

# Linux and High-Performance Computing

David A. Bader  
New Jersey Institute of Technology  
Newark, NJ 07102, USA

## Abstract

In the 1980s, high-performance computing (HPC) became another tool for research in the open (non-defense) science and engineering research communities. However, HPC came with a high price tag; the first Cray-2 machines, released in 1985, cost between \$12 million and \$17 million, according to the Computer History Museum, and were largely available only at government research labs or through national supercomputing centers. In the 1990s, with demand for HPC increasing due to vast datasets, more complex modeling, and the growing computational needs of scientific applications, researchers began experimenting with building HPC machines from clusters of servers running the Linux operating system. By the late 1990s, two approaches to Linux-based parallel computing had emerged: the personal computer cluster methodology that became known as Beowulf and the Roadrunner architecture aimed at a more cost-effective supercomputer. While Beowulf attracted attention because of its low cost and thereby greater accessibility, Roadrunner took a different approach. While still affordable compared to vector processors and other commercially available supercomputers, Roadrunner integrated its commodity components with specialized networking technology. Furthermore, these systems initially served different purposes. While Beowulf focused on providing affordable parallel workstations for individual researchers at NASA, Roadrunner set out to provide a multi-user system that could compete with the commercial supercomputers that dominated the market at the time. This paper analyzes the technical decisions, performance implications, and long-term influence of both approaches. Through this analysis, we can start to judge the impact of both Roadrunner and Beowulf on the development of Linux-based supercomputers.



## 1 INTRODUCTION

The Beowulf and Roadrunner projects represent two distinct philosophies for translating these theoretical advances into working systems, and their divergent approaches illuminate enduring tensions in parallel system design between accessibility and performance, standardization and specialization, that remain relevant to contemporary challenges in exascale computing and beyond.

This historical analysis contributes to the anniversary celebration by documenting a pivotal moment when academic innovation and open-source collaboration fundamentally altered an industry previously controlled by a handful of vendors. By 2017, Linux-based systems achieved complete dominance of the TOP500 supercomputer list—a remarkable outcome that vindicated the commodity cluster approach but obscured the distinct contributions of its early pioneers. Understanding which architectural decisions proved prescient and which represented evolutionary dead ends provides lessons for today’s researchers navigating similar inflection points in heterogeneous computing, AI accelerators, and quantum-classical hybrid systems. The comparison presented here demonstrates that effective system design requires balancing cost accessibility with performance requirements—a lesson as pertinent to the next 35 years of parallel and distributed computing as it was to the transformative era this paper examines.

Computer systems can tackle larger problems by distributing work across multiple machines connected through a network. When these systems are built using standard, off-the-shelf hardware components rather than specialized equipment, this approach is called commodity cluster computing. This cost-effective method leverages inexpensive, commercially available servers and networking gear to create powerful distributed computing systems that can rival the performance of expensive supercomputers. This work builds upon and significantly expands the historical account presented in the author’s IEEE Annals of the History of Computing manuscript “Linux and Supercomputing: How My Passion for Building COTS Systems Led to an HPC Revolution” [8]. COTS stands for Commercial Off-the-Shelf, referring to ready-made products or software that can be purchased and used immediately, rather than being custom-developed for specific requirements. While that anecdotal piece provided a first-person narrative of developing the Roadrunner Linux supercomputer, this present article conducts a comprehensive comparative analysis of the two foundational approaches to commodity cluster computing: Beowulf and Roadrunner, the latter the work of the present author. This expanded study provides detailed technical analysis of both architectures, systematic comparison through structured tables, expert testimonials from scientific users, and thorough examination of the divergent design philosophies that shaped these seminal systems. Additionally, this paper incorporates substantial new historical research into the Beowulf project’s development, technical limitations, and positioning within the broader HPC ecosystem—elements that were not addressed in the prior personal account. The result is an assessment of how these two distinct approaches influenced the evolution of Linux-based supercomputing and established the architectural foundations for modern high-performance computing.

## From Big and Bulky to COTS and Open Source

Vector machines, a type of computer architecture designed for high-speed processing of numerical calculations using Single Instruction, Multiple Data (SIMD) vector processors, had dominated supercomputing since the introduction of the Cray-1 in 1976 [74]. The Cray-1 was the first commercially successful supercomputer using vector instructions, and it made the technology mainstream in the supercomputing world. Other systems were developed in the 1980s, including massively parallel multiprocessor systems such as Thinking Machines' CM-5 Connection Machine, launched at the Massachusetts Institute of Technology by W. Daniel Hillis and Lewis W. Tucker [31]. Cray introduced a new system called X-MP in 1982, featuring the company's first shared-memory parallel vector processor. An update to that system was the Cray Y-MP, which featured more memory and faster performance [23]. The Cray-2 debuted in 1985. They soon replaced the MP systems as the world's fastest machines and were known for their distinctive design featuring total immersion cooling, using a special liquid to cool the densely packed circuit boards. These systems were big: The Cray-1 occupied 2.7m × 2m of floor space and contained 60 miles of wires [74]. They were expensive: in 1976 that same Cray-1 sold for as much as \$10 million. And, unless you worked in national defense, were part of a research team at a large government or academic lab, or with a major industrial user, these machines were out of reach. They ran on proprietary hardware and software, and nothing was compatible with other systems. Moreover, the software itself became harder to create over time. "The architectures of these systems pose major software problems that the computing industry is ill-equipped to handle, especially for special-purpose systems with limited markets," stated the 1982 foundational "Lax Report" [39] that led directly to NSF establishing supercomputer centers in 1985 and represented early government recognition of the growing software challenges in supercomputing. Eugene Brooks of Lawrence Livermore National Laboratory predicted that vector processors would not be able to keep up with the new generation of microprocessors using scalar instructions as far back as 1990, when he delivered a talk at Supercomputing 1990 called "Attack of the Killer Micros" [18]. Clearly, improvements in the field were needed, and they came in the form of clustered high-end servers running Linux, a formula for supercomputers that still dominates supercomputing today.

The ASCI Red supercomputer [46], deployed at Sandia National Laboratories in 1996–1997, represented the pinnacle of traditional vendor-built massively parallel processing at the time, achieving 1.34 TFLOPS with 9,216 Intel Pentium Pro processors organized in a 38 × 32 × 2 mesh topology. While ASCI Red shared with both Beowulf and Roadrunner the use of commodity commercial off-the-shelf (CCOTS) microprocessors, the similarities largely ended there. ASCI Red relied on proprietary operating systems—the Intel Paragon OS (a distributed UNIX) for service and I/O nodes, and Cougar, a lightweight kernel derived from Sandia's Puma, for compute nodes—rather than Linux. Its custom Interconnection Facility (ICF), built with proprietary VLSI components (the Mesh Routing Component and Network Interface Component), delivered 800 MB/sec bidirectional bandwidth between nodes, far exceeding what commodity Ethernet could provide. However, unlike the early Linux-based approaches, ASCI Red required a massive budget (approximately \$46 million) and a large engineering team from Intel, demonstrating that while commodity processors had become viable for supercomputing, the full system integration—operating system, interconnect, and software stack—remained within the domain of traditional supercomputer vendors. The Linux-based systems discussed in this paper demonstrated that comparable architectural decisions could be implemented at far lower cost using open-source software and, in Roadrunner's case, commercially available high-performance networking.

My journey toward COTS supercomputing began in junior high school in 1981, when I discovered an article about a parallel computing system designed for image processing and pattern recognition [61]. The concept immediately captivated me—I knew I had to build my own parallel computer. Eight years later, as an undergraduate student at Lehigh University in 1989, serendipity provided the perfect opportunity. While exploring the university's resources, I stumbled upon several donated Commodore Amiga 1000 personal computers gathering dust in a forgotten closet. These machines, combined with my growing knowledge of parallel processing, gave me all the ingredients I needed to construct my first parallel computer and explore new applications that required more computational power. The Amiga cluster nodes ran Commodore's AmigaOS 1.0 operating system, and programming was done through node programs that could send messages over a network to each other. Tools such as job launchers were not provided and needed to be developed.

A year later in 1990, I designed parallel divide and conquer algorithms for combinatorial problems such as sorting and searching on a 128-processor nCUBE hypercube parallel computer donated by AT&T Bell Laboratories. Programming the nCUBE was very different and required the user to write separate host and node programs. The nCUBE host processor ran the AXIS operating system similar with UNIX, "but was lacking many of the most useful utility programs that UNIX users are accustomed to having" [76]. These early experiences taught me that the development of powerful parallel machines required a simultaneous development of scalable, high-performance algorithms and services. Otherwise, application developers would be forced to develop algorithms and basic tools from scratch every time vendors introduced newer, faster hardware platforms.

At the University of California, Berkeley, a pioneering distributed computing research project was launched in the mid-1990s called Network of Workstations (NOW), led by David Culler [3]. Concurrently, Miron Livny, head of the University of Wisconsin's Condor project, aimed to make a building full of desktop computers act as a single large computer [44]. These projects leveraged high-bandwidth, switch-based local area networks with a custom low latency network interface and global layer operating system. NASA began the Beowulf project in 1994, following the "Pile-of-PCs" methodology to build a parallel workstation from a cluster of personal computers (PCs). In their 1995 paper, "Beowulf: A Parallel Workstation

for Scientific Computing” [71], Beowulf developers Thomas Sterling *et al.* describe “the Beowulf parallel workstation [as] a single user multiple computer with direct access keyboard and monitor.” The term “multiple computer” refers to combining multiple PCs into a unified system to increase processing power and computational capabilities.

For over a decade, the computing world accepted a compelling origin story: Beowulf clusters were named after the epic hero who challenged formidable opponents, symbolizing how commodity hardware could take on expensive supercomputers. This narrative, told consistently by creators Thomas Sterling and Donald Becker from 1994 to 2004, fit perfectly with their David-versus-Goliath approach to parallel computing. But in NASA’s 2020 Spinoff publication, Sterling finally revealed the truth. The naming was completely accidental and occurred under pressure. When a NASA Goddard Space Flight Center (GSFC) program administrator called demanding an immediate project name in 1994, Sterling was “helplessly looking around for any inspiration” in his office. His mother had majored in Old English, leaving him with a copy of the Beowulf epic. In desperation, he told the administrator: “Oh hell, just call it Beowulf. Nobody will ever hear of it anyway” [49]. Sterling admits the heroic justification was “invented in hindsight” for public relations purposes. The fabricated narrative served the project well, providing a memorable and meaningful explanation that resonated with the cluster computing community’s understanding of their battle against established supercomputer manufacturers.

Beowulf clusters targeted individual researchers who needed affordable parallel computing workstations. Beowulf, according to NASA’s James R. Fischer, was created to provide NASA staff with gigaflops workstations that would allow them to use and share software over various platforms [26]. The frustration of dealing with difficult-to-use and often incompatible hardware and software motivated the project, as well as the need for better price-to-performance ratios. Its scope was limited to building a prototype scalable workstation for NASA’s scientists that could run the same Earth and Space Sciences software (but not as efficiently) as supercomputers of the day. However, the Beowulf project did show that commodity hardware and open-source operating systems and software had a role to play in the world of supercomputing. C. Gordon Bell, National Academy of Engineering (NAE) member, Microsoft Emeritus Researcher and founding Assistant Director of the National Science Foundation’s Computing and Information Science and Engineering Directorate, and colleague Jim Gray, then manager of Microsoft Research’s eScience Group, described the Beowulf technology in the early 2000s as “Beowulf enabled do-it-yourself cluster computing using commodity microprocessors” [11]. By that time, Beowulf systems offered a single platform standard that allowed applications to be written and to run on more than one computer. Bell and Gray acknowledged that the Beowulf framework, while cost effective, could not meet all supercomputing needs and “perform[ed] poorly on applications that require large shared memory” [11]. They encouraged research into next-generation “Beowulfs” that would stimulate cluster understanding and training so they can serve the needs of research centers that depend on high-end supercomputing.

While the Beowulf project zeroed in on price/performance, a different focus motivated me, then a professor of electrical and computer engineering at the University of New Mexico, as I took the concept of using commodity clusters for speed and compatibility at a lower price point but prioritizing performance above all. My alternative Linux-based supercomputing architecture combined Linux, COTS components, and (newly) high-speed, low-latency interconnection networks. Bell, looking back on these efforts, commented that “Bader was first to design a Linux supercomputer with the speed, performance and services of a large, centralized and general-purpose supercomputer” [10]. In April 1998, the Roadrunner Phase 1 system used the new Myrinet System Area Network (Myrinet/SAN) [16], which was about 256 times the total bandwidth available using standard Ethernet on which Beowulf’s networks ran. I named the system Roadrunner after the New Mexican state bird, which is the fastest running bird capable of flight. This name combined “speed” and “New Mexico” together with mental images of the cartoon Roadrunner outsmarting Wiley E. Coyote. Roadrunner also included compilers, a job scheduler, and features to enable parallel programming, such as software-based distributed shared memory and Message Passing Interface (MPI). The Roadrunner Phase 2 system became the first Linux supercomputer available for open use by the national science and engineering community through the National Science Foundation’s National Technology Grid, entering production in April 1999 [27]. Within a decade, this architectural approach became the predominant model for COTS supercomputing, which in turn became the dominant model for all supercomputers worldwide.

Bell, reflecting much later on the development of Roadrunner, highlighted the importance of performance focus, noting the “Bader Roadrunner design could efficiently run the national science community’s most demanding supercomputing applications at a fraction of the cost of traditional supercomputers—unlike Beowulf clusters that were used by individuals and were not competitive in performance” [10]. The two approaches served different purposes and made different contributions to the evolution of supercomputing. This paper highlights these design approaches, with Beowulf focused on mass-market components and low price points and Roadrunner focused on higher performance at the cost of some specialized components.

## Comparative Analysis: Beowulf Vs. Roadrunner Architectures

### 2 THE BEOWULF VISION OF PERSONAL PARALLEL COMPUTING

From their inception, Beowulf systems were conceived as enhanced parallel workstations rather than traditional supercomputers. Sterling and his collaborators were explicit about this positioning. In his 1996 *Communications of the ACM* article, Sterling characterized Beowulf as “an experimental distributed PC system developed to evaluate this new PopC opportunity in single-user environment computing,” emphasizing that “the PopC approach as demonstrated by Beowulf

TABLE 1  
Comparison of Beowulf versus Roadrunner.

Feature	Beowulf	Roadrunner
Development Target	Low-cost replacement for single-user workstation	Low-cost replacement for traditional supercomputer
Design Philosophy	“Mass-market commodity off-the-shelf” (M <sup>2</sup> COTS) with strict vendor-neutral requirements	Balanced integration of commodity components with specialized high-performance technologies
Development Timeline	NASA project begun in 1994, first system operational in 1995	Prototype developed in 1998, production operation April 1999
User Model	Single-user, single-application workstation	Multi-user, multi-application shared resource
Network Technology	Ethernet	Specialized three-network architecture including 1) high-speed, low-latency interconnects for data network, 2) RS-232 serial network for diagnostics, and 3) an Ethernet control network
Node Architecture	Uniprocessor nodes	Multiprocessor nodes
System Management	None	Tools for job scheduling and resource allocation
Target Applications	Earth and Space Science satellite image processing	Scientific applications from astrophysics and cosmology, climate and weather research, physics research, engineering and applied science, materials science, computer science and systems research, graph analytics, and national security
Positioning	Explicitly designed as complementary to—not a replacement for—commercial HPC systems	Direct alternative to commercial supercomputers
Integration Model	“Just-in-place” DIY integration by end-users	Vendor integration
Scalability	10’s of processors	1000’s of processors
Historical Influence	Popularized commodity cluster concept	Established architectural template for modern supercomputing
Development Team	Five engineers (Thomas Sterling, Don Becker, John Dorband, Don Jacob, and Jim Fischer) with NASA institutional support	Single individual (David A. Bader)

does not so much challenge the workstation market as reveal a possible direction for its advance and advantage” [65]. The 1995 *HPDC* paper made the design target even more explicit: “While most distributed computing systems provide general purpose multi-user environments, the Beowulf distributed computing system is specifically designed for single user workloads typical of high end scientific workstation environments” [72]. That paper further noted that “the very nature of Beowulf as a single-user workstation demanded that the marginal replacement cost of the system be consistent with pricing of contemporary high end workstations”—approximately \$40,000 for the prototype [72]. These parallel workstations would give researchers a development environment running the same software stack as production supercomputers, enabling code development and testing at smaller scales before scaling up to larger institutional systems. The Beowulf team acknowledged inherent scalability limitations consistent with this role: “It is true that Beowulf is limited in the scope of its scaling. Its configuration is only intended to exploit the first order of magnitude in parallelism” [72]. The 1997 *IEEE Aerospace Conference* paper reinforced this positioning, stating that the Pile-of-PC methodology “is not for everyone. Rather, it is an emerging opportunity in the high performance computing field and complements rather than competes with the HPC industry commercial products” [59]. The fundamental philosophy emphasized democratizing parallel computing access rather than directly challenging established supercomputing centers. Roadrunner, by contrast, was designed from inception as a general-purpose, multi-user supercomputer serving a national user community, directly competing with commercial supercomputer offerings rather than complementing them.

Beowulf, short for “Beowulf Parallel Workstation,” (see Figure 1) was envisioned to augment a single user with more compute power using mass market PCs and Ethernet, its creators reported at the *IEEE Aerospace Conference* in 1997 [59]. Thomas Sterling, one of Beowulf’s co-creators, put it this way: “The Beowulf parallel workstation defines a new operating point in price-performance for single-user computing systems” [71]. James Fischer, the project manager for NASA’s High-Performance Computing and Communications Earth and Space Science Project at GSFC during the development of Beowulf recollected, “Looking back to the origins of the Beowulf cluster computing movement in 1993, it is well known that the driving force was NASA’s stated need for a gigaflops workstation costing less than \$50,000” [26].

The 1998 NASA workshop assessment confirmed this workstation-centric origin, reporting that Beowulf’s charter was “a single user ‘gigaflops’ science workstation” and that the primary design consideration was not floating point performance but rather disk access: “An evaluation of the requirements for a scientific station for NASA showed that disk access capacity and bandwidth was far more important to user response time than was floating point performance” [68]. The project’s cost target was explicitly set at “the price of a high-end scientific workstation, assumed to be \$50,000,” with the goal of assembling a cluster of low-cost PCs with an order of magnitude larger disk capacity and approximately eight times the disk bandwidth of a single workstation [68].

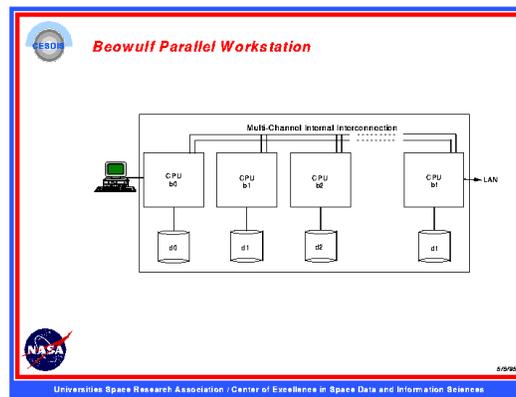


Fig. 1. NASA Slide of Beowulf Parallel Workstation. Image source: [67].



Fig. 2. Wiglaf, the original Beowulf prototype. Image source: NASA Goddard Space Flight Center.

The original Beowulf prototype, named "Wiglaf," (see Figure 2) was built by Thomas Sterling, Don Becker, John Dorband, and Dan Jacob, at NASA GSFC in late 1994. Wiglaf contained 16 uniprocessor nodes (each with a 66 MHz Intel 80486 and SiS 82471 chipset motherboard), connected by commodity 10BaseT Ethernet. The developers chose the Slackware Linux distribution to run on each node. At the end of 1994, the developers updated Wiglaf's processors, replacing them with 100 MHz Intel 80486 DX4 processors.

### Strict Commitment to Commodity Hardware

The Beowulf design philosophy, in its effort to keep costs down, enforced adherence to mass-market commodity components, rejecting any technology that was not widely available through multiple vendors. This approach was codified as "M<sup>2</sup>COTS" (mass-market commodity off-the-shelf) [36]. In 1997, Michael Warren *et al.* published a parallel computing conference paper where they define "Beowulf-class systems [as] off the shelf M<sup>2</sup>COTS PCs [that] are interconnected by low cost local area network (LAN) technology running an open source, Unix-like operating system and executing parallel applications programmed with an industry standard message passing model and library" [79]. Sterling emphasizes this point in his 1999 How to Build a Beowulf book: "Beowulfs use only mass-market components and are not subject to the delays and costs associated with custom parts and custom integrations" [70]. By restricting itself to the most commoditized

hardware, Beowulf systems were true to the M<sup>2</sup>COTS ideal, at the expense of performance potential. Sterling formalized this boundary in the 1998 NASA workshop assessment, stating that “Beowulf-class systems exclusively employ COTS technology, usually targeted to the mass market where cost benefits of mass production and distribution drive prices down” [68]. The assessment acknowledged one exception: networking technology, “which while COTS, is not mass market due to the relatively small volume.” However, even for this exception, Sterling imposed a cost ceiling: networking components were included only if vendors could “provide their product within that cost framework” [68]. This formulation reveals a critical distinction: a technology could be commercially available—as Myrinet was—and still fall outside the Beowulf class if it exceeded the cost bounds that defined the M<sup>2</sup>COTS paradigm. This stance extended to all aspects of system design, including processors, networking equipment, and software components. The use of vendor-neutral components meant Beowulf clusters could not benefit from the performance of specialized hardware, but enjoyed flexibility in system construction. As Sterling explained, “Basically, you can order most of Beowulf’s components from the back pages of Computer Shopper or get them for free over the ‘Net’” [49].

A *Pile-of-PCs* is the term used today to describe the loose ensemble or cluster of PCs applied in concert to a single problem. Similar in principle to the Berkeley NOW project, it emphasizes mass market commodity components, dedicated processors (rather than scavenging cycles from idle workstations), and a private system area network (SAN), all with the goal of achieving the best system cost/performance ratio. Beowulf added to this the following principles: no custom components, easy replication from multiple vendors, a freely available software base, using freely available distribution computing tools with minimal changes, and returning the design and improvements to the community. The approach exploits components that respond to widely accepted industry standards and benefits from lower prices resulting from heavy competition and mass production. Subsystems provide accepted, standard interfaces such as PCI bus, IDE and SCSI interfaces, and Ethernet communications. This is the Beowulf approach, and one advantage is that no single vendor owns the rights to the product. In a Computer History Museum talk in 2000, Sterling remarks that “Beowulfs formed a do-it-yourself cluster computing community using commodity microprocessors, local area network Ethernet switches, Linux (and now Windows 2000), and tools that have evolved from the user community. This vendor-neutral platform used the MPI message-based programming model that scales with additional processors, disks, and networking” [63]. As Steve Elbert noted at the 1<sup>st</sup> Pentium Pro Cluster Workshop in 1997: “The difference between Beowulf and other parallel processing systems is that it has no custom hardware or software but consists of standard, off-the-shelf computers, . . . connected by Ethernet and functioning as a single machine on the Linux operating system” [25].

### Beowulf Network Choices

The Beowulf architecture was defined in ways that excluded high-performance interconnects. Sterling’s 1999 MIT Press book states that “A Beowulf-class system is a cluster of mass-market commodity off-the-shelf (M<sup>2</sup>COTS) PCs interconnected by low cost local area network (LAN) technology running an open source code UNIX-like operating system and executing parallel applications programmed with an industry standard message passing model and library” [70], and that systems using specialized HPC networks would not be a Beowulf. The Los Alamos National Laboratory’s (LANL) Avalon Project FAQ explained that Myrinet was not used because it would “double the cost” of the system and was not considered a mass-market component [45]. The 1998 NASA workshop assessment quantified why Myrinet fell outside Beowulf’s design envelope. Sterling and his co-authors established a cost guideline for Beowulf networking: “A rule of thumb is that the network should cost approximately one quarter of the total system cost. Given that a current node cost is between \$1,600 and \$1,800 per node without communication, the cost for communication per node should be between \$530 and \$600 per node” [68]. By contrast, the assessment reported that Myrinet NICs alone cost approximately \$1,400 per port—more than double the entire acceptable per-node networking budget—leading Sterling to conclude that “a system employing Myrinet can dedicate half of the cost to the communication network; probably not the balance one would choose” [68]. Fast Ethernet, at roughly \$225 per node for small systems, comfortably fit within these constraints. The cost differential was not incidental; it was the architectural boundary that defined the Beowulf class.

This narrow definition of “commodity”—limited to components sold in retail computer stores for desktop systems—had significant architectural consequences. Any network technology designed specifically for HPC, regardless of its commercial availability, fell outside Beowulf’s self-imposed boundaries. Roadrunner adopted a broader definition: any component purchasable from a commercial vendor without custom engineering qualified as commodity. Myrinet, though manufactured for the HPC market, was a standard catalog item available to any buyer. This definitional difference was not merely semantic—it determined whether Linux-based clusters could achieve competitive supercomputer performance or would remain limited to embarrassingly parallel workloads.

Sterling’s own assessment confirms that commodity processors alone did not qualify a system as Beowulf-class. In the 1998 NASA workshop proceedings, Sterling noted that ASCI Red was “not a Beowulf-class system” despite the fact that it “incorporates the identical chip level technology as that of Beowulf with the exception of the NIC (network interface controller)” [68]. This exclusionary principle—that a system sharing Beowulf’s commodity processors but employing a different networking architecture fell outside the Beowulf class—applied with equal force to Roadrunner. Roadrunner used commodity Intel processors but integrated Myrinet as its data network, a technology that the same 1998 assessment explicitly treated as incompatible with Beowulf’s cost constraints.

However, by limiting systems to commodity Ethernet, Beowulf clusters suffered performance bottlenecks in communication-intensive applications. These bottlenecks restricted the types of computational problems that could be efficiently solved,

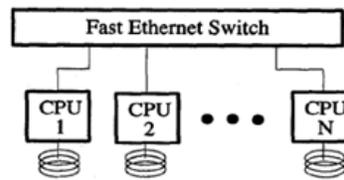


Figure 2: Caltech 'Hyglac' cluster

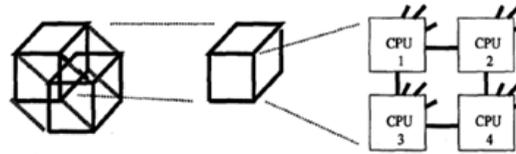


Figure 3: Los Alamos 'Loki' cluster

Fig. 3. Beowulf network topology examples.

with limitations most apparent when running tightly-coupled parallel applications requiring frequent inter-node communication. This was particularly problematic since such workloads represented precisely the applications that drove most supercomputing research and development of the era. But for the Beowulf team, cost was the most important consideration as “the driving force was NASA’s stated need for a gigaflops workstation costing less than \$50,000” [26].

In 1994 Sterling recruited Don Becker to NASA for his skill at writing operating system software [49]. Because Beowulf’s “processors were too fast for a single Ethernet and Ethernet switches were too expensive” [64]. “To balance the system Don Becker rewrote his Ethernet drivers for Linux and built a ‘channel bonded’ Ethernet where the network traffic was striped across two or more Ethernets” [64]. Channel bonding represented an innovative but ultimately temporary solution. In 1995, Fast Ethernet was introduced. At 100 Mbps, Fast Ethernet could transfer data at 10× the rate of standard Ethernet. At a 1995 conference, Beowulf developers acknowledged that one of the limiting factors for scaling up Beowulf clusters was the network, and they believed Fast Ethernet was the solution [67]. Beowulf developer Michael Warren stated: “Nothing can beat Fast Ethernet for price/performance at the moment” [78]; and the community rejected Myrinet as “not commodity” [25]. By 1998, industry documentation acknowledged the channel bonding technique was already being supplanted: “As 100Mbit/s Ethernet and 100 Mbit/s Ethernet switches have become cost effective, the need for channel bonding has gone away (at least for now)” [57].

According to Sterling, the Beowulf project provided improvements over prior personal computer clusters that included: “Ethernet drivers, channel bonding, advanced topologies, applications, [and] ensemble management tools” [64]. The Beowulf developers focused their research efforts on mitigating the relatively long latencies and modest interconnection bandwidth provided by low-cost networking such as Fast Ethernet. “This is being addressed by software performance tuning, aggregating networks, and rich interconnect topologies,” the team reported [65].

### Beowulf Network Topology

In 1997 Beowulf developers then attempted to overcome Ethernet’s limitations by using complex network topologies. These efforts included elaborate multi-network designs, as well as hyperlinked topologies to reduce communication latency through strategic node connections. The most notable example was the “hypercube plus switch” topology used in LANL’s “Loki” cluster [78].

While these complex topologies provided performance improvements for specific communication patterns, they added significant system management. As Fast Ethernet switches became more affordable, this created a new choice for Beowulf designers: continue using complex topologies with software-based routing, or adopt simpler network designs that relied on dedicated hardware switches. As the team reported in 1996: “The original networking strategy for Beowulf was to use multiple Ethernet networks in parallel, each connecting all the nodes within the system. Both 10 Mbps and the new 100 Mbps Fast Ethernet were employed in separate Beowulf systems. The parallel networks were managed through a technique called channel bonding that uniformly distributed packets among the interconnects in a manner transparent to the user code. . . . It is shown that in many circumstances the more complex topologies perform better, and in some circumstances software routing techniques compare favorably to more expensive hardware switch mechanisms” [65]. This research demonstrated that software-based routing in complex topologies could still compete with hardware switching solutions.

By this time, major research institutions were deploying 16-node Pentium Pro Beowulf clusters tailored to their specific computational needs. Notable examples included Caltech’s “Hyglac” cluster and LANL’s “Loki” machine, identical systems

except for their topologies, both optimized for N-body galactic gravitational simulations, as well as a system at NASA GSFC. These installations demonstrated different networking approaches—from Caltech’s Fast Ethernet crossbar switch configuration to LANL’s hybrid hypercube-plus-switched network design that improved performance by bypassing longer hypercube routes (Figure 3) [78].

However, investigations of topologies for Ethernet networks declined when commodity Gigabit Ethernet switches became available in 1999. By 2000, tutorials on building Beowulf clusters advocated for buying these switches: “Switches have become inexpensive enough that there’s not much reason to build your network by using cheap hubs or by connecting the nodes directly in a hypercube network” [43].

### Beowulf Uniprocessor Node Architecture

Beowulf developers initially avoided symmetric multiprocessing (SMP) nodes in favor of uniprocessor architectures. At the time Roadrunner was being developed, the prevailing Beowulf guidance discouraged multiprocessor configurations. At cluster workshops in 1997, Beowulf advocates argued that “SMP is a terrible idea” for three main reasons: research computing centers were already short on memory bandwidth and network bandwidth, and multiple CPUs on a single node were seen as making the problems worse; message passing programs should not require users to also deal with shared memory complexity; and multiple processors sharing memory was seen as making it harder to write a stable and efficient operating system [78]. At another 1997 workshop, Beowulf developers stated their view that “using multiple processors within each node (SMP) is unlikely to be a good idea for many applications” [66].

The 1998 NASA workshop assessment reinforced this position on technical grounds, reporting that while motherboards with up to four processors per SMP node were available, “the memory bandwidth is inadequate to support good utilization of these processors except in particularly carefully crafted codes that make particularly favorable usage of the two layers of caches on each processor” [68]. The assessment further noted that Linux’s SMP support at the time was limited: “the current implementation does not permit more than one O/S service call to be performed simultaneously so that operating system work does not speed up with additional processors within an SMP node” [68]. These constraints reflected genuine technical limitations of the era—limitations that Roadrunner’s custom kernel work specifically addressed.

However, the Beowulf community’s thinking on this issue evolved as multiprocessor systems became more commodity and cost-effective. By 2001, Sterling’s MIT Press volume *Beowulf Cluster Computing with Linux* acknowledged that dual-processor SMP nodes might represent the “price/performance sweet spot” for cluster configurations [69]. This evolution reflects the Beowulf project’s pragmatic response to changing hardware economics rather than rigid adherence to original design principles.

Roadrunner’s 1998 adoption of dual-processor nodes thus anticipated a direction that the broader commodity cluster community would eventually embrace. At the time of Roadrunner’s development, however, this architectural choice represented a deliberate departure from contemporaneous Beowulf guidance, reflecting my emphasis on computational density and hierarchical parallelism over strict adherence to the uniprocessor model then prevalent in the Pile-of-PCs community.

### Beowulf’s Contribution to Cluster Approaches

The Beowulf project’s technical innovations focused on administrative improvements. These differences included dedicated (rather than timeshared) compute nodes, isolated network infrastructure, and operating system optimizations. The software contributions focused primarily on system management tools, Ethernet driver modifications, and the channel bonding techniques described previously. The Beowulf team described their differentiating factors thus in 1997:

“A Beowulf-class cluster computer is distinguished from a Network of Workstations by several subtle but significant characteristics. First, the nodes in the cluster are dedicated to the cluster. This helps ease the load balancing problem, because the performance of individual nodes are not subject to external factors. Also, since the interconnection network is isolated from the external network, the network load is determined only by the application being run on the cluster. This eases the problems associated with unpredictable latency in NOWs. All the nodes in the cluster are within the administrative jurisdiction of the cluster. For example, the Beowulf software provide a global process ID which enables a mechanism for a process to send signals to a process on another node of the system. This is not allowed on a NOW. Finally, operating system parameters can be tuned to improve performance. For example, a workstation should be tuned to provide the interactive feel (instantaneous responses, short buffers, etc), but in a cluster the node can be tuned to provide better throughput for coarser grain jobs because they are not interacting with users” [58].

### Scope and Positioning within HPC

Beowulf developers were clear about their design goals and target market, positioning their system as a complement to existing high-performance computing solutions, focusing on providing affordable parallel computing access to individual researchers. This would suit scientists who required parallel computing capabilities but lacked access to traditional supercomputing resources. They wrote: “The Pile-of-PC methodology is still experimental, and does not match all of the valuable services provided by computer vendors. It is not for everyone. Rather, it is an emerging opportunity in the

high-performance computing field and complements rather than competes with the HPC industry commercial products” [59]. This positioning was consistent from Beowulf’s inception. In their foundational 1995 paper, Sterling *et al.* explicitly stated that “the Beowulf distributed computing system is specifically designed for single user workloads typical of high end scientific workstation environments,” contrasting it with “general purpose multiuser environments” [71]. Sterling reiterated this vision in 1998, that Beowulf was not aimed at replacing traditional computers. “Instead, we’re trying to augment genuine supercomputers with superior price performance for a lot of real-world problems” [47]. The 1998 NASA workshop assessment candidly acknowledged these limitations. Sterling and his co-authors reported that while Beowulf-class systems could deliver multi-gigaflops performance at unprecedented price-performance, “software environments were not fully functional or robust, especially for larger ‘dreadnought’-scale systems” [68]. The assessment noted that “few users are completely happy in either case at this stage” and that support for larger multi-user systems “is still an open question” [68]. These acknowledged gaps—in system software maturity, multi-user support, and scalability beyond modest node counts—were precisely the challenges that Roadrunner’s design set out to address through its integrated approach to system software, job scheduling, and high-performance networking. Fischer later clarified that Beowulf’s goal was “to complement the large parallel computers they were using” [22], and that the project arose fundamentally from the need for “software environments integration” and software sharing—not from an ambition to challenge supercomputer performance [26].

In his 2000 Computer History Museum talk [63] Sterling said Beowulf’s contributions were primarily “the majority of Ethernet drivers” and “channel bonding to support multiple simultaneous and many other low level tools for managing clusters of PCs.” The abstract for his talk says Beowulf systems could “equal in performance that of much more costly machines” while providing “a price-performance advantage of an order-of-magnitude or more.” Such benefits were, however, restricted to applications that could tolerate the communication bottlenecks inherent in commodity Ethernet networks.

### Trade-offs of the DIY Integration Model

Sterling characterizes the Beowulf approach as providing “tremendous flexibility” through its “just in place” integration model [66]. This approach offered distinct advantages by allowing organizations to customize systems according to their specific needs and budgets. However, this flexibility came with trade-offs that organizations needed to consider when adopting Beowulf clusters, as it would “more fully engage the talents of the on-site technical staff to enable operation than would a vendor provided turn-key system” [66].

The DIY integration model required organizations to assume system integration responsibilities typically handled by commercial vendors. This meant that adopting institutions needed to develop or maintain in-house expertise for system assembly, configuration, driver compatibility management, and ongoing maintenance. For organizations with existing technical staff, this represented an opportunity to gain deep system knowledge and maintain full control over their computing environment. However, for institutions without such expertise, these requirements could offset some of the initial cost savings through increased personnel needs or external consulting costs.

This integration model naturally favored organizations with strong technical capabilities and limited the adoption primarily to institutions that could effectively handle the system administration requirements. As a result, Beowulf clusters found their strongest adoption among technically sophisticated users who valued the flexibility and cost control that the approach provided. This emphasis on customization was intentional. As the Beowulf team noted, “no two Beowulf Pile-of-PCs. . . are exactly the same, although they all run the same software” [59]. This variability enabled what they termed “user-driven decisions” about system evolution, allowing organizations to “pick and choose from a wide array of sources, try things out, and change the configuration over time” rather than being “limited to the vendor’s current options lists which may be months out of date” [59]. For example, the Computational Biophysics Section at the National Institutes of Health built a Beowulf cluster in 1997 for cost-effective molecular modeling [15].

### Beowulf’s Legacy and Impact

Beowulf today is recognized as a pioneering COTS Linux cluster system using networked PCs. NASA calls Beowulf “the foundation of today’s high-end computing systems” and “a new model for enabling the efficient storage and retrieval of massive datasets and scalable parallel computing” [37]. After NASA built the first Beowulf cluster in 1994, others began building and deploying COTS systems running Linux. One of the advantages of Beowulf systems is their flexibility, which was not available in traditional supercomputers. There is no required Beowulf software, meaning researchers can bring their own software codes and tools to the system. Essentially, Beowulf clusters comprise Linux, the software and tools the researcher brings to the project, and an active community that promotes best practices, provides tutorials, and offers to help each other [40]. Beowulf is not solely responsible for the Linux cluster revolution that now dominates in supercomputing, building as it did on previous NOW and Pile-of-PCs approaches, but they did focus attention on the idea of deploying networked high-end workstations as a low-cost method of achieving more speed and performance. That philosophy was a relatively cheap and easy solution for some business enterprises and research scientists, but the sheer performance still lagged behind special-purpose commercial supercomputers. There was the space for a new project building on the ideas of Linux clusters, with pure performance as the major criterion.



Fig. 4. Linux prototype on lower-left, and Roadrunner (right). A Myricom dual 8-port SAN Myrinet switch sits on top of the left-most cabinet of the prototype, and four octal 8-port SAN Myrinet switches (not visible) connect Roadrunner. Above Roadrunner's console is a 72-port Foundry Fast Ethernet switch with Gigabit uplinks to the vBNS and Internet. (Image credit: Courtesy of the author.)

### 3 THE ROADRUNNER APPROACH TO LINUX SUPERCOMPUTING

In 1998 I developed the Roadrunner supercomputer that represented a different emphasis on Linux-based high-performance computing. Roadrunner's architecture integrated commercial off-the-shelf components with specialized high-performance networking technology, specifically incorporating a COTS network technology, Myrinet, which provided low-latency, high-bandwidth network. Unlike the Beowulf project, which was developed by a team of five engineers—Thomas Sterling, Don Becker, John Dorband, Jim Fischer, and Dan Jacob—with NASA institutional support, I designed, built, and deployed Roadrunner entirely on my own. There was no team. As a newly-arrived assistant professor at the University of New Mexico, I served simultaneously as system architect, kernel programmer, network engineer, system administrator, application porter, and benchmarker. Every line of kernel code I modified, every cable I connected, every driver I debugged, and every application I ported to Linux was my own work. This was not a collaborative research project with distributed responsibilities; it was a singular effort to prove that a Linux-based system could compete with commercial supercomputers.

My experience with commodity-based parallel computing predated the Roadrunner project by several years. In 1993, prior to the Beowulf project, I built a parallel computer using Ethernet-connected, Intel-based PCs running the FreeBSD operating system. After completing my Ph.D. in May 1996, I spent the next 18 months as a postdoc and National Science Foundation research associate at the University of Maryland's Institute for Advanced Computer Studies (UMIACS). During this period, I constructed an experimental computing cluster comprising 10 DEC AlphaServer nodes, each with four DEC Alpha RISC processors and DEC PCI cards connected to a DEC Gigaswitch ATM switch.

My work on Roadrunner began in January 1998 when I joined the University of New Mexico (UNM) as an assistant professor of electrical and computer engineering. As the principal investigator for the UNM Albuquerque High Performance Computing Center (AHPCC) SMP Cluster Computing Project, and working on my own, I constructed the first operational Intel/Linux supercomputer prototype (Roadrunner Phase 1) by April 1998, using eight dual, 333 MHz, Intel Pentium II nodes. This system used Red Hat Linux 5.0b with a custom 2.0.34 SMP kernel, later upgraded to custom 2.1.126 SMP kernel. In addition to the kernel modifications, this prototype required significant engineering work that I undertook single-handedly. Working alone, I ported essential software components to Linux to provide necessary system functionality, modified the Linux kernel and shell for running parallel applications, and ported scientific application codes from National Computational Science Alliance (NCSA) members to Linux—none of which had previously run on the Linux operating system. The Portable Batch System (PBS) job scheduler, originally designed by MRJ Technology Solutions for NASA's supercomputers, required hand-porting to Linux—work I completed without assistance from the original developers or any collaborators. Each component of the system stack, from low-level kernel modifications to high-level application tuning, passed through my hands alone.

Building a production supercomputer as a solo effort required a breadth of expertise that specialized teams typically distribute across multiple engineers. When the kernel crashed—which happened frequently during early development—there was no operating systems specialist to consult; I debugged it myself, often working late into the night tracing through kernel code to identify race conditions or memory management failures in the SMP implementation. When Myrinet drivers needed modification to work with my custom kernel, there was no network engineer to assign the task; I read the specifications, studied the existing driver code, and made the necessary changes. When scientific applications failed to compile or produced incorrect results on Linux, there was no applications team to investigate; I worked directly with the source code, identifying portability assumptions and compiler incompatibilities, then fixing them one by one. The three-network architecture—separating control, data, and diagnostic functions—emerged from my own experience





Fig. 6. Inside a Roadrunner cabinet with each node attached to three networks: Myrinet (ribbon cable), Fast Ethernet (CAT5), and Diagnostic (RS232 serial port). (Image credit: Courtesy of the author.)

NCSA strategically selected a diverse portfolio of national science community supercomputing applications to benchmark Roadrunner Phase 1's performance across multiple computational domains, using these results to inform their decision on proceeding with Roadrunner Phase 2 [4]. The comprehensive application suite included:

- **CACTUS**—A modular, manageable high-performance 3D Numerical Relativity Toolkit for solving Einstein equations numerically in computational astrophysics, testing the system's ability to handle gravitational wave physics calculations—a field that would later achieve breakthrough recognition with LIGO's 2017 Nobel Prize-winning detection of gravitational waves.
- **MILC**—MIMD Lattice Computation code for quantum chromodynamics (QCD), with the conjugate gradient algorithm for Kogut-Susskind quarks achieving over 60 MFLOPS/node for larger problems.
- **ARPI3D**—A 3-D numerical weather prediction model demonstrating the system's capabilities for atmospheric science applications.
- **BEAVIS**—Dynamic simulation of particle-laden viscous suspensions, evaluating performance for complex three-dimensional multiphase flow analysis spanning industrial and biological applications.
- **AIPS++**—Astronomical Information Processing System for radio astronomy data processing.
- **ASPCG**—A 2-D Navier-Stokes solver for computational fluid dynamics.

This benchmark suite was carefully chosen to test the system's versatility and computational capabilities across fundamental problems in scientific and engineering contexts, from quantum physics to weather prediction to astrophysics.

Based on demonstrations of this 16-processor prototype, the NSF and NCSA, led then by Larry Smarr, allocated \$400,000 to development. The resulting system, Roadrunner (Phase 2), was fully operational in April 1999 with hardware comprising 64 dual 450 MHz Intel Pentium II processors (128 processors total), 512 KB cache, 512 MB SDRAM with ECC, 6.4 GB IDE hard drives, and Myrinet interface cards. Roadrunner ranked among the 100 fastest supercomputers in the world when it went online in April 1999 [27].

The Roadrunner (Phase 2) system development proceeded on the following timeline in 1999:

- March 17: Received the first 32 nodes
- March 31: Received the next 32 nodes
- April 8: Running benchmarks on 64 nodes with Fast Ethernet
- May 3: Running Alliance test applications and benchmarks on 64 nodes with Myrinet
- June 1: Began test production
- June 15: Began test production with Alliance Grid (Globus)
- July 1: Allocations started for general user community

By the time allocations opened in July 1999, Roadrunner had attracted an early adopter user community of 78 researchers from 27 institutions including University of New Mexico (27 users), Rice University (5 users), NCSA (4 users), National Radio Astronomy Observatory (4 users), plus users from University of Texas at Austin, University of Oklahoma, University of Washington, Brown University, University of Illinois, Sandia National Laboratories, Arizona State University, Indiana University, Iowa State University, Los Alamos National Lab, MIT, Pennsylvania State University, Princeton University, University of Kentucky, and New Mexico State University. User projects spanned diverse scientific domains including modelling adhesion/adhesive wear, irregular parallel computation in linear scaling quantum chemistry,



Fig. 7. The launch of Roadrunner makes the news. “Machine One of 100 Speediest in World” with David Bader pictured at Roadrunner’s console. (Copyright: The Albuquerque Journal. Reprinted with permission. Permission does not imply endorsement.) [27].



Fig. 8. NCSA Director Larry Smarr (left), UNM President William Gordon, and U.S. Sen. Pete Domenici turn on the Roadrunner supercomputer in April 1999. (Image credit: Reprinted with permission of NCSA.)

PTreeSPH benchmarking, multiphase flow simulation, F15 turbulence simulation, gravitational waves from black hole collisions, molecular dynamics, galaxy formation, parallel least squares finite element methods, particle physics simulations, and weather modeling.

The system also incorporated specific supercomputing services that were absent in Beowulf clusters, including resource allocation, job scheduling, and monitoring capabilities. Roadrunner’s system software included the Red Hat Linux 5.2 operating system with a custom 2.2.10 SMP kernel, sets of compilers from both the GNU Compiler Collection and the Portland Group, and the Portable Batch System (PBS) job scheduler hand-ported to Linux. These management capabilities enabled Roadrunner to function closer to a turnkey commercial capability supercomputing platform that could support multiple simultaneous users running diverse applications across different scientific domains.

Unlike Beowulf clusters, which were designed primarily as parallel workstations for local computational tasks, Roadrunner was architected from its inception to be grid-enabled, specifically designed to integrate with distributed scientific infrastructure. Roadrunner became a foundational node on the National Technology Grid, providing researchers across disciplines and across the nation with seamless access to not only supercomputing capabilities but also geographically-distributed scientific instruments, datasets, and other supercomputers from their desktops. The Grid was envisioned as a way to give researchers access to an interconnected ecosystem of computational and scientific resources for large-scale problem solving from their desktops, no matter their location, through the nation’s fastest high-performance research networks. Alliance Director Larry Smarr likened the National Technology Grid to the power grid, where users could plug in and get the compute resources they needed, without having to worry about where those resources came from or their own location.

While Beowulf clusters could theoretically be retrofitted with grid middleware and connected to the grid, Roadrunner was purpose-built with grid integration as a core design requirement. This included native support for grid protocols, security frameworks, resource management systems, and the specialized software stack needed to seamlessly share resources across institutional boundaries—capabilities that would require significant additional engineering to retrofit onto Beowulf systems designed primarily for local parallel processing.

### A Tool for Enabling Science

Roadrunner’s performance on the Cactus application benchmark showed near-perfect scalability, outperforming systems such as NASA’s Beowulf cluster, NCSA’s Microsoft Windows NT cluster [50], and Silicon Graphics’ Origin 2000 [5]. Roadrunner’s design philosophy—optimizing for performance rather than simply minimizing cost—produced a system

that could effectively compete with traditional proprietary supercomputers on both performance and price. Roadrunner's entry into production in April 1999 as part of the National Science Foundation's National Technology Grid gave researchers across disciplines access to supercomputing capabilities from their desktops and established a blueprint for Linux supercomputing that would eventually become the dominant architecture in high-performance computing.

Edward Seidel, former head of the Numerical Relativity and E-Science Research Groups at the Albert Einstein Institute and now President of the University of Wyoming, recalls: "It was a very exciting time; Linux clusters were emerging as a huge force to democratize supercomputing and software frameworks providing community toolkits to solve broad classes of science and engineering problems were also taking shape. The collaboration we had between the Cactus team at the Albert Einstein Institute in Germany and David Bader's team with the Roadrunner supercluster was a pioneering effort that helped these movements gain traction around the world. The collaboration helped advance the goals of the Cactus team, led by Gabrielle Allen, whose efforts continue to this day as the underlying framework of the Einstein Toolkit. That toolkit now powers many efforts globally to address complex problems in multi-messenger astrophysics" [60].

Roadrunner transformed computational physics and astronomy research across multiple domains. Astrophysics researchers utilized the system's dual-processor architecture and Myrinet networking for large-scale cosmological simulations, with teams like Brandon Allgood's from UC Santa Cruz running PKDGRAV N-body codes to simulate 1.3 million particles representing cosmic structure formation [1]. Climate scientists, including Dan Weber and Kelvin Droegemeier, leveraged Roadrunner's superior network performance for weather prediction codes and detailed thunderstorm simulations, finding significant improvements in forecast turnaround time and resolution [8]. In fundamental physics, Steven Gottlieb's lattice quantum chromodynamics research used Roadrunner's balanced architecture for MILC collaboration calculations studying quark-gluon interactions, achieving sustained performance of 1.25–2.0 Gflops and enabling breakthrough observations of meson decay on the lattice [14], [21], [29], [30]. The system also enabled quantum chemistry advances through linear scaling SCF calculations and large-scale electronic structure studies of molecular systems like water clusters and polymers [20].

Engineering and materials science applications demonstrated Roadrunner's versatility across computational domains. In fluid dynamics, researchers used the system for parallelized RMA2 hydrodynamic modeling [56], advanced aircraft simulations using spectral/hp element methods for F-15 configurations [35], breakthrough parallel multipole algorithms for vorticity-based CFD methods that reduced computational complexity from  $O(N^2)$  to  $O(N \log N)$  [19], and environmental flow studies of toxic chemical dispersion with up to 29 million unknowns [77]. Computational electromagnetics researchers performed the first full-scale numerical simulations of Maxwell's equations for ultrashort optical pulses in nonlinear media [12], [13], while materials scientists conducted Monte Carlo simulations of charge carrier transport in organic semiconductors using  $64^3$  lattice sites [51], [52], [53]. Advanced materials research included comprehensive first-principles quantum mechanical calculations of defect physics and radiation effects in silicon dioxide for semiconductor applications [34], [54], [55].

Beyond traditional scientific computing, Roadrunner pioneered parallel algorithm development and distributed computing research. My research group developed novel graph algorithms achieving nearly linear speedup on problems with up to 256 million vertices [6], [7], [9], while Mohammad Mikki's distributed Barnes-Hut tree codes demonstrated 10–45% performance improvements through optimization techniques tested with up to 64,000 particles [48]. The system also served as a testbed for hierarchical broadcast algorithms showing 20–30% improvements over standard MPI implementations [73], parallel application sensitivity measurement tools [41], [42], distributed software design models [24], and the Adaptive Distributed Virtual Computing Environment (ADViCE) for middleware and virtualization research [38]. This diverse research portfolio established Roadrunner as a transformative platform that democratized supercomputing access and validated Linux-based clusters as viable alternatives to expensive proprietary systems across the computational science spectrum.

### Impact and Legacy of Roadrunner

The University of New Mexico's Albuquerque High Performance Computing Center pioneered a collaborative model between academic institutions and commercial Linux vendors that would reshape the supercomputing industry. An early example of this partnership was "BlackBear," a 16-node Linux cluster built around dual Intel Pentium III 550 MHz processors per node with 512 MB of SDRAM, running VA Linux Systems' Red Hat 6.0.4 distribution with a Linux 2.2.12smp kernel. BlackBear featured a dual-network architecture combining 10/100 BaseT Ethernet for control functions with a Myrinet interconnect for high-speed data communication, while its name honored New Mexico's official state animal and cultural heritage. This technical partnership with VA Linux Systems demonstrated the growing commercial viability of Linux-based supercomputing solutions and established a template for future academic-industry collaborations.

Building on this collaborative foundation, I also led the development of "LosLobos" with IBM following Roadrunner's construction—IBM's first Linux production system and a significant escalation in scale and ambition [2]. LosLobos premiered at number 24 on the Top500 supercomputer list in summer 2000, featuring 256 dual-processor Intel-based IBM servers with Myrinet connections (512 processors total) capable of 375 gigaflops, running Red Hat Linux 6.1 with a custom 2.2.13smp kernel. The knowledge IBM gained through this UNM collaboration enabled the company to create its first preassembled and preconfigured Linux server clusters for business within just one year, marking a crucial transition from experimental academic systems to commercial products.

These successful demonstrations of Linux-based high-performance computing attracted widespread industry attention and investment. Companies including IBM, VA Linux Systems, and Apple, established direct relationships with UNM to access the hardware configurations, software implementations, and design methodologies that had proven Linux clusters could compete with traditional supercomputers. The commercial stakes became evident when VA Linux Systems capitalized on this growing enthusiasm by going public on December 9, 1999, with a record-breaking IPO that saw its stock price surge 737% on the first day of trading [28], reflecting both investor confidence in Linux’s potential and broader market recognition of the paradigm shift toward open-source supercomputing. These academic-industry partnerships created crucial bridges that rapidly translated technical breakthroughs into market-ready products, democratizing access to supercomputing capabilities across the industry.

Roadrunner’s architectural innovations and operational success established a template for modern Linux-based supercomputing that continues to influence high-performance computing design and deployment strategies across the global research enterprise. This was solidified through Roadrunner’s performance achievements and practical viability. By successfully executing real-world scientific applications across multiple domains, Linux-based systems like Roadrunner provided concrete evidence that commodity-based architectures could serve as viable alternatives to traditional proprietary supercomputers in production environments. This proof of concept was particularly significant because it addressed longstanding concerns about the reliability, performance, and scalability of open-source computing platforms for mission-critical scientific research. The validation provided by Roadrunner’s success accelerated industry adoption: within just a few years, the approach developed by Roadrunner became the dominant paradigm in supercomputing, fundamentally altering market dynamics and procurement strategies.

Larry Smarr, Founding Director of NCSA, recalled the historical importance of Roadrunner’s development: “One of the most significant events that occurred in this period was when David [Bader] at University of New Mexico as a member of the Alliance created the first commercial off-the-shelf supercomputer, in other words a supercomputer built of PC server technologies, and he put it on the National Technology Grid. So here was a commodity-built, PC-based endpoint going into the technology grid. . . This is an historic event. It took resources from the Alliance, but it took David’s creative energies and innovation to do that” [62].

Roadrunner’s achievement validated that democratized computing approaches could scale to compete with the world’s most powerful systems, representing perhaps the most profound societal impact of its architectural approach. Prior to the emergence of Linux-based supercomputing, access to high-performance computing capabilities was largely restricted to well-funded government laboratories, major research universities, and large industrial corporations that could afford the substantial capital investments required for proprietary systems. The cost barriers were not merely financial but also technical, as these systems required specialized expertise for operation and maintenance that was scarce and expensive. Roadrunner’s demonstration that supercomputing capabilities could be achieved through more accessible technologies fundamentally altered this equation. Linux-based supercomputing opened pathways for a broader range of institutions to participate in cutting-edge computational research and innovation. Smaller universities, regional research institutions, and emerging technology companies could now access computational capabilities that had previously been beyond their reach. This democratization fostered a more inclusive scientific and technological ecosystem, enabling research breakthroughs from previously underrepresented institutions and geographic regions. The ripple effects extended beyond traditional academic and industrial research settings, as the reduced barriers to entry allowed innovative applications of supercomputing to emerge in fields ranging from financial modeling to entertainment and media production.

The economic impact of my Linux-based supercomputing design has been transformative on a global scale. Hyperion Research [33] quantified the remarkable economic contribution of this technological shift, finding that over the 25 years following Roadrunner’s introduction, Linux-based HPC systems contributed to the development of products and services worth more than \$100 trillion globally. This staggering figure reflects not only direct economic activity but also the multiplier effects of scientific discoveries, technological innovations, and industrial breakthroughs enabled by accessible high-performance computing. The economic impact extends across virtually every sector of the economy, from pharmaceutical development and materials science to climate modeling and artificial intelligence research. Most recently, the critical role of Linux-based HPC became evident during the COVID-19 pandemic, when these systems powered the computational research that enabled rapid vaccine development, epidemiological modeling, and public health response strategies [17]. The ability to rapidly deploy computational resources for urgent societal challenges demonstrated the strategic importance of maintaining robust, accessible supercomputing infrastructure based on open technologies.

As the global economy continues to evolve and worldwide challenges increasingly threaten human wellbeing—from climate change and pandemics to resource scarcity and geopolitical instability—Linux supercomputers continue to serve as the powerhouse systems that drive economic growth, solve complex problems, and ensure collective safety and security. The architectural template established by Roadrunner has proven remarkably durable and adaptable, providing a foundation for continuous innovation in computational science and engineering that remains as relevant today as it was at the dawn of the new millennium.

## 4 CONCLUSIONS

The divergent approaches represented by Beowulf and Roadrunner provide insights into high-performance system design philosophy. Beowulf clusters emerged as an accessible approach to parallel computing, democratizing access through

a design philosophy focused exclusively on mass-market components. The Beowulf project brought awareness to the potential of COTS components in HPC; however, this approach came with certain architectural constraints, particularly in network performance and system management capabilities, which affected its suitability for communication-intensive workloads and multi-user environments.

Roadrunner’s design philosophy took a different path, integrating commercial off-the-shelf components with specialized networking technology and implementing comprehensive resource management capabilities. This balanced approach prioritized overall system performance and usability, enabling the support of multiple simultaneous users across diverse scientific domains. By optimizing for computational efficiency and flexibility, Roadrunner established a blueprint for Linux supercomputing that effectively combined the cost advantages of commodity components with the performance characteristics needed for demanding scientific applications.

The fact that Roadrunner was designed, built, and deployed by a single individual—without the institutional infrastructure, dedicated staff, or team-based approach that characterized both the Beowulf project and traditional supercomputer development—demonstrates that transformative contributions to high-performance computing need not require large teams or extensive resources. What Roadrunner required was a willingness to work across the entire system stack, from kernel internals to application algorithms, and the determination to solve every problem personally rather than delegating to specialists who did not exist.

This historical comparison illuminates an important lesson for computing system design: system developers can choose different approaches based on their target applications and user communities. The Beowulf team’s strict adherence to mass-market commodity components successfully served individual researchers seeking affordable parallel computing access, while Roadrunner’s integration of commodity and specialized components addressed the needs of multi-user supercomputing environments. Both approaches made valuable contributions to the evolution of Linux-based high-performance computing. Beowulf demonstrated the viability of commodity cluster computing and democratized access to parallel processing, while Roadrunner established an architectural template that balanced cost-effectiveness with performance requirements. The success of both approaches demonstrates that effective system design depends on clearly understanding target requirements and user needs rather than adhering to a single universal philosophy.

The validation of Roadrunner’s architectural approach is demonstrated by examining the specific design elements that define modern supercomputing. By 2017, Linux-based systems achieved 100% dominance of the world’s fastest 500 supercomputers [75], but more significantly, these systems universally adopted Roadrunner’s core architectural principles rather than Beowulf’s design philosophy. Modern supercomputers employ multi-core nodes connected via high-performance interconnects—exactly the template Roadrunner established with its dual-processor nodes and Myrinet networking, not Beowulf’s uniprocessor-plus-Ethernet approach. Today’s exascale computing systems, capable of calculating at least  $10^{18}$  floating point operations per second, require the sophisticated resource management, multi-user capabilities, and scalable networking architecture that Roadrunner pioneered. While Beowulf demonstrated the viability of commodity components, its design constraints—single processors per node, commodity Ethernet networking, and lack of system management—proved inadequate for the communication-intensive, massively parallel workloads that define serious supercomputing. Roadrunner’s architectural template succeeded because it combined commodity economics with performance-oriented design choices, creating a blueprint that could scale from hundreds to millions of processors while maintaining the cost advantages that made Linux supercomputing accessible.

The architectural template that Roadrunner established—commodity processors, high-performance interconnects, multi-user resource management—now underpins systems from departmental clusters to exascale machines. The lessons from this transition period remain instructive: that effective parallel system design requires matching architectural choices to target applications and user communities, and that open platforms can achieve performance competitive with proprietary alternatives. As the field confronts new challenges in heterogeneous computing, AI acceleration, and energy-efficient design, the debates between accessibility and performance that distinguished Beowulf from Roadrunner will continue to shape the next generation of parallel and distributed systems.

## ACKNOWLEDGMENTS

The work of David A. Bader is supported in part by NSF grants CCF-2109988, OAC-2402560, and CCF-2453324. The author wishes to thank Troy Kaighin Astarte, Lecturer at Swansea University, and Brian Carpenter, honorary professor at the University of Auckland, for their contributions to this article. We acknowledge Karen Green for her copyediting that improved the readability of this article.

## REFERENCES

- [1] B. Allgood, J. Stadel, and T. Quinn, “PKDGRAV Performance Testing,” University of Washington, Seattle, 1999. [Online]. Available: <https://faculty.washington.edu/trq/hpcc/faculty/trq/brandon/>
- [2] “Alliance Introduces 512-Processor LosLobos Supercluster,” *University of New Mexico*, Albuquerque, NM, 21 Mar. 2000. [Online]. Available: <https://web.archive.org/web/20000819060614/http://www.unm.edu/~paaffair/news/news%20releases/Mar21hpcc.html>
- [3] T. E. Anderson, D. E. Culler, and D. A. Patterson, “The Berkeley Networks of Workstations (NOW) Project,” in *Digest of Papers. COMPCON’95. Technologies for the Information Superhighway*, IEEE Comput. Soc. Press, San Francisco, CA, USA, 1995, pp. 322–326.
- [4] D. Bader, “Analysis of the Alliance/UNM Roadrunner Linux Supercluster,” *NSF/NCSA Alliance Roadmap ’99 Meeting*, Chicago, IL, 12 May 1999. [Online]. Available: <https://davidbader.net/events/19990512-alliance/>

- [5] D. Bader, "SuperClusters: A New Approach for High-Performance Computing," NCSA Chautauqua, University of New Mexico, 10 Aug. 1999. [Online]. Available: <https://davidbader.net/talk/ncsa-chautauqua-talk-1-superclusters-a-new-approach-for-high-performance-computing/>
- [6] D. A. Bader, "A Practical Parallel Algorithm for Cycle Detection in Partitioned Digraphs," Technical Report, University of New Mexico, 1999. [Online]. Available: [https://digitalrepository.unm.edu/cgi/viewcontent.cgi?article=1044&context=ece\\_rpts](https://digitalrepository.unm.edu/cgi/viewcontent.cgi?article=1044&context=ece_rpts)
- [7] D. A. Bader, "High-Performance Algorithms and Applications for SMP Clusters," in *5th NASA HPC/CAS Workshop*, NASA Ames, Moffett Field, CA, Feb. 2000. [Online]. Available: <https://ntrs.nasa.gov/citations/20000064579>
- [8] D. A. Bader, "Linux and Supercomputing: How My Passion for Building COTS Systems Led to an HPC Revolution," *IEEE Ann. Hist. Comput.*, vol. 43, no. 3, pp. 73–80, Jul. 2021.
- [9] D. A. Bader and A. K. Illendula, "An Experimental Comparison of Parallel Algorithms for Ear Decomposition of Graphs using Two Leading Paradigms," Technical Report, University of New Mexico, 1 May 2000. [Online]. Available: [https://digitalrepository.unm.edu/ece\\_rpts/2/](https://digitalrepository.unm.edu/ece_rpts/2/)
- [10] C. G. Bell, "Personal Communication," 26 Feb. 2021. [Online]. Available: [https://ieeemilestones.ethw.org/Milestone-Proposal:Linux-based\\_Supercomputing](https://ieeemilestones.ethw.org/Milestone-Proposal:Linux-based_Supercomputing)
- [11] C. G. Bell and J. Gray, "What's next in high-performance computing?," *Commun. ACM*, vol. 45, no. 2, pp. 91–95, Feb. 2002.
- [12] P. M. Bennett, "Parallel Numerical Integration of Maxwell's Full—Vector Equations in Nonlinear Focusing Media," Ph.D. dissertation, Univ. New Mexico, Dec. 2000.
- [13] P. M. Bennett and A. Aceves, "Numerical integration of Maxwell's full-vector equations in nonlinear focusing media," *Physica D: Nonlinear Phenomena*, vol. 184, no. 1–4, pp. 352–375, Oct. 2003.
- [14] C. Bernard *et al.*, "QCD spectrum with three quark flavors," *Phys. Rev. D*, vol. 64, no. 5, p. 054506, Aug. 2001.
- [15] E. Billings, "Lots of Boxes of Shelves (LoBoS) Supercomputers," n.d. [Online]. Available: <https://web.archive.org/web/19981212030622/https://www.lobos.nih.gov/>
- [16] N. J. Boden *et al.*, "Myrinet: a gigabit-per-second local area network," *IEEE Micro*, vol. 15, no. 1, pp. 29–36, Feb. 1995.
- [17] J. Brase *et al.*, "The COVID-19 High-Performance Computing Consortium," *Comput. Sci. Eng.*, vol. 24, no. 1, pp. 78–85, Jan. 2022.
- [18] E. Brooks, "Attack of the Killer Micros," in *Teraflop Computing Panel, 1990 ACM/IEEE Conf. Supercomputing*, New York, NY, Nov. 1990.
- [19] M. J. Brown, A. A. Mammoli, and M. S. Ingber, "Parallel multipole implementation of the generalized Helmholtz decomposition for solving viscous flow problems," *Numer. Meth. Eng.*, vol. 58, no. 11, pp. 1617–1635, Nov. 2003.
- [20] M. Challacombe, "A general parallel sparse-blocked matrix multiply for linear scaling SCF theory," *Comput. Phys. Commun.*, vol. 128, no. 1–2, pp. 93–107, Jun. 2000.
- [21] N. H. Christ, "Computers for lattice QCD," *Nucl. Phys. B—Proc. Suppl.*, vol. 83–84, pp. 111–115, Apr. 2000.
- [22] "COE alumnus honored for supercomputing innovations," *Engineering Communications*, North Carolina State University, 6 Dec. 2022. [Online]. Available: <https://enr.ncsu.edu/news/2022/12/06/coe-alumnus-honored-for-supercomputing-innovations/>
- [23] Cray Research, Inc., "The CRAY-YMP Series of Computer Systems," 1989. [Online]. Available: [https://cray-history.net/wp-content/uploads/2021/08/Y-MP8D\\_red\\_redux.pdf](https://cray-history.net/wp-content/uploads/2021/08/Y-MP8D_red_redux.pdf)
- [24] M. Dwivedula, S. Hariri, and M. Parashar, "A software design model for parallel applications on heterogeneous systems," in *Proc. 16th Int. Parallel Distrib. Process. Symp.*, IEEE, Ft. Lauderdale, FL, 2002, p. 9 pp.
- [25] S. Elbert, "PPC Workshop Summary and Lessons Learned," *Pentium Pro Cluster Workshop*, Des Moines, Iowa, 10 Apr. 1997. [Online]. Available: <https://web.archive.org/web/20001002145033/http://www.scl.ameslab.gov/workshops/Talks/Elbert/index.htm>
- [26] J. R. Fischer, "The Roots of Beowulf," in *Proc. 20 Years of Beowulf Workshop*, ACM, Annapolis, MD, 13 Oct. 2014, pp. 1–6.
- [27] J. Fleck, "UNM To Crank Up \$400,000 Supercomputer Today," *Albuquerque Journal*, 8 Apr. 1999, p. D1.
- [28] J. Glasner, "VA Linux Sets IPO Record," *Wired Magazine*, 9 Dec. 1999. [Online]. Available: <https://www.wired.com/1999/12/va-linux-sets-ipo-record/>
- [29] S. Gottlieb, "Comparing clusters and supercomputers for lattice QCD," *Nucl. Phys. B—Proc. Suppl.*, vol. 94, no. 1–3, pp. 833–840, Mar. 2001.
- [30] S. Gottlieb, "Cost-effective clustering," *Comput. Phys. Commun.*, vol. 142, no. 1–3, pp. 43–48, Dec. 2001.
- [31] W. D. Hillis and L. W. Tucker, "The CM-5 Connection Machine: a scalable supercomputer," *Commun. ACM*, vol. 36, no. 11, pp. 31–40, Nov. 1993.
- [32] J. Jájá, *An Introduction to Parallel Algorithms*. Reading, MA: Addison-Wesley, 1992.
- [33] E. Joseph, M. Riddle, T. Sorensen, and S. Conway, "Special Study: The Economic and Societal Benefits of Linux Supercomputers," Hyperion Research, Apr. 2022. [Online]. Available: <https://davidbader.net/publication/2022-hyperionresearch/>
- [34] S. P. Karna, A. C. Pineda, R. D. Pugh, W. M. Shedd, and T. R. Oldham, "Electronic structure theory and mechanisms of the oxide trapped hole annealing process," *IEEE Trans. Nucl. Sci.*, vol. 47, no. 6, pp. 2316–2321, Dec. 2000.
- [35] G. E. Karniadakis, "Final Report: Dynamic DNS/LES of transition and turbulence in high-speed flows and flow-structure interactions," AFOSR, 31 Dec. 2000. [Online]. Available: <https://apps.dtic.mil/sti/tr/pdf/ADA390165.pdf>
- [36] D. S. Katz and T. Cwik, "Cost-Effective Parallel Computational Modeling," in *8th Biennial IEEE Conf. Electromagnetic Field Computation*, IEEE Magnetics Soc., Tucson, AZ, Jun. 1998.
- [37] S. Keefe, "Satellites, the Space Race, and Supercomputing: How NASA Goddard's Beowulf Cluster Computer Became an Award-Winning Space Technology," NASA NCCS, 13 May 2022. [Online]. Available: <https://www.nccs.nasa.gov/news-events/nccs-highlights/beowulf>
- [38] D. Kim, "An adaptive distributed virtual computing environment (ADVICE): Design and evaluation," Ph.D. dissertation, Syracuse University, Dec. 2000. [Online]. Available: [https://surface.syr.edu/eecs\\_etd/122/](https://surface.syr.edu/eecs_etd/122/)
- [39] P. D. Lax, "Report of the Panel on Large Scale Computing in Science and Engineering," Dept. Defense and NSF, Dec. 1982. [Online]. Available: [https://science.osti.gov/-/media/ascr/pdf/program-documents/archive/Lax\\_report.pdf](https://science.osti.gov/-/media/ascr/pdf/program-documents/archive/Lax_report.pdf)
- [40] J. Layton, "How Linux and Beowulf Drove Desktop Supercomputing," *HPC Admin Magazine*, 2025. [Online]. Available: <https://www.admin-magazine.com/HPC/Articles/How-Linux-and-Beowulf-Drove-Desktop-Supercomputing/>
- [41] E. A. León Borja, "An MPI Tool to Measure Application Sensitivity to Variation in Communication Parameters," Ph.D. dissertation, Univ. New Mexico, May 2003.
- [42] E. A. León, A. B. Maccabe, and R. Brightwell, "An MPI Tool to Measure Application Sensitivity to Variation in Communication Parameters," in *Recent Advances in PVM and MPI*, J. Dongarra, D. Laforenza, and S. Orlando, Eds. Berlin: Springer, 2003, pp. 108–111.
- [43] J. Lindheim, *Beowulf Tutorial: Building a Beowulf System*, California Institute of Technology, 25 Aug. 2000. [Online]. Available: <https://web.archive.org/web/20001121163100/http://www.cacr.caltech.edu/beowulf/tutorial/building.html>
- [44] M. J. Litzkow, M. Livny, and M. W. Mutka, "Condor—a hunter of idle workstations," in *Proc. 8th Int. Conf. Distributed Computing Systems (ICDCS)*, IEEE, San Jose, CA, 1988, pp. 104–111.
- [45] Los Alamos National Laboratory, "Avalon, Frequently Asked Questions," n.d. [Online]. Available: <https://web.archive.org/web/20000824181455/http://cnls.lanl.gov/avalon/FAQ.html>
- [46] T. G. Mattson, D. Scott, and S. Wheat, "A TeraFLOP supercomputer in 1996: the ASCI TFLOP system," in *Proc. Int. Conf. Parallel Processing*, IEEE, Honolulu, HI, 1996, pp. 84–93.
- [47] S. McGrath, "Supercomputing Gets A New Hero," *Communications News*, 1 Aug. 1998. [Online]. Available: <https://www.thefreelibrary.com/Supercomputing+gets+a+new+hero.-a021071072>
- [48] M. A. Mikki, "Distributed tree code on cluster of workstations," *Islamic Univ. Gaza J. Natural Studies*, vol. 10, no. 1, pp. 43–73, 2002.

- [49] NASA, "Beowulf Clusters Make Supercomputing Accessible," 2020. [Online]. Available: [https://spinoff.nasa.gov/Spinoff2020/it\\_1.html](https://spinoff.nasa.gov/Spinoff2020/it_1.html)
- [50] "NCSA's NT supercluster achieves supercomputer performance," *HPCwire*, 27 Apr. 1998. [Online]. Available: <https://www.hpcwire.com/1998/04/27/ncsas-nt-supercluster-achieves-supercomputer-performance/>
- [51] S. V. Novikov, "Charge carrier diffusion in energy landscape created by static charges: Poole-Frenkel model revised," *Phys. Status Solidi (b)*, vol. 236, no. 1, pp. 119–128, Mar. 2003.
- [52] S. V. Novikov, "Rough electrode surface: effect on charge carrier injection and transport in organic devices," *Macromol. Symp.*, vol. 212, no. 1, pp. 191–200, Apr. 2004.
- [53] S. V. Novikov and A. V. Vannikov, "Charge Carrier Transport in Nonpolar Disordered Organic Materials: What is the Reason for Poole-Frenkel Behavior?," *Mol. Cryst. Liq. Cryst.*, vol. 361, no. 1, pp. 89–94, May 2001.
- [54] A. C. Pineda and S. P. Karna, "Effect of Hole Trapping on the Microscopic Structure of Oxygen Vacancy Sites in  $\alpha$ -SiO<sub>2</sub>," *J. Phys. Chem. A*, vol. 104, no. 20, pp. 4699–4703, May 2000.
- [55] A. C. Pineda, S. P. Karna, H. A. Kurtz, W. M. Shedd, and R. D. Pugh, "The effect of network topology on proton trapping in amorphous SiO<sub>2</sub>," *IEEE Trans. Nucl. Sci.*, vol. 48, no. 6, pp. 2081–2085, Dec. 2001.
- [56] P. Rao, "A parallel RMA2 model for simulating large-scale free surface flows," *Environ. Modelling & Software*, vol. 20, no. 1, pp. 47–53, Jan. 2005.
- [57] Red Hat Software, Inc., "Announcing Extreme Linux," 13 May 1998. [Online]. Available: <https://web.archive.org/web/20000229181702/http://charlotte.redhat.com/product.phtml/EX1000>
- [58] C. Reschke, T. Sterling, D. Ridge, D. Savarese, D. Becker, and P. Merkey, "A design study of alternative network topologies for the Beowulf parallel workstation," in *Proc. 5th IEEE Int. Symp. HPDC*, IEEE, Syracuse, NY, 1996, pp. 626–635.
- [59] D. Ridge, D. Becker, P. Merkey, and T. Sterling, "Beowulf: harnessing the power of parallelism in a pile-of-PCs," in *1997 IEEE Aerospace Conf.*, IEEE, Snowmass, CO, 1997, pp. 79–91 vol.2.
- [60] E. Seidel, "Personal Communication," 25 Feb. 2021.
- [61] H. J. Siegel *et al.*, "PASM: A Partitionable SIMD/MIMD System for Image Processing and Pattern Recognition," *IEEE Trans. Comput.*, vol. C–30, no. 12, pp. 934–947, Dec. 1981.
- [62] L. Smarr, "Toward a National Research Platform to Enable Data-Intensive Computing," *Institute for Data Science Seminar Series*, NJIT, 27 Oct. 2021. [Online]. Available: <https://www.youtube.com/live/HO1dhtV-Pbg>
- [63] T. Sterling, "Beowulf-class PC Clusters: An Historical Perspective," Computer History Museum, 13 Apr. 2000. [Online]. Available: <https://computerhistory.org/events/beowulfclass-pc-clusters-an-historical/>
- [64] T. Sterling, "Pile of PCs: Unthinkable, Obvious, Inevitable: A Beowulf Perspective," *Pentium Pro Cluster Workshop*, Des Moines, Iowa, 10 Apr. 1997. [Online]. Available: <https://web.archive.org/web/19980131191706/http://www.scl.ameslab.gov/workshops/Talks/Sterling/sterling.cover.html>
- [65] T. Sterling, "The scientific workstation of the future may be a pile of PCs (Viewpoint Column)," *Commun. ACM*, vol. 39, no. 9, pp. 11–12, Sept. 1996.
- [66] T. Sterling, "Tutorial: How to build a Beowulf (PC Cluster)," *Cluster Computing Conf. (CCC)*, Emory University, Atlanta, GA, 9 Mar. 1997. [Online]. Available: <https://web.archive.org/web/19990508124540/http://www.mathcs.emory.edu/ccc97/tutorials.html>
- [67] T. Sterling, D. Becker, and D. Savarese, "Distributed Computing with Linux on the Beowulf Parallel Workstation," *DEC User's Symposium*, Washington, DC, 10 May 1995. [Online]. Available: <https://web.archive.org/web/19970710051129/http://cesdis.gsfc.nasa.gov/linux/talks/DECUS.html>
- [68] T. Sterling, D. Becker, M. Warren, T. Cwik, J. Salmon, and B. Nitzberg, "An assessment of Beowulf-class computing for NASA requirements: initial findings from the first NASA workshop on Beowulf-class clustered computing," in *1998 IEEE Aerospace Conf. Proc.*, IEEE, Snowmass at Aspen, CO, 1998, pp. 367–381.
- [69] T. L. Sterling, Ed., *Beowulf Cluster Computing with Linux*. Cambridge, MA: MIT Press, 2001.
- [70] T. L. Sterling, Ed., *How to Build a Beowulf: A Guide to the Implementation and Application of PC Clusters*. Cambridge, MA: MIT Press, 1999.
- [71] T. L. Sterling, D. Savarese, D. J. Becker, J. E. Dorband, U. A. Ranawake, and C. V. Packer, "BEOWULF: A Parallel Workstation for Scientific Computation," in *Proc. 1995 Int. Conf. Parallel Processing*, CRC Press, 1995, pp. 11–14.
- [72] T. Sterling, D. Savarese, D. J. Becker, B. Fryxell, and K. Olson, "Communication overhead for space science applications on the Beowulf parallel workstation," in *Proc. 4th IEEE Int. Symp. HPDC*, IEEE, Washington, DC, 1995, pp. 23–30.
- [73] Y. Sun, X. Lin, Y. Ling, and K. Li, "Broadcast on Clusters of SMPs with Optimal Concurrency," in *Proc. Int. Conf. PDPTA*, CSREA Press, Las Vegas, NV, 2002, pp. 1558–1564.
- [74] The Computer History Museum, "The Cray-1 Supercomputer," n.d. [Online]. Available: <https://www.computerhistory.org/revolution/supercomputers/10/7>
- [75] The Linux Foundation, "Linux Runs All of the World's Fastest Supercomputers," 20 Nov. 2017. [Online]. Available: <https://www.linuxfoundation.org/blog/blog/linux-runs-all-of-the-worlds-fastest-supercomputers>
- [76] D. Tolle, "A collection of NCUBE UNIX utilities," in *Proc. 3rd Conf. Hypercube Concurrent Computers and Applications*, ACM, Pasadena, CA, 1988, pp. 832–834.
- [77] T. H. Tsang, S. A. Yost, and Z. Bai, "1999 Progress Report: Parallel Least-Squares Finite Element Method for Large Eddy Simulation of Large Scale Environmental Flows and Transport Processes," EPA, 31 Oct. 1999. [Online]. Available: [https://cfpub.epa.gov/ncer\\_abstracts/index.cfm/fuseaction/display.abstractDetail/abstract\\_id/721/report/1999](https://cfpub.epa.gov/ncer_abstracts/index.cfm/fuseaction/display.abstractDetail/abstract_id/721/report/1999)
- [78] M. S. Warren, "Loki: Commodity Parallel Processing in Practice," *Pentium Pro Cluster Workshop*, Des Moines, Iowa, 10 Apr. 1997. [Online]. Available: <https://web.archive.org/web/19980131191932/http://www.scl.ameslab.gov/workshops/Talks/Warren/warren.cover.html>
- [79] M. S. Warren, D. J. Becker, M. P. Goda, J. K. Salmon, and T. L. Sterling, "Parallel Supercomputing with Commodity Components," in *Int. Conf. PDPTA*, Las Vegas, NV, 30 Jul. 1997. [Online]. Available: [https://digital.library.unt.edu/ark:/67531/metadc695974/m2/1/high\\_res\\_d/522740.pdf](https://digital.library.unt.edu/ark:/67531/metadc695974/m2/1/high_res_d/522740.pdf)

**David A. Bader** is a Distinguished Professor and founder of the Department of Data Science and inaugural Director of the Institute for Data Science at New Jersey Institute of Technology. Prior to this, he served as founding Professor and Chair of the School of Computational Science and Engineering, College of Computing, at Georgia Institute of Technology. Bader is a Fellow of the IEEE, ACM, AAAS, and SIAM and has co-authored more than 400 scholarly papers. Bader is the 2021 recipient of the IEEE Sidney Fernbach Award; the 2022 Innovation Hall of Fame inductee of the University of Maryland's A. James Clark School of Engineering; a 2025 inductee of the Mimms Museum of Technology and Art's Hall of Fame; and the 2025 recipient of the Heatherington Award for Technological Innovation. In 2025, *HPCwire* named Bader as one of its "35 Legends" of HPC.