High Performance Computing

Clusters, Constellations, MPPs, and Future Directions

Jack Dongarra

University of Tennessee

Thomas Sterling

California Institute of Technology

Horst Simon

Erich Strohmaier

Lawrence Berkeley National Laboratory

June 10, 2003

Abstract

Last year's paper by Bell and Gray [1] examined past trends in high performance computing and asserted likely future directions based on market forces. While many of the insights drawn from this perspective have merit and suggest elements governing likely future directions for HPC, there are a number of points put forth that we feel require further discussion and, in certain cases, suggest alternative, more likely views. One area of concern relates to the nature and use of key terms to describe and distinguish among classes of high end computing systems, in particular the authors' use of "cluster" to relate to essentially all parallel computers derived through the integration of replicated components. The taxonomy implicit in their previous paper, while arguable and supported by some elements of our community, fails to provide the essential semantic discrimination critical to the effectiveness of descriptive terms as tools in managing the conceptual space of consideration. In this paper, we present a perspective that retains the descriptive richness while providing a unifying framework. A second area of discourse that calls for additional commentary is the likely future path of system evolution that will lead to effective and affordable Petaflops-scale computing including the future role of computer centers as facilities for supporting high performance computing environments. This paper addresses the key issues

of taxonomy, future directions towards Petaflops computing, and the important role of computer centers in the 21st century.

1. Introduction

In their paper [1] of last year, Bell and Gray put forth a view of the past, present, and future of high performance computing that is both insightful and thought provoking. Identifying key trends with a grace and candor rarely encountered in a single work, the authors describe an evolutionary past, drawn from their vast experience and project a future for HPC that is both enticing and compelling. Yet, implicit in this important treatment is a number of underlying assumptions, particularly related to terminology and dominant trends, that is in conflict with our own experiences and common practices, as well as our shared view of the future directions for high performance computing. Taken from our vantage points of the Top500 list [2], the Lawrence Berkeley Laboratory NERSC computer center [3], Beowulf-class computing [4], and research in Petaflops-scale computing architecture [5], we offer an alternate perspective on a number of key issues in the form of constructive counterpoint, continuing the dialogue elicited by the previous authors.

Terminology and taxonomies are subjective. No absolute truths exist (except that no absolute truths exist), and common usage dictates practical utility, even when self-contradictory or incomplete in meaning. Yet, in spite of it's imperfections, technical nomenclature can be a powerful tool for describing, distinguishing, and delineating among related concepts, entities, and processes. In their recent paper, Bell and Gray incorporate a fundamental assumption throughout their reasoning, which while defensible and advocated by some notable colleagues in the field [6], nonetheless corrupts the power of our terminology as a tool to represent and differentiate. Specifically, implicit in their paper is the assertion that essentially every parallel system employing replicated resources is a *cluster*. In a well-intentioned effort to provide a unifying principle, the authors have eliminated a powerful concept even as they intended to reinforce it. The concept of the commodity cluster is one that has driven an important trend in parallel processing over the last decade, delivering unprecedented performance-to-cost and providing exceptional flexibility and technology tracking. By expanding the scope of the term "cluster", they have deprived it of its seminal meaning and implication. One objective of this paper is to restore the strength and value of the term "cluster" by degeneralizing its applicability to a restricted subset of parallel computers. This paper further considers this class in conjunction with its complementing terms "Constellation", "Beowulf-class", and "MPP", based on the classification employed by the TOP500 list, which has tracked the field of high performance computer systems for almost a decade. In so doing, we are motivated to reflect on other aspects of the author's assertions about the meaning and value of these terms as well. We find to their credit that our own previous assumptions, too, lack sufficiency upon close examination. We offer a revised framework and taxonomy that differs from both that of Bell and Gray and

our own previous usage on the Top-500 classification. This conceptual infrastructure is founded on the principle dimensions that determine the nature of parallel computing structures and distinguish among them.

Such delineation guides consideration of future directions for high performance computing structures and methods as well. Certainly, as the previous authors convincingly articulate, the impact of Moore's law and economy of scale of mass market computing components in easily integrated ensembles will have a significant, even dominant, impact on the evolution of high performance systems in the near future. The Top-500 list already clearly reflects this trend with at least half of all systems represented on the list being products of some form of clustering (as designated by NOW-clusters or Constellations) and, as Bell and Gray point out, Beowulf-class clusters are having a more significant impact on the medium to high scale throughout the science and technical computing arena as well as the commercial sector. Also referred to as Linux clusters or PC clusters, Beowulfs are perhaps more widely used than any other type of parallel computer because of their low cost, flexibility, and accessibility. Indeed, among the top 5 systems on the most recent list, three are commodity clusters, one of which is a Linux cluster, not unlike the original Beowulf class systems. But many aspects of this strategy are limited in their capabilities for many important classes of application and fail to fully exploit the potential of the underlying technology that custom architectures could employ through superior organization and mechanisms. The difference might be between one and two orders of magnitude of sustained performance. The class of vector supercomputers cited by the previous authors is only one possible such structure and not necessarily the most important across a broad range of applications, in spite of the recent impressive demonstrations of the Japanese Earth Simulator. The long-term future of high performance computer architecture may evolve other innovative structures that support new paradigms of execution models to greatly enhance efficiency in terms of performance, cost, space, and power while enabling scalability to tens or hundreds of Petaflops. The conceptual framework offered here implies the directions of such developments and resonates with recent advances being pursued by the computer architecture research community.

One consequence of the progress anticipated beyond that envisioned by the previous authors is the form and content of future computer centers, which will evolve as the available technologies and system architecture classes advance. Instead of becoming obsolete, the role of the computer center is likely to grow in importance, evolving to meet the support challenges of new architectures, programming models, mass storage, and accessibility via the *grid*. The emergence of Beowulf and other commodity clusters will definitely alter the mix of resources that will comprise a medium to large computer center. But the responsibilities and services that constitute the reason d'etre of such facilities will continue to prove of critical importance especially to the high end computing and large data archive communities. Already we see within the DOE and NSF sector the development of new and larger computing centers (e.g. LANL, LLNL, ORNL)

to house the next generation high-end systems including very large commodity Linux clusters, successors to the original Beowulf clusters. Computer centers of the future will be charged with the administration, management, and training associated with bringing these major resources to bear on mission critical applications.

This paper, while congratulating Bell and Gray in opening up this line of discourse, offers a constructive expansion on their original themes and seeks to correct specific areas of their premise with which we take exception. Section 2 of this paper presents a detailed discussion on Beowulf-class computing and commodity clusters with the objective of establishing a clear distinction between these systems and the broad range of alternative parallel computing architectures. Section 3 then expands on this initial codification to provide an alternative conceptual framework with which to classify all general-purpose parallel computer architectures, differing not only with the previous authors but also with our own categorization employed in the Top-500 list. Section 4 employs this breakdown to project future directions for supercomputing taking in to consideration the opportunities provided by new approaches and the early insights being derived from the DARPA High Productivity Computing Systems program industry projects. Section 5 explores the role of the computer center within this context noting the many important services that only computer centers will be capable of providing in support of Petaflops scale supercomputers. Finally, in Section 6 like Bell and Gray, we suggest areas of investment by the Federal government and the high end computing industry to advance the state of the art, resolving critical problems and devising innovative approaches and solutions to ultimately realize the objective of effective Petaflops scale computers.

2. Commodity Clusters

Bell and Gray, in conjunction with a number of their distinguished colleagues, see an important unifying principle emerging in the evolution of high performance computing, where the integration of highly replicated components, many of which are designed and fabricated for more general markets (e.g. microprocessors, DRAM), is providing the driving force for a convergent architecture. They call this architecture “clusters” and distinguish it only from the minority set of vector supercomputers (e.g. NEC SX-6, Cray X1), which exploit vectors in custom processor architecture designs. This convergent architecture model of the evolution of supercomputer design is compelling, readily apparent, and wrong. We respectfully assert an alternate perspective; one that is rich in detail and has value in its ability as an enabling framework for reasoning about computing structures and methods. In this section we discuss the narrow topic of the cluster, and then in the following section expand this to encompass a complete taxonomic framework.

In particular, we assert that the term “cluster” is best employed, not as a synonym for essentially the universal set of parallel computer system organizations, but rather as a specific class of such systems. Therefore we state that:

NOT everything is a cluster

We limit the scope of the definition of a cluster to a parallel computer system comprising an integrated collection of independent “nodes” each of which is a system in its own right capable of independent operation and derived from products developed and marketed for other standalone purposes. Moreover, a commodity cluster is a cluster in which the network(s) as well as the compute nodes are commercial products available in the market for procurement and independent application by organizations (either end users or separate vendors) other than the original equipment manufacturer. A special case of a commodity cluster is the *Beowulf-class PC clusters* that comprise mass-market components both for hardware and software to achieve the best performance to cost and no or limited dependence on any single vendor. Beowulf-class clusters and clusters of workstations were at one time distinct system types but with the blurring or elimination of any meaningful differences between PCs and workstations in capability, the differentiation between the two types of clusters has also largely lost any meaning. This is particularly true with the wide usage of Linux as the base node operating system; a strategy originally pioneered by Beowulf-class clusters.

In our presentation of the Top-500 list, two broad classes of clusters have been represented: “cluster-NOW” and “constellation” systems. Both are commodity cluster systems and are distinguished by the dominant level of parallelism. Although more complex system structures can be devised (e.g. super clusters), commodity clusters usually comprise two levels of parallelism. The first level is the number of nodes connected by the global communications network where a node contains all of the processor and memory resources of the cluster. The second level is the number of processors in each node, usually configured as an SMP (symmetric multiprocessor). If there are more nodes in a commodity cluster than microprocessors in any one of its nodes, then the dominant mode of parallelism is at the first level and this is represented in the category of “cluster-NOW”. If there are more microprocessors in a node than there are nodes in the commodity cluster, then the dominant mode of parallelism is at the second level and this is referred to as a “constellation”. The distinction is not arbitrary as it can have a serious impact on the way the clusters are programmed. For example, it is likely that a cluster-NOW system will be programmed almost exclusively with MPI using a message-passing model, while a constellation is likely to be programmed, at least in part, with OpenMP using a threaded model. Very often a constellation will be space shared with each user getting his/her own node while space sharing a cluster-NOW system would mean allocating some number of

nodes to a particular user. This delineation of cluster types, while justifiable, has some unfortunate properties in practice as will be discussed in the next section.

The critical distinction between this usage of the term “cluster” and that proposed by Bell and Gray is that in this narrower definition, all constituent top level components are commodity with no significant cost to fabrication of the full cluster system other than that of installation and network integration. To field a cluster as part of a market line requires no hardware development investment other than possibly in packaging which may be little more than cosmetic. It is this zero cost to deployment that distinguishes commodity clusters from all other forms of scalable parallel systems and it is this important distinction that we wish to retain in the definition of the term “cluster”. This is in contrast with the Bell and Gray proposal that clusters include besides this narrower set of system types, MPP, DSM, SMP, etc. To accept this broader interpretation would be to sacrifice the crucial benefit that commodity clusters bring and that has triggered a revolution in parallel computing. Commodity clusters and especially Beowulf class systems have provided an exceptional opportunity in performance to cost, invulnerability to specific vendor decisions, flexibility in configuration and expansion, rapid tracking of technology advances, direct use of a wide range of available and often open source software, portability between clusters, and a wide array of choices of component types and characteristics. In addition, commodity clusters have provided scaling between the very small (a few nodes) to the very large (approaching 10K processors). The majority of these benefits have been derived from the specific attribute that the constituent components are off the shelf (COTS) with no specialty parts. This single property has made commodity clusters the dominant training environment for parallel programmers. Yet, the definition proposed by the previous authors would obscure, even eliminate, this seminal quality. We propose to retain it. Therefore, we offer the following definition for the term:

commodity cluster – a parallel computer exclusively comprising commodity computing subsystems and commercial networks such that the computing nodes are developed and employed in standalone configurations for broad (even mass) commercial markets and the networks are dedicated to the private use of the cluster (non-worldly).

3. Taxonomy of HPC Systems

The current architectures are not well characterized by the notion that two architectures exist: Cray-style vector supercomputers, and parallel clusters. The first perhaps only refers to the SX NEC product line, the remaining Cray T90s, and the new Cray Inc. X1 line. What is considered in the Bell/Gray paper to fall under the second category is more accurately represented by a number of distinct classes of parallel computing systems presenting less of a monoculture than might first appear including

- Multicomputers or multiprocessors
- SMP
- DSM
- Distributed memory tightly integrated MPP
- Commodity clusters including (but not restricted to) Beowulf-class systems
- Constellations

These terms in most cases have common meaning within the literature. But the previous authors observe that while constituting the common lexicon, this set of terms is not very useful in providing a general or coherent terminology to consider alternatives or to represent new architectures when they come in to being.

3.1 Distinguishing Properties of Parallel Systems

Supercomputers differ from more broadly commercialized systems in several key ways. These may help determine the seminal parameters or dimensions distinguishing among different classes of systems.

Performance

While the ultimate measure of the effectiveness of execution of a given system is its response time or time to completion for a given application, other imperfect measures attempt to correlate with this fundamental metric to facilitate comparative evaluation of competing systems and strategies. Mips and Gflops are among these. But these represent the dependent variable, or the resulting value derived from the other system structures and characteristics. Except as the primary driver for the highest levels, this need not be part of the classification scheme.

Parallelism

Hardware and software (application) parallelism determines the amount of concurrent work and therefore peak performance that can be achieved. The semantics of parallelism including its granularity and the hardware mechanisms that support parallel execution both determine the amount of parallelism that can be effectively exploited. Classes of architecture may be distinguished by the kinds of parallelism they employ.

Control

The hardware support mechanisms incorporated in the architecture to control the system efficiently can significantly change the operation and performance of the system and distinguish among classes of system. A SIMD computer (e.g. MasPar 2) and a cluster differ dramatically in the hardware support for system

wide parallel execution. The amount of overhead that is in the critical time path for controlling parallel actions is largely dependent on the hardware control mechanisms (or lack thereof).

Latency Management:

The efficiency and scaling of a supercomputer is strongly determined by the impact of the delay for waiting for remote accesses and services. This includes the long distances that messages have to travel, the delays for contention for shared resources such as network bandwidth, and the service times for actions such as assembling and interpreting messages. Latency management includes pipelining including vectors, multithreading, avoidance through explicit locality management and caching, and message driven computing. How a system manages latency is an important distinguishing characteristic.

Name space Distribution:

From an abstract point of view, a system comprises a collection of name spaces and actions that can be performed on such named entities. Shared memory systems versus distributed memory systems are one such division. Names of I/O ports or channels including whether they're local to part of the system or globally accessible is another. Process ID again local or global is another discriminator as is whether processor nodes constitute an explicit name space or are simply a pool of anonymous resources. Therefore, systems may be characterized, at least in part, by their logical name spaces.

Reliance on Commodity:

In recent years, the economics of system implementation has become a dominant theme. The degree to which system architecture exploits commodity components, subsystems, or systems as building blocks for very large structures is considered by some to be an essential attribute determining the likely success or failure of a system concept. Beowulf clusters exclusively comprise commodity components, systems, and networks. The SX-6 employ custom vector processor architectures, motherboards, and networks but use commodity memory chips. Commodity components may be cheaper because of their economy of scale but may lack many functional attributes essential for efficient scalable supercomputing.

3.2 A Framework for Characterizing Parallel Architectures

We propose a naming schema that delineates parallel computing systems according to key dimensions of system attributes rather than the random terms with which we are all familiar. For example, as Bell and Gray make clear, the term "MPP", although used pervasively throughout the history of the Top-500 list,

is confusing, misleading, and provides little specification of system types. The notion of massively parallel processing has been abused, confused, and ironically derived from a different type of system (i.e. SIMD) than that to which it is ordinarily applied. There is a long history of attempts to develop such a methodology going back as far as the classic form by Flynn (i.e. SISD, SIMD, MIMD) but every strategy including this one reflects the critical sensitivities of its time. We suggest four dominant dimensions to characterizing parallel computing systems:

1. clustering,
2. name space,
3. parallelism, and
4. latency/locality management.

Any system can be clustered with like systems to yield a larger ensemble system. This does not mean that all of the attributes of the constituent uniform system is conveyed unmodified to the aggregate system. The important factor here is a synthesis of existing standalone subsystems developed for a different, presumably larger, market and user workload. The alternative, a monolithic system, is not a product of clustering but a structure of highly replicated basic components. There may be other appropriate designators, perhaps that reflect a specific hierarchy. For example, how would we represent a supercluster, i.e. a cluster of clusters?

Name space indicates how far a single name space is shared across a system. While there are many possible name spaces (e.g. variables, process ID, I/O ports, etc.), here we simply consider user variables to illustrate the concept. A distributed name space is one in which the variables of one “node” are not visible directly to another. A shared name space is one in which all variables in all nodes are visible to all other nodes. Yes, by discussing nodes, we are probably over constraining the problem. Cache coherent adds to the shared name space attribute by providing hardware support for management of copies. This illustrates that multiple attributes can be combined (lexically concatenated) to provide a complex descriptive.

The parallelism reflects the kind of forms of action concurrency that is exploited and supported by the hardware architecture. Conventional distributed memory MPPs (old usage) are limited to communicating sequential processes (i.e. message passing) while vector computers, obviously, exploit fine grain vectors and pipelining. There are many other forms as well, even by the processors themselves (e.g. ILP). Note this property field exposes the means by which overhead of managing parallel resources and concurrent tasks is supported and therefore made efficient.

Latency/locality management defines the mechanisms and methods incorporated to tolerate the effects of latency.

Queuing models use a method of a few descriptors separated by slashes to describe a broad range of queue system types. We suggest a similar syntax using four fields. However, we extend such nomenclature to permit multiple designators within any given field to permit a richer description space. These fields and examples (not an exhaustive or even comprehensive set) for each are suggested as follows:

Cluster/Naming/Parallelism/Latency

Clustering := c for commodity cluster or m for monolithic system.

Naming := d for distributed, s for shared, c for cache coherent

Parallelism := t for multithreading, v for vector, c for communicating sequential processes or message passing, s for systolic, w for VLIW, h for producer/consumer, p for parallel processes, etc.

Latency := c for caches, v for vectors, t for multithreaded, m for processor in memory, p for parcel or message driven split-transaction, f for Prefetching, and a for explicit allocation.

Admittedly there may be other designations that should be added to this list. The Earth Simulator would be m/s/v/v, the Tera MTA would be m/s/t/t, the SGI Origin would be m/c/p/c, and Red Storm would be m/d/c/a. The precise codification will have to be worked out by the community in greater detail and accuracy. In this we recognize that better guidelines with greater clarity is needed for such a representation schema to be established or to foster community acceptance. Rather, here we simply suggest a strategy to addressing this challenge. The basic concept permits the system to be described by its attributes rather than a collection of terms not necessarily making up a coordinated lexicon. For example, MPP could mean a distributed memory system, a large shared memory system with or without cache coherence, a large vector system and so on. It covers too many categories and hides too many salient differences to be a useful tool for description. On this we have come to agree with the previous authors.

4. Traversing the trans-Petaflops Performance Regime

It is quite probable that clusters, as Bell and Gray suggest, will have a long lifetime as an architecture principle because there are many workloads that can tolerate their limitations and economy of scale do to mass market forces related to their constituent components. It is quite probable that sometime between 2009 and 2012 that there will be one or more commodity clusters exhibiting a peak performance of 1 Petaflops or greater. However, the evolution of high end computing is not likely to ultimately lead to the convergent cluster architecture Bell and Gray predict but rather to a new class of parallel architectures that

respond to the opportunities and challenges presented by the technology trends that drive it. The memory wall or von Neumann bottleneck among other terms represents the disparity between processor clock rates and memory cycle times as well as the remote access latencies seen system wide. With no changes in structure, the increase in memory densities and processor clock speeds will cause it to take a hundred times as long (measured in processor cycles) to touch every word once within a given memory chip. Without methods of tolerating latency, computations will suffer a thousand cycle critical time delay for accesses to remote memory, perhaps even longer. If no changes occur, the most expensive parts of the system, the I/O bandwidth and memory bandwidth will be configured through the use of ever larger and expensive caches to optimize for the least expensive component, the arithmetic logic unit. Instead, new architectures will be able to exploit between one and two orders of magnitude more ALUs for a given scale system than exists today. While in the short term, Moore's law will apply unabated and commodity components will dominate system designs, eventually, Moore's law will *flat line* due to atomic and quantum effects and conventional components will provide low efficiency (this is already happening) such that little gain will be achieved for larger systems, even for larger sized problems (there will always be embarrassingly parallel problems that are the exceptions to this). Even as problems do get larger, they often increase their sequential time scale as well (more simulation time steps for the same simulated period). At some point, architecture will begin to dominate and the field will ultimately become emancipated for the COTS prison mentality. Its only a matter of time. It is not possible to predict the future but some possibilities are evident even now.

In one sense we agree with Bell and Gray. Clusters (even in our narrower sense of the term) will be an important part of the realm of supercomputing indefinitely. This is because no matter how large and innovative a system may be, one can always assemble a larger one through clustering; it's like turbo charging. But it may be, as has been the subject of research for more than a decade (e.g. Shrimp, SCI), that clustering could be achieved while retaining global name space and with some cross system latency hiding. This is not unreasonable as long as the logical interface to the external world includes access to the address management and translation mechanisms and that the different cluster nodes can subdivide the name space in a mutually exclusive, collectively exhaustive, and logically consistent manner. But to do so means that the processing components have to "know" about each other and their local worlds and this is outside the scope of COTS architectures and enter the realm of custom system designs.

In terms of the conventional balance of bytes per flops, the ratio of 1:1 will be lost. There is about a thousand to one difference in the area of memory required to the area for ALUs required to match it. Instead an entirely different set of metrics and balance requirements will drive future architectures based on bandwidth, overhead time, and latency tolerance. It is possible that the concept

of the processor as we know it will be lost as aggregates of finer grained cell like constructs integrating logic, state, and data transfer are all merged in to one element, highly replicated. Power and reliability through active reconfiguration (graceful degradation) will become as important as optimized throughputs and computation will have to be abstracted (virtualized) with respect to the underlying physical execution medium in order to allow it to adapt on the fly to the constantly shifting organization. System on a Chip, SMP on a Chip, and Processor in Memory will become important elements that bring the memory closer to the logic. At the same time, Logic Intensive Processor Architectures (LIPA) such as Stanford's streaming architecture, UT Austin's Trips architecture, and Cray's Cascade architecture will exploit large internal arrays of ALUs. New technologies may permit systems such as the HTMT architecture to achieve far higher densities of computation as would 3-D packaging, capable of putting half a million chips in a cubic meter. For important applications, special purpose devices may still play a role and field programmable gate arrays might prove of value to make these more accessible and general. High bandwidth optical communication, perhaps even directly connected to the chips will permit bisection bandwidths well beyond many Petabits per second across a system. The choices are so rich and the driving motivation so compelling that at some point COTS based systems will go the way of the mainframe uniprocessors. Supercomputers and desktops are just not the same thing in spite of the fact that they both perform calculations.

5. Centers in the 21st Century

Historically, computing centers invoked images of white lab coats and large front panels behind layers of glass to which mere mortals had but limited and indirect access. They housed the largest processing and storage facilities costing millions of dollars, some of them even ranking as the ultimate high IQ system: the supercomputer. Access was by submitted decks, either physical (punched cards) or virtual (jobs submitted from terminals) for batch processing. As minicomputers, workstations, and ultimately PCs incrementally permeated the computing community, the role of the computer centers evolved and narrowed but retained their critical contribution for large scale computation and users. With the emergence of the cluster and particularly Beowulf-class systems, some contend that the computer center as an institution is at an end. There is no question that commodity clusters and Beowulfs in particular have resulted in local sites obtaining, applying, and maintaining systems with capabilities ordinarily reserved for the pristine conditions of the classic machine room. Often this is accomplished due to minimal up front costs, leveraging extant talent for system support services. For certain contexts such as academic environments and research laboratories this works well for systems of a few dozen to a couple of hundred nodes. But beyond this, resources are usually stressed, sometimes severely, especially when such clusters are shared among a number of potential users and applications. It has been estimated that as few as 50 nodes can

demand a full support person. While this seems a little high, at some level, managing a commodity cluster can become a full time job. But more than cluster nodes are important to high end computing, even using clusters. Mass storage can be an important part of the capability of a system, even one assumed to be dedicated to compute intensive applications. Large archival tertiary storage facilities are becoming an important component of a full service scientific computing environment and can require expert administration. Similarly, networking of cluster systems to the external user base, either within an administrative domain or over the internet adds to the responsibilities of such systems. Upgrades of software, diagnosis and replacement of faulty hardware, and management of accounts, jobs, and user interfaces can all entail significant usage of time and talent in managing the continued operation of a large cluster.

Many labs, groups, and modest organizations can now acquire from within their budget a cluster system capable of substantial performance but not be prepared to engage the resources or fulfill the responsibilities for maintaining and managing the complex computing facility. The Computer Center can be redefined to manage the operation of a large cluster for the owner organization, providing the expertise, infrastructure, and environment necessary for maintaining continued cluster operation. These facilities can be amortized across a diversity of systems limiting the burden on any one system, permitting rapid deployment and high availability, avoiding the learning curve of untrained administrators, and simplifying decommissioning at the end of the system's lifetime. Co-locating a moderate sized commodity cluster under the same administrative facility with other comparable systems enables their synthesis to form superclusters the peak capability of which can be brought to bear of significant user applications by common agreement and sharing protocol. It also makes available large data storage reservoirs within the computer center that might not otherwise be accessible. Thus computer centers complement the strengths of the role and resources of the commodity cluster, extending their price-performance advantage by leveraging investment in the needed administrative facilities. In all likelihood, computer centers will remain if not grow in importance in response to the new trends in cluster computing.

6. Conclusions

Bell and Gray touch on some potential future directions that may drive high end computing through the end of this decade and beyond, thus justifying investment of industry, government sponsors, and the research community. We share much of their view but draw out certain areas to emphasize. Commodity clusters will play an important role for the foreseeable future as there is a significant subset of the user community workload that is well suited to this class of architecture and capable of exploiting mass-market investment in commercial grade systems used as building blocks. Nonetheless, research in this area is still required. Packaging, interconnection networks, and system software, as well as algorithms and fault

tolerance are areas demanding further pursuit. Much progress has been made in packaging of clusters starting with the original tower cases, transitioning to the rack mounted 1u boxes and now the emerging blades technologies providing high density SMP and cluster nodes in very small boxes. Infiniband may open the door to continued advances in cluster networking, leveraging mass-market products, while reducing latency to a few microseconds and increasing bandwidth well beyond 10 Gbps per channel. System software research must establish fully supportive system-wide environments for resource management, administration, and programming including tools for correctness and performance debugging. Such strategies as reflected by Scyld, OSCAR, and ROCS typify progress in this direction but there is still much to do. This includes scaling up these tools and software packages to efficiently handle up to a million processors, far more than anything existing today – yet typical of systems to perform within the trans-Petaflops regime.

But the restriction of devising systems constrained to comprise mostly COTS components precludes innovation critical to achieving high efficiency as well as programmability and reliability. Non-COTS or custom architecture is an important opportunity to be pursued, in spite of the conventional wisdom that dismisses specialty designs as infeasible in today's market climate. A new class of processor architecture intended for a role in highly (multi-million-way) parallel systems must be simple in design, permitting short design cycle as well as easy modeling, simulation, debugging, and compilation. This can be done when permitting multi-threading and adjusting the metric of efficiency to a realistic measure of spatial or cost effectiveness, not ALU utilization. One such strategy was embodied by the HTMT architecture that combined dynamic adaptive resource management mechanisms and methods with advanced device technologies to yield a superior operating point. Only a paper study, HTMT nonetheless demonstrated that an efficient Petaflops scale computer could be implemented within a foot pad of 400 square feet and a power consumption of less than 1 Mega Watt (not including secondary and tertiary storage). HTMTX-class systems may be fabricated from semiconductor devices and still benefit from its processor-in-memory (PIM), multithreading, light weight *parcel* message driven computation, and *percolation* task data prestaging all intended to deliver efficient execution from an easy to program computing model. Other concepts may apply, enabled by custom designed components.

References

1. Gordon Bell and Jim Gray, "High Performance Computing: Crays, Clusters, and Centers. What Next?", *Communications of the ACM*, Vol. #, No. #, Jan. 2002.

2. Erich Strohmaier, Jack J. Dongarra, Hans W. Meuer, and Horst D. Simon, “The Marketplace of High-Performance Computing”, Parallel Computing, 25, (1999) pg. 1517-1544.
3. Horst D. Simon, William T. C. Kramer, and Robert F. Lucas , Building the Teraflops/Petabytes Production Supercomputing Center, Proceedings of EuroPar '99, Toulouse, France, September 1999, Springer Lecture Notes in Computer Science 1999.
4. Thomas Sterling, “Beowulf Cluster Computing with Linux”, MIT Press, 2001.
5. Thomas Sterling, Paul Messina, Paul H. Smith, “Enabling Technologies for Petaflops Computing”, MIT Press, 1995.
6. Greg Pfister, “In Search of Clusters”