Horses for Courses: Optimizing for Workloads

Insight

Gordon Haff
20 October 2009

"Horses for courses" is an old British phrase. It refers to how a horse often performs best at a particular track or in a certain type of weather. Perhaps the horse's running style fits the track and the tightness of its turns, or the horse may just be more comfortable with the track's overall environment for some reason.

Of course, such an observation is hardly unique to horse racing. A movie studio is unlikely to tap a director of intimate period films for its latest special effects-fueled summer blockbuster. And the accounting software package well-suited for the corner store won't begin to meet the needs of a multinational corporation. Indeed, the observation is almost a truism. It applies in so many circumstances that we rightfully treat "one size fits all" claims with skepticism and suspicion.

Server designs are no exception. It was once a commonsensical notion that a wide range of system architectures and approaches was not just the norm but a goodness. Today, by contrast, one commonly sees IT problems of all descriptions presented as nails to be hammered with generic x86 servers. There are many reasons that scale-out x86 boxes running volume operating systems became, and remain, enormously popular. However, many new applications consist of multiple workloads with differing attributes that, in turn, best match specific types of system architectures.

This pairing of task and system, as illustrated primarily by examples from IBM's product portfolio, is the subject of this Illuminata Insight.

Applications Get Smarter

Why the interest in the topic of workload optimization today?

One reason is just that it's something that doesn't get discussed and thought about as much as it should. Server design minutiae, interconnect bandwidths, cache architectures, and the myriad other design details that make a system perform well—or not well—in a particular role were once matters of relatively broad interest within enterprise IT shops.

Today, not so much. There's no disputing that most server purchasing decisions default to a rather narrow set of technologies—namely dual-socket x86 servers connected with Gigabit Ethernet and running either Linux or Windows. This reflects the often attractive costs and the broad ecosystem of developers associated with high volume components. But it also leads to a tacit assumption that this is inherently the best approach for every job—which it isn't.

However, there's also a second reason.

Historically, a given application tended to be characterized by a particular type of workload. For example, an airline reservation system was mostly a big online transaction processing (OLTP) engine. Analysis happened in a different application—typically using a periodic snapshot of the production data. Outside access to this software was relatively limited; for years, travel agents made their living in no small part through their role as gatekeepers to such systems.

Different applications gravitated to different types of servers. For example, big OLTP workloads most naturally fit with "Big Iron," whether mainframes or RISC systems running Unix. Matching application to server was straightforward because software mostly did one thing and the hardware needs to do that one thing were usually well-understood. Core applications with specialized needs often found themselves lumped under the derisive moniker "legacy." The implication was that their particular requirements came about for historical reasons, and didn't really have anything to do with the modern IT world.

Contrast that with many of today's applications, which are composed of many different workloads of different types. IBM calls them "smart applications." You can also think of them as composite applications or an interrelated catalog of services associated with a data repository. An airline reservation system still has an OLTP component; that's the part that takes your credit card number and confirms that you have a seat. (Or overbooks you, at any rate.) This part of the application requires the transactional integrity, security, consistency, and so forth that large database applications all do. It's also relatively predictable; airlines have a fairly good idea of how many bookings they can expect at a given time of year.

However, today an airline reservation system has many other parts as well. There's the outwardly-facing price and availability information that potential customers access from their PCs or, increasingly, their mobile phones, whether directly or through online aggregators like Kayak. Many other Web sites and services are also accessing current flight status. And the airline itself is no longer doing data analysis in what was effectively batch mode. It's making decisions based on its current transactional data store to adjust prices or make other changes.

Without heading down into the architectural weeds of this particular example, suffice it to say that there are many different types of workloads in play here—OLTP, business intelligence (BI), an application tier with distributed content, and multiple network-facing Web services. Modern commerce web sites can be composed of dozens of services; sophisticated sites can have well over one hundred. Such collections of disparate workloads can often benefit from multiple system architectures.

Common Threads

Before getting to how workloads might be matched up with systems, it's worth pointing out that there are common characteristics to the architectures needed to support high-scale applications. The implementation details—and their relative importance—may well differ from one server to another, but certain patterns repeat:

Virtualization. With fewer and fewer exceptions, virtualization serves as part of the foundation of IT infrastructures. It typically comes in for pragmatic and tactical reasons: simplifying test and development processes or running multiple instances of an application environment on a single physical server. However, broader virtualization deployments are also about flexibility and mobility. Virtualization helps to add resources to workloads on-the-fly as their needs and their priority to the business dictates. And it can reallocate those resources when they're no longer required. In short, rewiring and reprovisioning a datacenter happens a lot faster in software than it does in hardware.

High-speed Fabrics. One of the big reasons that people buy large SMP servers is that the optimized connections within the box let processors and memory communicate quickly and at high data rates. These are useful behaviors when an application needs to have a view of an operation from beginning to end, such as if it needs to know whether a database transaction has completed successfully. However, even when a single SMP image is not necessary or even desired, high-performance links—whether within a virtualized server using a virtual network, or between servers using a connection such as InfiniBand—are often important to allow the different parts of composite applications to efficiently exchange data.

Accelerators. General-purpose computers are mostly based on general-purpose processors. On the one hand, mainstream CPUs have gotten faster at such dizzying rates. It's also often proven difficult to couple them with other types of processors that deliver sufficiently dramatic price/performance improvements across a wide enough range of applications to justify the additional complexity that they bring.[1] However, certain specific tasks are important for certain workloads and see substantial benefit from hardware acceleration. Cryptographic operations are one of these. We're also starting to see considerable industry interest in leveraging the computational abilities of graphics processors for workloads in which a set of operations can be applied in parallel to large sets of data.

System and Service Management. Finally, management plays an ever larger role. Part of this is management of the virtualized resources within a homogeneous pool of systems; while virtualization may help reduce the number of physical components, it often increases the number of logical pieces that need to be monitored and controlled. The application as a whole and the services that it delivers both within an organization and to consumers on the outside also have to be coordinated across heterogeneous systems and even the communications networks connecting them to each other and to the rest of the world.

The Many Parts of an Application

Today's systems are typically more general-purpose than in the past, and are correspondingly adaptable to a great many uses. Nonetheless, some fits are more natural and more optimal than others. Proper fit, in turn, leads to better performance, scale, and efficiency than would otherwise be possible.

So is workload optimization about saving IT money? Yes, in part. Better utilizing and simplifying the hardware and software in a datacenter does indeed reduce both capital costs and ongoing operations expenses. However, it's at least as much about improving an organization's business processes and allowing it to offer services that its customers want.

Earlier, we touched on the different components of an airline reservation system. Now, we consider another example that IBM calls "Smart Traffic." As with other such applications, smart traffic is fundamentally about integrating multiple views and ways of using interrelated data into a single application.

Electronic toll collection for autos and public transit are transaction processing systems. They connect directly to consumers' financial instruments such as bank accounts and credit cards, so they require correspondingly high degrees of security and transactional integrity. These activities are also part of more complex business applications. For example, the activity of toll collection may also link to enforcement mechanisms.

---

[1] One of the issues is commonly referred to as Amdahl's Law; it predicts the total performance speedup when only part of a system is made faster. Thus, even if a network protocol accelerator ran infinitely fast, if networking were only a tenth of the total, the overall performance benefit would only be 11 percent.

Traffic flow prediction, on the other hand, is a business analytics workload. It involves various types of models, including neural networks, which look for patterns in a large volume of real-time and historical data. This includes data collected by other components of the traffic application, such as toll collection, but ideally also incorporates information about weather and public events that are known to have a major impact on traffic throughput.

Many of these application components also need a public face. Some of these, such as refilling a transit payment card or checking subway schedules on a PC don't require anything beyond a typical work-a-day e-commerce Web site. However, as we move beyond payment interfaces and relatively static information to real-time data, the requirements increase dramatically—and all the more so because data of all sorts is increasingly "mashed up" with other services, especially into location-based applications running on mobile clients.[2]

Match the Workload to the System

Different styles of applications tend to align with specific types of systems.

Transaction processing is the traditional domain of Big Iron, and it remains so today. Databases and the applications that interact with them have been, and remain, the most vertically scaled of mainstream software. Transactions are typically small, but they change data frequently. And when data changes, the database has to maintain atomicity, consistency, isolation, and durability (the so-called "ACID" properties). This requires a high degree of coordination across the database as a whole—typically implemented using locks, logs, and other mechanisms. The closely coupled components of an SMP server help all this to happen efficiently.

Historically, the mainframe was the canonical OLTP engine. And today, as rejuvenated by modern technologies and open interfaces, IBM's System z is still a common choice for heavy-duty transaction processing.[3] The IBM mainframe is where virtualization was first created. The tight integration of the z/VM operating system and System z hardware mean that it's still the gold standard for virtualizing even the most demanding workloads. In recent iterations, IBM has also added on-chip acceleration for data compression and cryptography, and augmented already impressive reliability and availability features.[4]

Of course, the mainframe isn't the only choice for OLTP. Over time, mainframe-ish virtues such as scale, workload management, and the ability to detect and route around failures have been repurposed or reimagined for other types of systems. Today, this is most evident in high-end Unix servers such as those in IBM's Power Systems lineup, HP's Superdome, or Sun's SPARC Enterprise line.

Business intelligence, on the other hand, is more about processing power. Compared to OLTP, it's also better suited to being distributed across multiple servers; chunks of a BI problem can often be worked in relative isolation. (Although with large datasets, high bandwidth interconnects may still be important to distribute and aggregate the often huge volumes of data being processed.)

Thus, systems used for large-scale BI are often midrange systems with high-performance processors—whether x86, Power, or Itanium. In many respects, system architectures for BI are similar to those used for many high-performance computing (HPC) applications, although typical scale points are lower—at least for now. Thus, clusters rule, and the choice of interconnect, whether GbE or InfiniBand, is primarily a function of the degree to which individual servers need to coordinate with each other.

Network-facing workloads represent still a different set of requirements. Individual transactions are lightweight, as is often the case with OLTP. The difference is that there are a lot of them, they're unpredictable, and they typically don't need to touch a database—at least not

[2] See our Place and the Internet.

[3] See our The Mainframe Reloaded.

[4] See our z10 EC: The Mainframe Bulks Up.

directly.[5] To see the type of server used for these workloads, one need only to look at just about any service provider or Web 2.0 company like Facebook.

Individual servers are cheap and no-frills—dual-socket x86 is the norm—but a high-volume site needs many, so aggregate power and footprint become important. Off-the-shelf rackmount servers are common, although large providers such as Google create their own tailored servers that match a very specific and granular set of requirements. Large system and hardware vendors have begun to introduce products aimed at this segment. iDataPlex is IBM's product that focuses on compute density and reduced power at the rack level. More recently, HP introduced a ProLiant SL line pitched to this space, and Intel unveiled a "micro server" concept.[6] At the microprocessor level, Sun's line of thread-oriented CPUs (UltraSPARC T/"Niagara") likewise targets this type of workload.

Putting It Together

Just about every system vendor has introduced "bundles" from time to time. At their simplest, they're just a collection of hardware (and perhaps services or software) that the vendor either recognized were often ordered together or prescriptively wanted customers to order together. In more sophisticated guises, vendors promote them as "solutions" certified and optimized for a particular use. However, bundles rarely spanned multiple system architectures. Fundamentally, they're oriented towards simplifying or optimizing the ordering and installation process.[7]

Optimizing for the ongoing performance and operation of a system once it is installed is what most distinguishes what IBM is calling "workload optimized systems" from more typical simplification efforts. To illustrate, consider the IBM Smart Analytics Optimizer for DB2 for z/OS v1.1.

As its elongated name suggests, this is a business intelligence product based on an IBM System z. There are some benefits to running BI applications on a mainframe, such as a rich set of audit capabilities and the ability to integrate tightly with a production data store. However, for the reasons discussed earlier, BI is also about cheap processing horsepower, which—whatever its other virtues—is not the System z's forte.

Smart Analytics Optimizer avoids forcing this tradeoff. It introduces a new workload-optimized technology that essentially extends System z workload and service management across other integrated compute resources. This allows System z to do a task that it's good at (coordinating and securing access to a single source of data) while leveraging heterogeneous resources to do the tasks that they're good at (providing inexpensive compute cycles). The DB2 database also comes pre-tuned for BI. IBM estimates that taking this hybrid approach results in typical data warehousing queries executing five to ten times faster than on a System z alone while also reducing the amount of work associated with configuring the database initially.

Conclusion

One of the hallmarks of modern computer systems is their versatility. That the same type of system can model the interactions of subatomic particles and process a credit card transaction might seem unremarkable to many but actually represents a huge shift from the historical norm. However, without suggesting that we're in the midst of a return to the "good old days," we are starting to see a shift towards tailoring system selection and design to specific workloads.

---

5 This class of Web workloads that access dynamic data in a non-transactional manner has given rise to a variety of mechanisms to service them in a way that minimizes relatively expensive accesses to back-end SQL databases. See our The New Databases.

6 See our Microservers: Blades Rebooted and HP ProLiant SL: New Times, New Scales.

7 "Appliances" are arguably a cut at simplifying both installation and operations but they generally target specialized workloads and, in any case, haven't proven to be a popular approach outside of niches such as load balancing and perimeter security. See our The End of Cobalt and the Appliance Era that Never Was.

One example is large cloud computing companies such as Google, which customize servers with just the right processor, networking components, and sheet metal for a particular use. Another is IBM's focus on a workload optimized systems approach by which it matches a specific type of system to a specific workload within an application.

The culprit, if you want to call it that, is an unquenchable thirst for computing. We all want the world's information and the insight we can wring from it at our fingertips. We want complex processes of the physical world to adapt based on real-time data. All this interconnection, instrumentation, and quest for insight is generating IT demands at a scale that "good enough" optimization around high volume, generic parts starts to look less attractive than it once did.

DIL03009USEN-00