to this day. The power to model and manipulate our world in silicon has enabled vast changes in how we conduct science, business, and even our everyday lives. From the next scientific breakthrough to new and better products to a greener world, High Performance Computing (HPC) is playing a central role in all these efforts.

In simple terms, HPC enables us to first model then manipulate those things that are important to us. HPC changes everything. It is too important to ignore or push aside. Indeed, HPC has moved from a selective and expensive endeavor to a cost-effective technology within reach of virtually every budget. This book will help you to get a handle on exactly what HPC does and can be.

High Performance Computing For Dummies, 2nd AMD Special Edition is intended for anyone who has heard about the many benefits of using HPC (such as streamlining processes or saving money). This book explains what HPC is and shows how it can help you or others within your company.

About This Book

HPC is a powerful technique, but it is not available at the corner office supply center (yet) nor can it solve every product or process problem you have. It requires a dedicated effort and commitment to new and exciting technologies. Finding the right partners and technologies is critically important. Much of this book is designed to help with this process. Be forewarned: Results may not be instantaneous and it could take more than

These materials are the copyright of Wiley Publishing, Inc. and any dissemination, distribution, or unauthorized use is strictly prohibited.

a few business quarters before your efforts pay off. However, there are plenty of HPC success stories out there, and perhaps yours is next.

It may be hard to imagine, but we're at the beginning of the High Performance Computing era. This book is an attempt to give you a high level snapshot where things are today. Trying to predict what will happen next year or even next week is anyone's guess. Right now, there may be some research group designing a new product or on the verge of a great discovery because they can hold and manipulate objects in the digital domain. HPC will undoubtedly lead to future breakthroughs. Along with these successes, expect that there will be novel ways in which to derive a business advantage using HPC. Don't be afraid to let your mind wander.

This book was sponsored by and written in collaboration with AMD, and revised and updated as a second edition by the HPC gurus at AMD. A special thanks to Oracle for helping with the first edition.

Icons Used in This Book

Throughout the margins of this book, you find several helpful little icons that guide you to certain types of information:

Yes, I know. The whole book is full of technical stuff. But *this* technical stuff tends to be a little off the beaten track. You don't have to read it to follow along, but if you do, you earn extra credit.

This icon alerts you to key concepts that you might want to commit to memory.

Flagging a tip is just a nice way to warn you that if you're not careful, you may end up overlooking something important that can save you time or money.

These materials are the copyright of Wiley Publishing, Inc. and any dissemination, distribution, or unauthorized use is strictly prohibited.

Chapter 1

HPC: It's Not Just for Rocket Scientists Any More

In This Chapter

- Looking at the forms of HPC
- Examining who uses HPC today
- Explaining who ought to be using it

Mention the word *supercomputer* to someone and they automatically think of monstrously complicated machines solving problems no one really understands. Maybe they think of flashing lights and some super intelligence that can beat humans at chess or figure out the meaning of life, the universe, and everything. (It's "42" for those who are interested.)

Back in the day, this was not an altogether untrue view of supercomputing. With an entry fee of at least seven figures, supercomputing was for the serious scientists and engineers who needed to crunch numbers as fast as possible.

Today we have a different world. The custom supercomputer of yesteryear has given way to commodity-based supercomputing, or what is now called High Performance Computing (HPC). In today's HPC world, it is not uncommon for the supercomputer to use the same hardware found in Web servers and even desktop workstations. The HPC world is now open to almost everyone because the cost of entry is at an all-time low.

These materials are the copyright of Wiley Publishing, Inc. and any dissemination, distribution, or unauthorized use is strictly prohibited.

HPC mud flaps

HPC can show up in the most unexpected places — like under a truck. Large trucks spend a lot of time moving. Pushing against the air while driving takes energy (in this case gasoline) and energy costs money. Almost all trucks have some kind of mud flaps to prevent road dirt and debris from hitting the truck. A midsized truck maker wondered how much it cost to push those mud flaps through the air. Using HPC, they were able to determine that trimming and tapering the flaps can cut about $400 from a typical truck's annual gas bill. Although it might not sound like much for a single truck, when you have a fleet of 1,000 trucks, those savings add up very quickly. The company has now moved far beyond mud flaps and is using HPC to help increase the efficiency of their truck designs, thus saving even more money.

To many organizations, HPC is now considered an essential part of business success. Your competition may be using HPC right now. They won't talk much about it because it's considered a competitive advantage. Of one thing you can be sure, however; they're designing new products, optimizing manufacturing and delivery processes, solving production problems, mining data, and simulating everything from business process to shipping crates all in an effort to become more competitive, profitable, and "green." HPC may very well be the new secret weapon.

The Forms of HPC

Ask ten people what HPC is and you will get ten different answers. Applying compute cycles to a problem can be done in many ways. Currently, there seem to be four modes in which you can obtain the cycles needed for typical HPC problems:

- **The commodity HPC cluster:** Over the last ten years, the HPC cluster has disrupted the entire supercomputing market. Built from standard off-the-shelf servers and high speed interconnects, a typical HPC system can deliver industry-leading, cost-effective performance. A typical cluster can employ hundreds, thousands, and even tens of thousands of servers all working together on a single problem (this is the high tech equivalent of a "divide and

These materials are the copyright of Wiley Publishing, Inc. and any dissemination, distribution, or unauthorized use is strictly prohibited.

conquer" approach to solving large problems). Because of high performance and low cost, the commodity cluster is by far the most popular form of HPC computing. Also keep in mind the compatibility advantage — x86 commodity servers are ubiquitous.

- **Dedicated supercomputer:** In the past, the dedicated supercomputer was the only way to throw a large number of compute cycles at a problem. Supercomputers are still produced today and often use specialized non-commodity components. Depending on your needs, the supercomputer may be the best solution although it doesn't offer the commodity price advantage.
- **HPC in the cloud:** This method is relatively new and employs the Internet as a basis for a cycles-as-a-service model of computing. The compute cycles in question live in the cloud *somewhere* allowing a user to request remote access to resources on-demand. An HPC cloud provides dynamic and scalable resources (and possibly virtualization) to the end-user as a service. Although clouds can be cost effective and allow HPC to be purchased as a pay-as-you-go expense and not a capital asset, it also places some layers between the user and hardware that may reduce performance.

The Top500

No, the Top500 isn't a car race. It is a list of the world's fastest computers. Of course, some background is needed. To get on the Top500 list, you must run a single benchmark program on your HPC system and submit it to the Top500 organization. The list is created twice a year and includes some rather large systems. Not all Top500 systems are clusters, but many of them are built from the same technology. Of course, as with all lists, there are some limitations. First, the list is for a single benchmark (HPL or High Performance Linpack). Results for other benchmarks may shuffle the standings. Second, the list only includes those systems that were submitted for consideration. There may be HPC systems out there that are proprietary or not interested in the Top500 ranking.

Yet despite these limitations, the Top500 list is the wealth of historical data. The list was started in 1993 and has data on vendors, organizations, processors, memory, and so on for each entry in the list. You can view the list (past and present) by going to `http://www.top500.org/`.

These materials are the copyright of Wiley Publishing, Inc. and any dissemination, distribution, or unauthorized use is strictly prohibited.

- **Grid computing:** Grid is similar to cloud computing, but requires more control by the end-user. Its main use is academic projects where local computer resources are connected and shared on a national and international level. Some computational grids span the globe while others are located within a single organization. A grid can be specialized to a particular application; however, it is more common that a single grid will run many different applications. They are also much more dependent on specialized libraries, also known as *middleware*.

Who Uses HPC Today?

The worldwide HPC market is growing rapidly. According to IDC, the total HPC market was $10 billion in 2007 and is expected to hit $11.7 billion around 2012. In terms of market share, IDC defines price bands by system size as shown in Table 1-1.

Table 1-1 2007 HPC Market Share as Function of System Cost

System Size	*Market Share*
Supercomputers (Over $500K)	$2.7B
Technical Divisional ($250K–$500K)	$1.6B
Technical Departmental ($100K–$250K)	$3.4B
Technical Workgroup (under $100K)	$2.4B

Successful HPC applications span many industrial, government, and academic sectors. The following is a list of major areas where HPC has a significant presence:

- **Bio-sciences and the human genome:** Drug discovery, disease detection/prevention
- **Computer aided engineering (CAE):** Automotive design and testing, transportation, structural, mechanical design
- **Chemical engineering:** Process and molecular design

These materials are the copyright of Wiley Publishing, Inc. and any dissemination, distribution, or unauthorized use is strictly prohibited.

- **Digital content creation (DCC) and distribution:** Computer aided graphics in film and media
- **Economics/financial:** Wall Street risk analysis, portfolio management, automated trading
- **Electronic design and automation (EDA):** Electronic component design and verification
- **Geosciences and geo-engineering:** Oil and gas exploration and reservoir modeling
- **Mechanical design and drafting:** 2D and 3D design and verification, mechanical modeling
- **Defense and energy:** Nuclear stewardship, basic and applied research
- **Government labs:** Basic and applied research
- **University/academic:** Basic and applied research
- **Weather forecasting:** Near term and climate/earth modeling

Cultivating the HPC edge

Interested in learning more about using HPC as a competitive tool? You aren't alone. As a mater of fact, the Council on Competitiveness (CoC) wants to help you! Who is the CoC? Glad you asked, they're a group of corporate CEOs, university presidents, and labor leaders committed to the future prosperity of all Americans and enhanced U.S. competitiveness in the global economy through the creation of high-value economic activity in the United States. In other words, they want to help your business succeed.

The CoC is a nonpartisan, nongovernmental organization based in Washington, D.C. They shape the debate on competitiveness by bringing together business, labor, academic, and government leaders to evaluate economic challenges and opportunities. The High Performance Computing (HPC) Initiative is intended to stimulate and facilitate wider usage of HPC across the private sector to propel productivity, innovation, and competitiveness. For more information, case studies, and surveys check out `www.compete.org`.

These materials are the copyright of Wiley Publishing, Inc. and any dissemination, distribution, or unauthorized use is strictly prohibited.

The list could be longer as more and more areas are finding HPC useful as a tool to better understand their science, market, products, and customers. As pioneers, the government and academic sectors have been successfully using and validating HPC methods for well over a decade. The secret is out and the same advantages enjoyed by leading researchers can be had by almost anyone.

Who Should Be Using HPC?

The easy answer to this question is: anyone interested in faster times to solution, better science, informed decisions, more competitive products, and how those things can drive profits higher. High Performance Computing represents a tremendous competitive edge in the marketplace because it can give users the ability to quickly model and then manipulate a product or process to see the impact of various decisions before they are made. Consider a car as it moves through the air. Once it has been modeled in an HPC system, the end-user can look at the air flow at points that would be almost impossible with a physical model in a wind tunnel.

The possibilities are endless. Existing products and processes can be optimized to reduce cost, and new avenues of development can be explored at lower cost and with faster times to market. The question to ask is "What information would allow you to make better decisions and products?" Chances are HPC can help with the answer.

Of coffee, chips, and soap

HPC has made huge inroads in the government and educational sectors. Perhaps one of the leading industrial champions of HPC is Procter & Gamble (P&G). This fact sometimes surprises people because P&G is often seen as a low-tech supplier of household products. Quite the opposite is true. P&G is one of the pioneers of industrial HPC. They can boast of significant cost savings and innovation across many product lines, including Folgers coffee, Pringles potato chips, Tide detergent, and many more. Unlike its competitors, P&G has been very vocal about its success with HPC. And, according to P&G, the successes have been well worth the investment.

These materials are the copyright of Wiley Publishing, Inc. and any dissemination, distribution, or unauthorized use is strictly prohibited.

Chapter 2

Getting to HPC

In This Chapter

- Examining the commodity cluster
- Looking at what's available
- Getting to know the cluster
- Drilling down into some details

In the past when you wanted to deploy a supercomputer, you picked up the phone and said, "Hello Supercomputer Company, I want one of your model Z3 systems, in blue." Sure enough, in a few weeks (or months) you were the proud owner of a new supercomputer (complete with a seven figure price tag). There was also a good chance the company you bought it from would teach your personnel how to use it. There would be manuals to assist users and administrators as well. When something needed to be changed or updated it was tested before it was turned over to your system administrators. There was even a support line where you could get questions answered about the system.

Once upon a time, this was the face of supercomputing. The entire system was integrated from top to bottom by a single company. From a hardware standpoint, supercomputers employed a handful of specialized processors that were designed to quickly operate on batches of floating point numbers.

So how did HPC move from something that was once confined to the elite of computing to that which almost any organization can now cost effectively deploy?

These materials are the copyright of Wiley Publishing, Inc. and any dissemination, distribution, or unauthorized use is strictly prohibited.

Enter the Commodity Cluster

As nice as the integrated supercomputer was, it did have some drawbacks. For one, the price was rather steep — seven figures. There was no doubt that leading edge required high cost. The high cost also reflected the significant amount of custom engineering required for these systems. The extreme performance came from specialized processors and memory systems configured in such a way as to move numerical calculations through a vector pipeline as fast as possible.

As supercomputer system performance increased, so did the cost of designing and fabricating custom processors and other components. At the same time, the commodity computer market was experiencing rapid growth and maturation of x86 processors. Often considered office processors, no one really considered x86s to be good at floating point calculations. This presumption began to change as floating point units were integrated into x86 processors and each new generation delivered significant incremental performance.

Today it's common for each commodity system based on x86 processors to actually contain multiple processors combined into a single physical processor. When multiple processors are integrated into a single physical processor, the individual processors are referred to as cores, hence the buzz about multi-core processors these days.

In addition to getting faster, supercomputer-specific processors also became more expensive to manufacture. Indeed, the expense increased to the point that it really was no longer feasible to design and fabricate supercomputer-specific processors. In addition, for certain problems it was found that by breaking a problem up into smaller sub-problems, collections of commodity processors could achieve performance levels that could rival those of their supercomputing kin. Indeed, as commodity processors from companies like AMD improved, collections (or *clusters*) of servers (each server in a cluster is commonly referred to as a *node*) began to outpace even the fastest supercomputers. The move to commodity hardware had begun.

At the same time, there was a small but growing open source software movement afoot. Many of the software tools needed to support a UNIX-like environment were freely available

These materials are the copyright of Wiley Publishing, Inc. and any dissemination, distribution, or unauthorized use is strictly prohibited.

under the GNU software license. The final piece of the software puzzle was the advent of the Linux operating system. Linux was essentially a plug-and-play replacement for UNIX — the then de facto supercomputing operating system. In addition, Linux didn't require a license fee and wasn't encumbered by restrictive intellectual property agreements.

Another enabler of the commodity cluster was the Internet. The Internet helped in two ways. First, it ignited the demand for Web servers whose compact data center design could offer a low-cost building block for commodity HPC clusters. And second, it opened up an avenue for international collaboration, helping solve many of the early HPC issues. Entire communities developed around various approaches to commodity-based HPC. Internet access to open software projects fueled this rapid growth and adoption.

By the turn of the millennium, the cost of entry into the supercomputing market had gone from seven figures to the low five figures. More people could now enter the HPC arena and have access to a level of computing previously reserved for the technical elite. Typical price-to-performance ratios, key metrics of the worth of an HPC system, changed as well. Many early adopters were reporting at least a factor-of-ten improvement of price-to-performance. Thus, many more problems could be explored, understood, and ultimately solved using HPC methods.

It's about Choice

As HPC cluster solutions began to appear, one thing became painfully obvious. The components required to create a commodity cluster would likely come from many different sources, without a computing company taking responsibility for the whole cluster. For example, the compute nodes could come from one vendor, and the interconnect from another. If you added a large storage component, then yet another vendor could be attending your cluster party. The software was just as scattered. There were many open software packages available to HPC users. Popular Linux distributions included many of the necessary software packages; however, successful HPC systems often required some blend of specialized open source and commercial packages.

These materials are the copyright of Wiley Publishing, Inc. and any dissemination, distribution, or unauthorized use is strictly prohibited.

Such variety can be both good and bad. On the good side, users can control what is (and is not) bought, and can optimize HPC investments for their specific needs. Unlike the large monolithic supercomputers, there's little need to buy technology you don't use or want. In addition, the price pressures from the commodity market ensure that hardware is both readily available and cost effective. In other words, compared to the days of old, commodity HPC today is definitely a bargain!

The other side of the coin is not so popular, however. Elite users have no problem specifying and designing HPC clusters, but those who are not as technically adroit are not as lucky. The right hardware and software choices could mean the difference between success and failure.

"Getting it right" is often seen as an impediment to cluster adoption. In addition to design, there is also the interoperability issue. There are a handful of vendors who specialize in cluster design and construction; however, integrating a large amount of hardware and software can lead to compatibility issues. Because the integrators are often not the source of the hardware and software components, identifying the appropriate person for "problem ownership and resolution" can be tricky. For this reason, choosing the right integrator/vendor with a good vertical product line can be very important.

Choosing the right vendor (with the right products) ensures that your HPC system "just works." This is more important today than ever before. Processors, memory, and networking are getting faster all the time. It's important to ensure that all the components of your environment work together and no one area becomes a bottleneck.

What Does a Cluster Look Like?

Trying to pin down what a cluster actually looks like is rather difficult. The nature of HPC cluster computing mandates that users consider a solution for their specific application needs. Since your needs are likely to be different from everyone else's, it is very likely that your cluster will be configured differently than everyone else's. The differences can be small or large depending on your application and users' needs.

These materials are the copyright of Wiley Publishing, Inc. and any dissemination, distribution, or unauthorized use is strictly prohibited.

There are, however, some common features that all clusters share. The first is a head or master node. This node is normally a gateway to a shared network where users "log in" to the cluster. The head node has one or more networks with which it communicates to the worker (or *compute*) nodes. These networks are private and are only accessible inside the cluster.

All clusters have worker nodes that do the bulk of the computing. These nodes are almost always identical throughout the cluster (although there is no strict requirement that all nodes need to be the same). Indeed in some cases it is advisable to configure your cluster with different compute nodes. This all depends on what applications you are trying to run. For example, some applications require more local memory than others, while some require a specific processor architecture in order to run at best performance.

In terms of networks, a cluster may have as little as one private network, which is usually Gigabit Ethernet (GigE), although InfiniBand is becoming a popular option. Almost all servers have at least two GigE connections on the motherboard and therefore all clusters have at least one private GigE network. A cluster will often generate three types of traffic:

- Computation traffic between compute nodes.
- File system traffic — often from an NFS (Network File System) server. (But not always. Direct attach storage has its advantages.)
- Administration traffic that provides node monitoring and job control across the cluster.

Depending on your applications, compute and/or file system traffic may dominate the cluster network and cause the nodes to become idle. For this reason, additional networks are added to increase overall cluster network throughput. In many cases, a high speed interconnect is used for the second network. Currently the two most popular choices for this interconnect are 10 Gigabit Ethernet (10GigE) and InfiniBand. As a lower-cost solution it is possible to use a second GigE network as well; however, this would not offer anywhere near the network performance of 10GigE or InfiniBand.

These materials are the copyright of Wiley Publishing, Inc. and any dissemination, distribution, or unauthorized use is strictly prohibited.

Figure 2-1 shows a "typical" cluster. There's a head node that may contain a large amount of storage that is shared via the network by all the worker nodes (using some form of Network File System). A varying number of worker nodes communicate with each other and the head node over the available private networks. The number of worker nodes can be quite large.

There are a few variations to be considered with any cluster design. The head node may be broken into several (or many) nodes. There can be multiple file serving nodes, multiple user login nodes, and separate administration nodes.

Compute nodes (or servers), networking gear, and storage devices are all placed in racks to help organize clusters and allow for proper ventilation when numerous systems are installed.

Note that the type and number of processors, amount of memory, and even the presence of a local storage devices are all determined by the user.

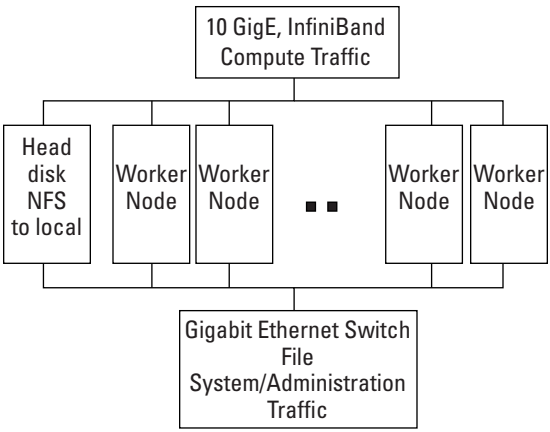

Figure 2-1: A typical cluster configuration.

Currently, there is a trend to move away from individual 1U "pizza box" nodes to the increasingly popular bladed systems. A blade system allows for shared/redundant power and cooling and management while improving the node density.

These materials are the copyright of Wiley Publishing, Inc. and any dissemination, distribution, or unauthorized use is strictly prohibited.

A blade-based cluster can also reduce maintenance costs and improve reliability.

Clusters are often broken into two categories: capability and capacity. A *capability* cluster is designed to handle (or be capable of handling) large compute jobs that may employ every node in the cluster. *Capacity* clusters, on the other hand, are those that are used to deliver a certain amount of computing capacity to the end users. For instance, a capacity cluster may support hundreds of users running any number of jobs that require a smaller number of nodes. Most clusters are used in a capacity fashion.

If You Need Speed

High performance clusters are used where time to solution is important. They are also used in cases where a problem is so big it can't "fit" on one single computer. To increase computing throughput, HPC clusters are used in a variety of ways.

The first and easiest way is to allow the cluster to act as a compute farm. Instead of running a job on a local workstation, it is submitted to the cluster for execution. The cluster will manage the resources needed for the job and assign the job to a work queue. When the resource (for instance, a server) is available, the job gets executed and the results are returned to the user. Users who need to run many similar jobs with different parameters or data sets find clusters ideal for this kind of work. They can submit hundreds of jobs and allow the cluster to manage the workflow. Depending on the resources, all the jobs may run at the same time or some may wait in the queue while other jobs finish. This type of computing is local to a cluster node, which means the node doesn't communicate with other nodes, but may need high speed file system access. An example of this usage mode is given in Figure 2-2. Each job can be from a different user or the same user. The jobs can run concurrently, thus increasing computing throughput. Even though more jobs can be run in the same time period, they never run any faster than normal. Because each job is independent, there is no communication between the jobs.

These materials are the copyright of Wiley Publishing, Inc. and any dissemination, distribution, or unauthorized use is strictly prohibited.

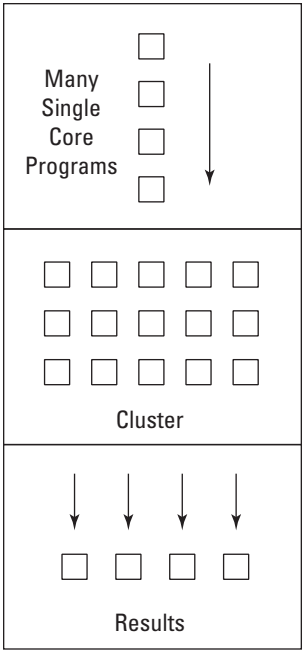

Figure 2-2: A cluster used to simultaneously run many core independent jobs (each small square box is a cluster node).

Another way to use clusters is to break large HPC jobs up into smaller sub-jobs and run each sub-job on different nodes. This process is done at the software level and, thus, programs may need to be changed to run in this fashion. The most popular method for creating cluster programs is to use the MPI (Message Passing Interface) library and insert communication calls in the program. Because these sub-jobs need to communicate with each other, they can create a large amount of compute traffic within the cluster. For this reason, high performance networks are used to handle this level of traffic. It is not uncommon to use many hundreds of nodes for a single program. Again, the program must request the resources (for example, "I need 100 processor cores") from the cluster and when the resources are ready, the cluster will run the program. This type of program is called *parallel* because it uses many cores to collaborate toward solving the same problem. An example of a parallel program running on a cluster is given in Figure 2-3. The program employs all the processors and

These materials are the copyright of Wiley Publishing, Inc. and any dissemination, distribution, or unauthorized use is strictly prohibited.

must communicate with the other nodes. The parallel program will run much faster than using a single node; however, the programmer must explicitly create a parallel program.

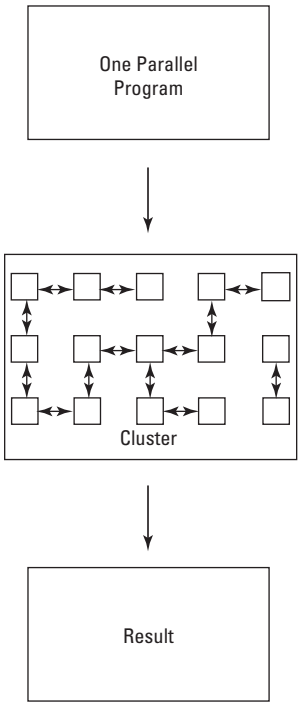

Figure 2-3: A cluster used to run a parallel program. (Each small square box is a cluster node).

Both modes of computation, compute farm and parallel, can operate on a cluster at the same time. Of course if a parallel program needs all the nodes, nothing else can run, but this is not often the case. There is usually scheduling software that handles the allocation policies for the cluster.

Breaking up a single program into parts to run across multiple cores is a rather obvious way to increase speed. Like building a house, the more workers you have, the faster you can complete the job. Some parts of the construction are highly *parallel*, like framing the walls, painting, applying siding, and so on. Other steps may depend on one worker and cause the others to wait. For instance, you can't finish the walls until the

These materials are the copyright of Wiley Publishing, Inc. and any dissemination, distribution, or unauthorized use is strictly prohibited.

electrical work is done. Thus, there are limits to the number of workers you can add before there is no increase in speed at which you build a house.

Parallel computing works in exactly the same way. Computer programs normally have "slow steps" that limit how fast they can actually run. This effect is called Amdahl's law (see the sidebar of the same name) and it essentially tells you how much speed-up you can expect from a given computer program when it is run across multiple cores.

Of Cores, Threads, and Nodes

When discussing HPC, terms like *cores, threads,* and *nodes* often get used in very informal ways. In this book, I refer to *cores* as processing units within a modern day processor. A typical motherboard may hold 1, 2, or 4 processors and is contained in a *node*. Thus, the total *cores per node* is a function of how many processors are present and how many cores are in each multi-core processor. A typical node may have 2 processors, each with 12 cores for a total of 24 cores per node. (A *node* can also be defined as one operating system instance.)

From a software standpoint, programs on a multi-core node are run in a Symmetric Multi-Processing (SMP) mode. This designation means that multiple programs (or processes) can be running at the same time on various cores. It also allows for a single program to use multiple cores by pulling the program apart into *threads*. Threaded programming is parallel programming designed for a single SMP node. (It is possible, but not advisable, to run threads between nodes — unless you have specialized hardware and software.) A threaded program thus starts out as a single program but soon breaks itself into separate but highly connected parts to take advantage of the multiple cores in the SMP motherboard node.

Thread programming for HPC can be accomplished using the OpenMP model — supported by almost all compilers. Like threads, OpenMP isn't designed for "off node" operation, whereas MPI is. A basic rule is to use MPI between server nodes and OpenMP/threads within a single server node. And, keep in mind, parallel MPI programs will run on SMP nodes as well. Finally, it is possible to create hybrid programs that use both OpenMP and MPI!

These materials are the copyright of Wiley Publishing, Inc. and any dissemination, distribution, or unauthorized use is strictly prohibited.

Amdahl's law

In the early days of parallel computing, Gene Amdahl threw a bucket of cold water on those who got overly excited about piling on processors. Amdahl showed you can't keep throwing more and more cores at a problem and expect it to keep speeding up. His law works as follows. The parallel speedup of any program is limited by the time needed for any sequential portions of the program to be completed. For example, if a program runs for ten hours on a single core and has a sequential (non-parallelizable) portion that takes one hour, then no matter how many cores you devote to the program it will never go faster than one hour. Seems obvious, and it is. Amdahl's law affects the scalability of a program as well.

What about the Data?

One final note about parallel computing. In addition to the parallel computation shown in Figure 2-3, there may also be a need for parallel data access. Imagine if all the nodes in Figure 2-3 were required to read and write data from the same file at the same time. A bottleneck would develop and limit the speed of your program. For this reason, parallel file systems have been developed. These solutions often involve additional hardware and software. I take a closer look at this issue in Chapter 3.

These materials are the copyright of Wiley Publishing, Inc. and any dissemination, distribution, or unauthorized use is strictly prohibited.

These materials are the copyright of Wiley Publishing, Inc. and any dissemination, distribution, or unauthorized use is strictly prohibited.

Chapter 3

HPC Building Blocks

In This Chapter

- Getting the best hardware for the job
- Making sure your software fits the bill
- Looking at cluster tool kits

HPC systems are built from many components. There are some common elements, but clusters may employ a wide range of hardware and software solutions. This chapter goes over some of the important cluster components.

Choosing Cluster Hardware

Obviously if you're looking into HPC, you're going to have to make sure you get hardware that can handle it! This section goes over what you need to make sure you have in place in order to set up an HPC system.

Crunching numbers: Processors and nodes

The processor is the workhorse of the cluster. And keeping the workhorse busy is the key to good performance. Parallel programs are often distributed across many nodes of the cluster (for more info on this, see Chapter 2). However, multi-core has changed this situation a bit. Cluster nodes may now have 16 or even 24 cores per node (for example, servers based on the AMD Opteron™ 6100 Series processors). It is possible for HPC applications to fit on a single cluster node. (Sometimes this helps performance and sometimes it hurts performance — it

These materials are the copyright of Wiley Publishing, Inc. and any dissemination, distribution, or unauthorized use is strictly prohibited.

all depends on the application architecture, memory requirements, and so on.)

The choice of processor is very important because cluster installations often rely on scalable processor performance as well as platform longevity. Advances in the x86 architecture, such as simultaneous 32/64 bit operations and integrated memory controllers; along with innovations similar to AMD's HyperTransport™ technology, have propelled commodity processors to the forefront of HPC. As an example, in a recent cluster procurement, Lawrence Livermore, Los Alamos, and Sandia National Labs all chose AMD Opteron processor-based clusters. As far as power users go, these government guys like to think big. They deployed a total of 48,384 cores as part of the project in 2007. That would be 12,096 Quad-Core AMD Opteron processors for a combined processing power of up to 438 TFLOPS. Announced in November of 2009, the Jaguar system at Oak Ridge National Lab exceeded 2 PFLOPS by upgrading their more than 1 PFLOP of performance from Quad-Core AMD Opteron processors to the Six-Core AMD Opteron processors. (See the "What the flip are FLOPS?" sidebar.)

Depending on the design of the cluster, nodes can be *fat* (lots of cores, disk, and memory), *thin* (small number of cores and memory), or anything in between. Some applications work well in either type of node while others work best with a particular configuration.

Pay attention to the amount of memory. In general, "more cores per node" means more memory per node because each core could be asked to run a totally separate program. Many HPC cores require a large amount of memory. Check your applications memory requirements and size your nodes appropriately. This is something sometimes overlooked when upgrading to greater core-count processors — if you don't increase the memory capacity, you effectively reduce the amount of memory per core, which could in turn limit some of the advantages from upgrading in the first place. Another point to consider is the speed of the system memory for maximum performance potential.

Another feature to consider for nodes is system management. The Intelligent Platform Management Interface (IPMI) specification defines a set of common interfaces to a computer system.

These materials are the copyright of Wiley Publishing, Inc. and any dissemination, distribution, or unauthorized use is strictly prohibited.

System administrators can use IPMI to monitor system health and manage the system. Often IPMI features can be managed over a network interface, which is very important with large numbers of nodes often located in a remote data center.

The communication: Interconnects

In order to keep all those nodes busy, a good *interconnect* (a port that attaches one device to another) is needed. As always, it all depends on your application set. In general, most high performance cluster systems use a dedicated and fast interconnect.

High performance interconnects are usually rated by *latency,* the fastest time in which a single byte can be sent (measured in nanoseconds or microseconds), and *bandwidth,* the maximum data rate (measured in Megabytes or Gigabytes per second). There are other numbers that help as well, like the N/2 packet size. This number is the size of the packet that reaches half the single direction bandwidth of the interconnect (a measure of how fast the throughput curve rises). The smaller the number, the more bandwidth (speed) that small packets will achieve. A final number to look at is the *messaging rate*. This tells you how many messages per second an interconnect can send and is important for multi-core nodes because many cores must share the interconnect.

Although the numbers listed are good for sizing up an interconnect, the ultimate test is your application. In most cases, your application will communicate using MPI (Message Passing Interface) libraries. This software is a communications layer on top of the hardware. MPI implementations vary and the true test is to run a few benchmarks. (For more on MPI, see the section "HPC glue: Cluster software.")

Bandwidth versus latency

Latency is the time required to move data from one point to another. Bandwidth is the data transmission throughput, or the maximum amount of information (bits/second) that can be transmitted

These materials are the copyright of Wiley Publishing, Inc. and any dissemination, distribution, or unauthorized use is strictly prohibited.

Fast interconnects also require switching so that any node can talk to any other node. One way in which high-end switches are rated, in addition to the point-to-point numbers mentioned previously, is *bi-section bandwidth*. This rates how well the switch supports many simultaneous conversations. A good bi-sectional bandwidth will allow all the nodes to be communicating at the same time. Although not a necessity for all clusters, large parallel applications usually require a good bi-sectional bandwidth in order to keep the processors busy. Keep in mind that large clusters use switch hierarchies that are composed of many smaller switching units. (This can be an important factor when sizing the number of nodes; extra layers of switching increase costs and network latency because traffic must traverse more "hops" to get from point A to point B.)

In terms of available technology, high performance computing networks are generally being deployed using two major technologies: InfiniBand (IB) and 10 Gigabit Ethernet. Of course, if your applications don't require a large amount of node-to-node communication, standard Gigabit Ethernet (GigE) is a good solution. GigE is often on the motherboard and there are very high density/performance GigE switches available.

InfiniBand has been successfully deployed in both large and small clusters using blades and 1U servers. It has a freely available industry backed software stack (`www.openfabrics.org`) and is considered by many to be among the best interconnects for clustering due to its low latency and high throughput. Another key feature of InfiniBand is the availability of large multi-port switches that provide exceptionally low latency. A large port density allows for less cabling and fewer sub-switches when creating large clusters.

The interconnect bottom line is as follows. Understanding application communication requirements allows you to optimize your price-to-performance. That is, your budget will be a balance of "nodes to network" numbers. A faster network means you may have to buy fewer nodes and could have less overall computing power. A slower network means you can buy more nodes and acquire more computing power. The balance between these two sides must be based on your application needs.

These materials are the copyright of Wiley Publishing, Inc. and any dissemination, distribution, or unauthorized use is strictly prohibited.

What the flip are FLOPS?

The term *FLOPS* is used quite a bit in HPC. Like any other industry, they have their share of acronyms. FLOPS stands for *floating point operations per second*. A *floating point operation* is a mathematical operation (addition, multiplication, square root, and so on) on a number with a decimal point and exponent (for example, 1.2345×10^3). To put these numbers in perspective, a simple desktop calculator will deliver 10 FLOPS and a typical desktop workstation computer can achieve somewhere between 20 and 30 GFLOPS (1×10^9 FLOPS). The fastest computers in the world have recently broken the 2 PFLOPS (2×10^{15} FLOPS) barrier.

Though not always stated, FLOPS can have different levels of precision and can vary depending on the program used to measure them. In general, there are two types of precision (single and double) used in HPC. The difference is due to the amount of memory used to store the numbers. Obviously, a double precision floating point number has more digits than a single precision number (for those curious readers, single precision is 32-bits and double precision is 64-bits). As a result, double precision calculations take longer than single precision. Keep this in mind when you see someone potentially overselling a FLOPS rating.

Remembering the storage

Storage is often the forgotten sibling of the HPC cluster. Almost all clusters require some form of high performance storage. The simplest method is to use the head node as an NFS server. Even this simple solution requires that the head node have some kind of RAID sub-system. Often, NFS doesn't scale to large numbers of nodes. For this reason, there are alternative storage designs available for clusters. Most of these designs are based on a parallel file system that in turn defines the type of storage hardware that will be required. A good example is the Lustre parallel file system. Lustre is a tested open-source solution that provides scalable I/O to clusters.

HPC applications notoriously create large amounts of data, and ensuring that an archiving system is available is crucial to many data centers. Moving data to tried-and-true backup technologies (magnetic tape) is an important step in protecting your HPC investment. A good archiving system will automatically move data from one storage device to another based upon policies set by the user.

These materials are the copyright of Wiley Publishing, Inc. and any dissemination, distribution, or unauthorized use is strictly prohibited.

The HPC optimized appliance

Dents and dings can cost. As any manufacturer knows, dented products don't sell at full price, if at all. Preventing product damage has a direct effect on the bottom line. Consider a leading manufacturer of home appliances. Dents and dings, no matter how small, can render an expensive washing machine unsellable. When the company noticed a high percentage of its washing machines were being dented between the factory and the retailer they turned to HPC for some help. By simulating the transit process, they not only identified the problem, but redesigned the packaging materials and the clamps that were used as part of the packaging. Fewer dents, more money.

Another area to watch closely is Flash. Flash modules can be integrated directly onto motherboards, or integrated using the PCIe bus. Keep in mind that a poor storage sub-system can slow a cluster down as much as, if not more than, a slow interconnect. Underestimating your storage needs and assuming NFS "will just handle it" may be a mistake.

A final aspect of the storage hardware is making sure it will work and integrate with the other cluster hardware. Sometimes storage sub-systems are purchased from separate vendors, causing incompatibilities that can often be traced to software driver issues or even physical incompatibilities. Many users avoid these issues by utilizing a top-tier vendor with well designed and tested sub-systems for both their compute and storage needs. The key here is to manage the data throughout its entire life cycle, from the cluster to home directories to bulk storage to tape.

Racking and stacking

Compute nodes can take the form of 1U servers or blade systems (for more on this, see Chapter 2). The choice is largely one of convenience. Blade systems often have a slightly higher acquisition cost but can offer much more manageability, better density, and some power and cooling redundancy. Regardless of which form of computer node you choose, you should ensure that all cluster equipment is "rack mountable" and can be placed in standard 19-inch equipment racks. The

These materials are the copyright of Wiley Publishing, Inc. and any dissemination, distribution, or unauthorized use is strictly prohibited.

standard rack chassis can hold 42 U of equipment (1U equals 1.75 inches or 44.45 millimeters). It is very useful to map out your rack chassis space for all equipment and allow some extra for future enhancements.

Power and cooling

Power and cooling has gone from an overlooked expense to a critical factor in cluster procurements. A general rule of thumb for forecasting power and cooling cost is that the yearly cost to keep a cluster powered and cooled will equal roughly one third of the purchase price of the cluster itself. For this reason, looking for a green solution makes both environmental and economic sense. For instance, choosing a cluster that uses AMD Opteron™ 6100 Series HE processors, each with 65-watt ACP (Average CPU Power) ratings can be a very smart move. Factoring in lower idle power compared to similarly configured competing systems may just make it a brilliant move, especially when coupled with an AMD SR56xx series chipset, which provides up to 42 lanes of PCI Express 2.0 at an idle power of just 7.5 watts. The power consumption of the overall platform including chipsets, memory, network, and other components of each server node in your cluster could add up to quite an expense over time when paying your energy bills; be sure to keep this in mind as you plan to make the most return on your cluster investment.

Finding Cluster Software

To get your HPC system up and running, you need something to run on it. That's where software comes in. Linux is by far the most common operating system that HPC users choose and any other software needs to work well with Linux. Other options include Solaris, which is "open" and has full support for Linux binary compatibility and Microsoft's Windows HPC Server 2008 R2.

Operating systems

The predominant operating system for HPC can be summed up in one word: Linux. Prior to the advent of Linux, the HPC or supercomputing market used UNIX exclusively. Linux represents a plug-and-play alternative and doesn't add any

These materials are the copyright of Wiley Publishing, Inc. and any dissemination, distribution, or unauthorized use is strictly prohibited.

licensing fees for the compute nodes (which can be quite large in number). In addition to the Linux kernel, much of the important supporting software has been developed as part of the GNU project.

The GNU/Linux core software is open-source (see Chapter 5) and can be freely copied and used by anyone. There are, however, requirements to ensure source code is shared. The openness and "shareability" of GNU/Linux has made it an ideal HPC operating system. It has allowed HPC developers to create applications, build drivers, and make changes that would normally not be possible with closed source.

Virtually all Linux installations are done with a commercial or freely available software distribution package. Although the commercial availability of "free" software may seem puzzling, most commercial open source vendors use a support-based model. You're free to look at and alter the source code, but if you need support, you have to open your wallet.

Users may recognize some of the Commercial GNU/Linux distributions such as Red Hat, SUSE, and others. There are community versions (no support options) available as well. Red Hat Fedora, Open SUSE, and CentOS are examples of this approach. It should be noted that although these distributions are highly polished in their own right, they don't contain all the software needed to support an HPC cluster. Cluster distributions are available that fill this gap.

HPC glue: Cluster software

There are several types of software tasks that are needed to run a successful cluster. These tasks include administration, programming, debugging, job scheduling, and provisioning of nodes.

From a user's perspective, programming is perhaps the most important aspect of the cluster. The most important HPC tool for programming is probably MPI (Message Passing Interface). *MPI* allows programs to talk to one another over cluster networks (this allows individual nodes to coordinate their participation in the overarching task at hand). Without this software, creating parallel programs would be, and was in the past, a very custom (and likely time-consuming) process. Today, there are

These materials are the copyright of Wiley Publishing, Inc. and any dissemination, distribution, or unauthorized use is strictly prohibited.

both open and commercial MPI versions. The two most popular open MPIs are MPICH2 from Argonne Lab and the Open MPI project. Platform MPI is a high performance and production quality implementation of the Message Passing Interface (MPI) standard for both the Linux and Microsoft Windows operating systems. Platform MPI 8.0 combines the broad adoption and scalability of HP-MPI with the performance of Scali-MPI and is fully compliant with the MPI 1.2 and 2.2 standards.

In addition to MPI, programmers need compilers, debuggers, and profilers. The GNU software includes very good compilers and other programming tools; however, many users prefer to use professional compiler/debugger/profiler packages such as the Open64 Compiler Suite, and those offered by Oracle, (Oracle Solaris Studio 12 for Solaris and Linux), the Portland Group (PGI), and Intel. All vendors supply their own tools and cluster stack software.

Although users like to run programs, administrators are required to keep "the cluster lights on" as it were. There are tools to help manage cluster installation (See the section "Provisioning: Creating the cluster") and monitor cluster operation.

No free lunch

In order to use the combined power of an HPC cluster, your software needs to be made *parallel*. By the way, the same goes for multi-core as well. A typical computer program is written to run on a single CPU or core. It will not automatically use extra cores or nodes in the cluster because there is no "free lunch" in HPC. To run in parallel, one must change the internal workings of the program. There are several ways to accomplish this task. If you want to run your program on multiple cores, then using pthreads or OpenMP is a good solution. If, on the other hand, you want to run across multiple cluster nodes (and the multiple cores in the node) then using the MPI (Message Passing Interface) may be appropriate. In yet other cases, a blend of both may be the right solution. Regardless, the level of effort required depends on the program.

Many commercial ISV (Independent Software Vendor) codes are already parallel and will work out of the box on clusters. The same is true for many of the open source applications, although you're often responsible for building the open source applications yourself.

These materials are the copyright of Wiley Publishing, Inc. and any dissemination, distribution, or unauthorized use is strictly prohibited.

Sometimes, the best things in life are free!

The x86 Open64 compiler system is a high performance, production quality code generation tool designed for high performance parallel computing workloads. This suite offers advanced optimizations, multi-threading features, and support for vectorization, interprocedural analysis, loop transformations, and other functions to help accelerate development and tuning of C, C++, and Fortran applications targeting 32-bit and 64-bit Linux platforms. Learn more at `http://developer.amd.com/open64`.

File systems

Almost all clusters use the standard NFS file system to share information across nodes. This is a good solution; however, NFS wasn't designed for parallel file access (for instance, multiple processes reading and wiring to the same file). This limitation had become a bottleneck for HPC systems. For this reason, parallel file systems were developed.

One of the areas where the open GNU/Linux approach has served the HPC community is with file systems. There are a multitude of choices, all of which depend on your application demands. HPC file systems are often called "parallel file systems" because they allow for aggregate (multi-node) input and output. Instead of centralizing all storage on a single device, parallel file systems spread out the load across multiple separate storage devices. Parallel file systems are very often "designed" because they must be matched to a particular cluster.

One popular and freely available parallel file system is Lustre from Oracle. Lustre is a vetted, high-performance parallel file system. Other options include PVFS2, which is designed to work with MPI. Cluster file systems cover a large area. In addition to massive amounts of input, scratch, and checkpoint data, most HPC applications produce large amounts of output data that are later visualized on specialized systems.

These materials are the copyright of Wiley Publishing, Inc. and any dissemination, distribution, or unauthorized use is strictly prohibited.

One thing to keep your eye on is pNFS (NFS Version 4.1), which is designed for parallel NFS access. Most of the existing parallel file systems plan to support the pNFS specification and bring some standardization to the parallel file system arena. ZFS, a file system designed by Oracle, offers some exciting possibilities for HPC because it is the first 128 bit file system with many advanced features (for all intents and purposes 128 bits means it will never hit any storage size limits). ZFS is the fix for silent data corruption.

Sharing is caring: HPC resource schedulers

Clusters usually have lots of cores and lots of users. Sharing all these resources is no trivial matter. Fortunately the allocation of cores is done by scheduling software and not users (thus avoiding the ensuing chaos that could occur). Depending on the application, a scheduler may closely pack the cores (for instance, keeping them all together on a small number of nodes) or distribute them randomly across the cluster.

A cluster schedule works as follows. Like the old mainframe days, all users must submit their jobs to a work queue. As part of the submission processes, the user must specify the resources for the job (for instance, how many cores, how much memory, how much time, and so on). The resource scheduler then determines, based on site-wide policies, whose job gets to run next. Of course, it depends on when the resources become available and thus, as users often find out, being first in line doesn't necessarily mean being first to execute. The resource scheduler is a critical part of the cluster because it would be almost impossible to share resources without some form of load balancing tool. One important feature of the scheduling layer enables administrators to take resource nodes "off-line" for repair or upgrade and users are none the wiser — users rarely have a say in which nodes they're assigned. Additionally, if a node fails, the running jobs that use that node may fail, but other nodes keep working and the scheduler will work around the failed node.

These materials are the copyright of Wiley Publishing, Inc. and any dissemination, distribution, or unauthorized use is strictly prohibited.

HPC crash test dummies

These brave devices endure real crashes to help make cars safer for the rest of us. Everyone agrees that making cars safer can save lives, reduce medical costs, and in general make for a better world. The only problem is that crash test dummies, for all their sacrifices, don't really tell us how a crash will affect the human body. To learn this, we will need to start testing humans instead of crash dummies. Of course, I'm talking about virtual humans in virtual cars crashing in to virtual walls. This idea is the next step in automotive simulation. Using HPC methods it will someday be possible to determine the effects on all aspects (for instance bones, organs, muscles) and forms (for instance, adult, child, male, female) of the human body. And, the results would not stop at the virtual crash wall. A virtual HPC human would allow a new epoch of design and safety engineering.

There are several popular and freely available resource schedulers. Choices include Torque, Lava, and Maui, and in the commercial sector there are fully supported versions of Moab, Univa UD UniCluster, and Platform LSF (Load Sharing Facility).

Ready to run application software

There are plenty of "cluster aware" software packages in both commercial and open source forms. Many of the commercial packages may require specific GNU/Linux distributions, but open source packages can be built within your specific environment. Commercial packages may or may not have better performance or feature sets; however, they all have professional support options that are not always available with open source packages. Due diligence will pay off in this area.

Provisioning: Creating the cluster

Turning raw hardware into a functioning cluster is not as difficult as it was in the past. This process is called provisioning and it involves installing and configuring the head node and

These materials are the copyright of Wiley Publishing, Inc. and any dissemination, distribution, or unauthorized use is strictly prohibited.

worker nodes of the cluster. There are two main methods that are used:

- **Local:** This type of provision involves placing a full operating system image on each node. Obviously, this method requires that each node have a hard disk drive (or other persistent local storage). Care must be taken to keep the node images synchronized (which consumes valuable time and resources). The advantages to this method are that the nodes can be started independently and they don't require a central server; there can also be advantages in quicker boot times and reduced peak network traffic as a large number of nodes are brought on-line.
- **Remote:** This method configures each node across the network when it boots using standard protocols such as DHCP (Dynamic Host Configuration Protocol) and TFTP (Trivial File Transfer Protocol). These methods often are called diskless because they don't *require* a hard drive on each node; however, they don't preclude using node hard drives for local and temporary storage. The advantage of remote provisioning is that the software environment for each node is centralized and a single change can be easily propagated throughout the cluster. In addition, in cases where nodes don't have hard drives, this is the only way to bring up the nodes.

Cluster Tool Kits

Most GNU/Linux distributions lack many of the key cluster software packages. To remedy this situation, there are now cluster distributions that provide turn-key provisioning and ready-to-run software. Many of these are also freely available and based on some of the above mentioned distributions. The freely available options include

- **Rocks Clusters:** `www.rocksclusters.org`
- **Oscar** (Open Source Clusters Application Resources) `http://oscar.openclustergroup.org`

These materials are the copyright of Wiley Publishing, Inc. and any dissemination, distribution, or unauthorized use is strictly prohibited.

How to become a cluster expert in one easy step

It is very simple to become a cluster expert. The answer to every question about clusters and HPC is basically the same: "It all depends." Clusters allow so much user choice it really does depend on your specific needs. If anyone asks you a question about HPC or cluster computing, simply rub your chin and say "It all depends." From there, you're on your own. Start by understanding your needs, ask specific questions, and pay attention to what others have done in your situation (with today's infinitely connected world there's no shortage of information at your disposal).

There are commercial versions as well. These are based on GNU/Linux and often provide extra commercial applications and support as part of the cluster package:

- Scyld Clusterware
- ClusterCorp Rocks+
- Platform Cluster Manager
- Red Hat HPC Solution

These materials are the copyright of Wiley Publishing, Inc. and any dissemination, distribution, or unauthorized use is strictly prohibited.

Chapter 4

Pulling It All Together

In This Chapter

- Knowing where to start
- Finding a trustworthy source
- Following some guidelines
- Looking at benchmarks

This chapter goes over some of the practical aspects of putting together an HPC system for your company.

Following the HPC Recipe

So you want to purchase an HPC cluster. Where do you start? The following guidelines may be helpful, because the HPC procurement process is an exercise in confirming assumptions and sorting out the details.

Start with the personnel

Do you have a staff that is well versed in HPC methods? In most cases, you probably don't. It may be time to consider a consultant or some educational alternatives. The Internet can be helpful here. There is a plethora of information about HPC clusters on the Web (see the "HPC Web resources" sidebar). You can also find mailing lists and community forums where you can submit questions and discuss issues. Chances are, someone may have been in a situation similar to yours. Depending on your area of interest, you may want to ask your colleagues. At this point, you may also want to contact some hardware vendors and ISVs (independent software vendors) that offer HPC clusters. These folks can be helpful and may have pre-sales resources of their own to offer. Be clear that you aren't at the "purchase stage" just yet.

These materials are the copyright of Wiley Publishing, Inc. and any dissemination, distribution, or unauthorized use is strictly prohibited.

Consider the software

Are you planning to use commercial applications? If so, you probably want to contact the ISV if you haven't already. They will undoubtedly have previous customers that once were in the same (or a similar) position as the one you currently face. They are likely to also have some suggestions as to the type of hardware and software that best supports their product (with so many moving parts, compatibility can be important). If you're using an open source solution, it may be time to stop lurking on the mailing lists and forums. Instead of just reading the forums, ask what type of design works best for a given software application. Be prepared for opinions that may differ. Good arguments are very informative!

In addition to the application software, also consider the software infrastructure. In many cases you will be using Linux. There are a number of ways to provision a cluster (see Chapter 3) and care should be taken to choose the one that fits your needs. A few words of advice may be helpful here. It is relatively simple to set up a cluster of "Linux boxes" with some clustering software on the nodes. Success may come quickly, but in the long term you may be creating a rat's nest of dependency problems. So careful planning at the outset can be important. There are other issues that arise when deploying do-it-yourself system software across clusters. First, how well does it scale? Something that works for 8 servers may fail on 128 servers. Second, consider how you will update and maintain the software across the entire cluster. Although these are basic system administration issues, it is surprising how many administrators trip over many of these issues after the cluster is up and running.

Specify the hardware

After you have a good feel for the type of hardware, start specifying the *exact* hardware you will need. Don't be afraid to ask a hardware vendor for some evaluation units to run some benchmarks of your own. Verifying assumptions is very critical before you spend any money on hardware. Depending on the hardware vendor, they may have systems available for external large-scale benchmarking (highly recommended if possible). Next you'll want to create a concise and specific RFP (request for proposal); this is how you communicate your needs to others interested in supporting you with your HPC

These materials are the copyright of Wiley Publishing, Inc. and any dissemination, distribution, or unauthorized use is strictly prohibited.

goals. Some companies and organizations can even help you assess your business needs or assist in running benchmarks and applications. The HPC Advisory Council, for example, has a dedicated center to help with this. To find out more, visit www.hpcadvisorycouncil.com.

Who Is Going To Stand By Me?

In the early days of cluster computing, most systems were of the do-it-yourself (DIY) variety. Although this method is still popular today because of its relatively low cost, it may not be the best way to achieve your HPC goals (especially if you're just entering the realms of HPC). Another choice — one that is especially good for newcomers to HPC — is to select a vendor that will integrate and support the entire cluster.

In today's market, a turn-key cluster has many advantages and is well worth the additional cost (over the raw hardware). A vendor that has access and expertise with a wide array of hardware and software can be essential to successful deployment of an HPC system. For instance, a vendor that can offer an integrated and tested parallel file system can be preferable over one that must engineer it on the fly or buy it from another vendor. Long term support can be essential as well. Over the lifetime of your cluster you will likely have questions about various aspects of the system. These may have to do with upgrading, modifying, and expanding your cluster. A strong partner will serve you well in this situation.

Does it scale?

The term *scalability* is often used with clusters (and parallel computing). It basically means how many processors you can throw at a program before it will not go any faster. Some programs have difficulty using even two processors while other use thousands. The difference is scalability. Scalability depends on the nature of the program (the algorithm) and the underlying hardware on which it runs. The situation is very analogous to framing a house. There's a particular number of carpenters that will build the house the fastest. Beyond that number, there will be no improvement, and possibly a slow down if you add more workers. Remember Amdahl's Law (see Chapter 2 for more on this).

These materials are the copyright of Wiley Publishing, Inc. and any dissemination, distribution, or unauthorized use is strictly prohibited.

HPC Web resources

Of course you can search the Web with HPC keywords, but the following sites will help you get started with focused content and news:

- **AMD.com/hpc:** Information about AMD's HPC solutions designed to help solve today's complex scientific and engineering problems.
- **Beowulf.org:** Home of the original Beowulf project. It also hosts the Beowulf mailing list, one of the best resources for learning and asking questions about cluster HPC. This is where the rocket scientists hang out.
- **ClusterMonkey.org:** An open community-oriented Web site for HPC cluster geeks (and non-geeks) of all levels. There are tutorials, projects, and many how-to articles on the site. There is also a links section with an up-to-date set of links to essential cluster topics including books, software distributions, and more.
- **HPCAdvisoryCouncil.com:** The HPC Advisory Council's mission is to bridge the gap between high-performance computing (HPC) use and its potential; bring the beneficial capabilities of HPC to new users for better research, education, innovation and product manufacturing; bring users the expertise needed to operate HPC systems; provide application designers with the tools needed to enable parallel computing; and to strengthen the qualification and integration of HPC system products.
- **HPCWire.com:** A good site to keep abreast of news, events, and editorials from the HPC market.
- **Inside HPC:** Another good site for news about the HPC market.
- **LinuxMagazine.com:** A general Linux site with a large focus on HPC. There are plenty of articles on many aspects of HPC and Linux.
- **OpenFabrics.org:** The penFabrics Alliance (OFA) develops, tests, licenses, supports, and distributes OpenFabrics Enterprise Distribution (OFED) — open source software for high-performance networking applications that demand low latency and high scalability. OFA also provides forums for education, training, and collaborative roadmap development related to OFED and use of the software by OEMs and end-users.
- **Scalability.org:** A great blog that provides real experiences and opinions from the HPC trenches.

These materials are the copyright of Wiley Publishing, Inc. and any dissemination, distribution, or unauthorized use is strictly prohibited.

Fortran, really?

When most people ask about HPC software they're surprised to learn that most HPC codes are written in Fortran. Although many consider Fortran to be an "ancient" language, it actually enjoys quite a bit of use within the HPC community. The reason for using the second oldest computer language is largely historical.

Many of the HPC programs were originally written in Fortran and users are very reluctant to replace what works. Indeed, some HPC programs are composed of over one million lines of source code, and some mature codes have quite literally hundreds or thousands of development years invested in them!

Converting all this to another fashionable language, besides being silly, is just not economically feasible. In addition, software tools, or *compilers,* understand Fortran extremely well and are very good at optimizing source code.

C and C++ are also popular HPC languages. Like Fortran, C and C++ are "close to the wires" and can give the user maximum performance in HPC applications. (By the way, do you know that Fortran stands for *formula translation?*) The Web site `http://developer.amd.com` is an excellent resource for tools, technologies, best practices, and expert guidance to optimize your software solution.

Another, often overlooked, area is local integration. Almost all clusters are required to fit into an existing data processing infrastructure. Making sure that the vendor has the capability to assist with this process is also very important. Similar to the "last mile" issue with Internet access, local integration can require more work than anyone expected.

Following Some Do's and Don'ts for HPC Cluster Procurement

Consider some of the best practices that can help ensure a successful HPC procurement:

- **Put together a project plan.** Include a plan that covers all the aspects of the procurement including the people who will touch the cluster (end users, administrators, faculty,

These materials are the copyright of Wiley Publishing, Inc. and any dissemination, distribution, or unauthorized use is strictly prohibited.

personnel, others). Pay attention to infrastructure costs and find a home for your cluster before you buy it!

- **Kick the tires.** If at all possible, run benchmarks and try the hardware before purchasing anything. Most HPC applications can be run on a few nodes. Some vendors have test systems for running codes as well. Take full advantage of this option. There can be bottlenecks, software incompatibilities, and other unseen issues that can cause delays and add cost.
- **Require acceptance testing.** Create a set of criteria that will ensure your cluster is working properly. Be sure to include this in an RFP (request for proposal) you create. It will give you peace of mind and eliminate the low-ball hardware vendors that are looking for a quick buck.
- **Seek an HPC partner.** Using a cluster vendor that has experience in deploying and supporting an HPC cluster can be very beneficial, especially if you envision your relationship as more of a partnership than a single transaction. Anyone can sell you a rack server; few will ensure that it delivers a certain number of TFLOPS or will assist with issues and questions after the hardware has been installed. Furthermore, be wary of the low-cost hardware trap. The industry is replete with disaster stories due to someone trying to save a few dollars on memory or networking components. When in doubt, remember that a quality HPC vendor will know what works and what doesn't.

What is a Beowulf?

If you have heard the name Beowulf mentioned in relation to HPC clusters, you aren't alone. However, what is surprising to some people is that a Beowulf is not a cluster. It was the name of a project at NASA where researcher Jim Fisher accepted Tom Sterling's offer to create a personal supercomputer. Sterling and Don Becker then went about creating a commodity cluster supercomputer for under $50,000. The project was a success and opened the door to other Beowulf-style clusters at many of the government labs. It also helped establish Linux as the de facto HPC cluster operating system.

These materials are the copyright of Wiley Publishing, Inc. and any dissemination, distribution, or unauthorized use is strictly prohibited.

HPC potato chips

Potato chips are about as low tech as one can imagine. That is unless you're a leading food processor who manufactures potato chips. Because the chips are manufactured and not sliced, the company has control over the shape and aerodynamics of each chip. You may ask why the shape of a potato chip is important. Simple: Chips need to move at a high rate of speed through the production line. If they move too fast, the chips fly off the line. Understanding how the chips respond to rapid movement means the manufacturing process can be optimized. More chips in the package and fewer chips on the floor mean more profit.

- **Ask for turn-key delivery.** Although the vendor may have to assemble a cluster at your site, make sure that when they hand you the keys, it is ready to run. Hopefully, you have included acceptance testing and you will be up and running right away.

Here are few things that may hurt you in the long run:

- **Don't buy an HPC solution from a data sheet, or a "ready program."** Data sheets can give lots of information, but a data sheet has never calculated a single FLOP. Clusters are a lesson in the details. There are always a large number of assumptions that can be verified by simply installing software and running some basic tests.
- **Specifying untested hardware may not be the right course of action.** There is a tendency in HPC to want to use the leading edge. A new motherboard, processor, or interconnect may look attractive, and even show good benchmark performance. However, knowing how well new-to-market hardware will stand up to 24/7 operation is always a gamble. The larger vendors always seem to lag the smaller vendors with new hardware, but there is a good reason for this difference. The top-tier vendors will certify hardware and ensure it works as intended. There is nothing that puts a lump in your throat faster than racks of servers experiencing random failures and a vendor that shrugs off your concerns.

These materials are the copyright of Wiley Publishing, Inc. and any dissemination, distribution, or unauthorized use is strictly prohibited.

- **Don't use single core benchmarks as a basis for processor or network comparisons.** If you're going to run in a multi-core environment then why not test in a multi-core environment? Single core benchmark numbers are basically meaningless if you want to run in a multi-core world. Quite often, users are disappointed with "real world" performance after they purchase the hardware because they used inappropriate benchmarks in their specifications. This difference becomes very apparent when one considers the memory bandwidth of x86 processors.

Understanding Benchmarks

The purpose of a running a benchmark is to eliminate faulty assumptions. Choosing a processor based solely on its SPEC rating (see `www.spec.org`) or Top500 performance (see `www.top500.org`) can be akin to a game of chance. For instance, many HPC applications are sensitive to memory throughput. Choosing an architecture with adequate memory bandwidth may have a bigger influence on performance than clock speed or SPEC rating. Benchmarks are useful indicators as long as you understand what's being tested and the associated results. It's important to understand your applications and workflow are the ultimate benchmarks for any cluster procurement.

In addition, not all multi-core processor architectures are the same. Some are sensitive to process placement, which is often out of the control of the end user. For instance, the AMD Opteron™ processor has a symmetric design coupled with a high performance and balanced memory architecture. This design helps achieve better scalability between cores and across the cluster. Without proper benchmarking, one might assume this is the case for all processor designs.

These materials are the copyright of Wiley Publishing, Inc. and any dissemination, distribution, or unauthorized use is strictly prohibited.

HPC coffee

Creating a good tasting coffee may be art, but keeping it fresh is a science. A leading food packager has found that instead of metal cans, plastic containers maintain coffee freshness longer after they're opened. The simple solution was to switch to plastic coffee containers right? Well not so fast. Plastic created another problem. Coffee continues to release gasses after packaging. A metal container can easily sustain the pressure, but a metal container doesn't keep the coffee as fresh. The solution was to design a check valve that releases the pressure that builds up in the plastic coffee container. This solution, while fixing the first problem, created a second issue. When the coffee is shipped to the stores, the truck can take mountainous routes where the outside pressure becomes quite low. Gases exit the container via the check valve, but when the pressure increases the container implodes. To avoid "crushed coffee" the company applied HPC technologies to engineer a container that would, for lack of better words, "implode gracefully." Using some clever design and HPC, they were able to create canisters that can travel far and wide yet still maintain their shape and keep the coffee fresh after it is open. This may be a good time to mention that there is often lots of coffee-drinking involved in the world of HPC too — good to the last FLOP.

These materials are the copyright of Wiley Publishing, Inc. and any dissemination, distribution, or unauthorized use is strictly prohibited.

These materials are the copyright of Wiley Publishing, Inc. and any dissemination, distribution, or unauthorized use is strictly prohibited.

Chapter 5

The HPC Community

In This Chapter

- Understanding the open source model
- Using open source

Open source software is used extensively with HPC clusters. In one sense "openness" has helped to foster the growth of commodity HPC by lowering the cost of entry. It has always been possible to cobble together some hardware and use freely available cluster software to test the whole "cluster thing" for little or no cost.

Understanding Open Source

Understanding free and open software is often a hurdle for many organizations. The term "free" is often misunderstood and used in the context of free as in "free lunch" (for instance, we don't have to pay a license fee for this software). When most open source practitioners speak of "free" they are talking about free as in "free speech." The distinction is important, and in the case of HPC, a significant competitive advantage.

Free as in "free speech" software means the user has the right to understand and modify the software as they see fit. In HPC this provides three key advantages. First, software can be customized in any number of ways allowing support for a multitude of file systems, interconnects, peripherals, and other "small market" projects that would otherwise never garner commercial interest. Second, it provides a safeguard against unplanned obsolescence. In the past, it was not uncommon for a large computer system to lose software support because a vendor went out of business, the system had been phased out, or the budget may have run out. With open source, users have a choice to continue using these systems and take over the support themselves. Finally, open

These materials are the copyright of Wiley Publishing, Inc. and any dissemination, distribution, or unauthorized use is strictly prohibited.

source allows a high level of optimization. Software installations can be minimized so that only essential HPC software is present. For instance, there is no need for a graphical user interface or sound card drivers on cluster nodes. Customers like choice and vendors like sharing the high cost of software development.

Openness also fosters community and collaboration. In the case of HPC, there is a large ecosystem of users, vendors, and developers from many companies and organizations that freely exchange ideas and software without the need for legal agreements. This community also provides a huge and open knowledge base of ideas, best practices, and experiences that help everyone involved. In HPC, "open" just works better.

Looking at How HPC Works with Open Source

There are many types of open source licenses. For instance, the GNU license from the Free Software Foundation has some requirements on how you must include the source code if you distribute a program in binary form. Others may only require attribution and recognition of copyright when binary codes are distributed. In any case, the goal of open software is shared development and use. In one sense, some projects, like creating and maintaining an operating system, are such big jobs that it makes sense to share the development effort across company and hardware lines. The same can be said for the HPC market.

Although there are many arguments for and against open software, the HPC market represents an area where both open and closed source solutions co-exist and work together. It is not uncommon for an entire cluster to be composed of open source software except for some commercial applications. For users that compile their own codes, a commercial compiler, debugger, and profiler are almost always present. Many clusters also employ commercial schedulers that allocate resources on the cluster (or groups of clusters) for an organization. Finally, though software may be open and freely distributed there are a certain number of companies that provide commercial support for cluster software. Overall, the combined open and closed aspects of the HPC software ecosystem seem to work rather well.

These materials are the copyright of Wiley Publishing, Inc. and any dissemination, distribution, or unauthorized use is strictly prohibited.

Chapter 6

Ten (Okay, Six) HPC Takeaways

In This Chapter

- Understanding HPC's advantages
- Developing a plan

Concluding this very brief introduction to the world of HPC computing is rather difficult. Many topics were given only cursory coverage, while others were totally omitted. It is possible however, to offer the following six HPC takeaways:

- **HPC can be a source of new competitive advantage:** HPC represents a new competitive advantage for many companies and industry sectors. Many organizations, maybe even your competitors, are using HPC as a way to help reduce cost, design new products and processes, solve problems, and increase profitability. This trend is not pie-in-the-sky hyperbole; it is good old fashioned R&D with some powerful new tools.
- **Choices invite optimum design:** Due to the growth of commodity hardware, HPC has moved from a pricey boutique market to a cost-effective technology that has moved into virtually every market sector. Many of the design choices that were once made by a supercomputer vendor are now made by the end-users themselves. In essence, customer choice allows HPC solutions to be engineered for specific problem sets and requirements, offering potentially tremendous value and ROI for the end-user.
- **Try before you buy:** Clustered HPC systems can be built from commodity off-the-shelf hardware and software. The cost to "try out" various configurations can be low,

These materials are the copyright of Wiley Publishing, Inc. and any dissemination, distribution, or unauthorized use is strictly prohibited.

but the payoff of making the right HPC decisions can be immeasurably high. Benchmarking your application(s) on actual HPC clusters is the only real test of true performance. Don't give too much weight to glossy data sheets and industry-wide benchmarks or "ready programs." To learn how you can accelerate your computational engineering programs with your own HPC design and simulation system, explore resources such as The HPC Advisory Council. This resource allows you to see solutions in action and learn more about the unique models and methods developed there. See `www.hpcadvisorycouncil.com` for available cluster resources and industry best practices material on many commercial and open source codes.

- **Develop a plan:** Purchasing HPC resources isn't as hard as it seems, if you follow some basic guidelines and ask the right questions. There are both hardware and software vendors that understand your needs and will work with you as a partner and not a parts supplier.
- **Join the community:** Open source software and the Linux operating systems have been big enablers for HPC clusters. There is tremendous value in using community-derived software and expertise to help meet your project goals. You may be pleasantly surprised to learn that many top tier HPC companies are already active participants in the community.
- **Have fun:** One final bit of advice. Have fun. HPC is, if nothing else, a great way to push the envelope and try new and exciting ideas. The cost to play in the HPC world is at an all-time low and there are plenty of resources to help you have a good time.

These materials are the copyright of Wiley Publishing, Inc. and any dissemination, distribution, or unauthorized use is strictly prohibited.