A Fault-Tolerant Parallel Scheduler for BLAST

Kevin Bealer, George Coulouris, Ilya Dondoshansky, Thomas L. Madden, Yuri Merezhuk, Yan Raytselis
National Center for Biotechnology Information
Bethesda, MD 20894
{bealer, coulouri, dondosha, madden, merezhuk, raytseli}@ncbi.nlm.nih.gov

Introduction

The National Center for Biotechnology Information (NCBI) provides a public service that allows users to search molecular sequence databases with the Basic Local Alignment Search Tool (BLAST). This service requires high availability, high performance, and must handle thousands of concurrent requests fairly. Because the sizes of molecular sequence databases now exceed the memory capacities of commodity servers, it is necessary to spread searches across multiple worker nodes to achieve high performance. Furthermore, to reduce the number of passes through the sequence data, it is desirable to combine multiple searches against the same sequence database. We describe the design and implementation of the BLAST scheduling system in use at the NCBI. This system has been in nonstop operation since the fall of 2003 and handles over 100,000 search requests daily.

Component overview

The system is logically composed of three tiers that communicate via a relational database. Components in the interface tier are responsible for communicating with end-user agents such as Web browsers, batch command-line clients, or interactive GUI clients. These components insert new requests into and fetch results from the database. The second tier contains the aggregation, parallelization, and scheduling components. These components maintain queues of searches ready to be processed and results ready to be merged. The third tier is the pool of worker nodes. The worker nodes request work, read sequence databases from network-attached storage (NAS), and write partial results to local storage.

Infrastructure

The relational database uses 2U dual-CPU commodity servers running Microsoft Windows 2000 and Microsoft SQL Server 2000; all other components use 1U dual-CPU servers running Linux. Sequence data are stored on highly available NAS appliances. All interconnect networks are commodity fast or gigabit ethernet.

Fault tolerance

In the event of a worker node failure, searches are simply rescheduled on another available node; searches that fail repeatedly are quarantined to avoid wasting resources. The relational database servers are replicated in a primary/backup arrangement. Backup queuing components come online automatically via an election-based failover protocol. A software loadbalancer routes requests to redundant interface components.

Sequence data updates

To provide up-to-date results, sequence databases are regenerated daily from GenBank. Presently, these databases occupy 175 GiB, 20 GiB of which are cached locally on the worker nodes; when new databases become available, they are distributed to the worker nodes via multicast. To provide nonstop operation, two copies of the sequence data are maintained. The worker nodes switch to the new data at a fixed pace to prevent NAS bottlenecks.

Workflow overview

An incoming search request is first ranked according to a fairness algorithm that assigns relative priorities to different classes of jobs. The request is then assigned an identifier and inserted into the database. Next, a scheduling host fetches requests from the database and combines them if possible. The request is then split into a number of pieces, and these partial searches are scheduled onto processors. When all of the partial searches are complete, the partial results are retrieved, merged, and the complete results are stored in the database. Finally, the alignment is computed and stored in the database. The user may then request the search results in a variety of formats, such as HTML, XML, or ASN.1.

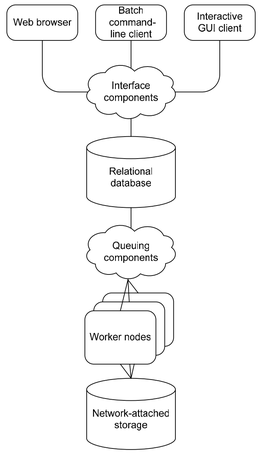

Figure 1. Component overview.

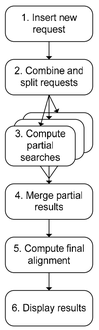

Figure 2. Workflow overview.

Features

- Parallelism adapts to load
- Requests are combined if possible
- No single points of failure
- Sequence databases are updated transparently daily and cached locally

Development timeline

- 4Q 2001—design
- 2Q 2002—prototyping
- 3Q 2003—deployment

Future work

- Multiple database servers
- Disaster tolerance
- Heterogeneous scheduling
- Split queries as well as databases for large searches

BLAST facts

- Over 100,000 searches per day
- Currently running on 280 processors
- Nonstop operation since September 2003
- > 90% memory cache hit rate
- Most searches complete in under 40 seconds

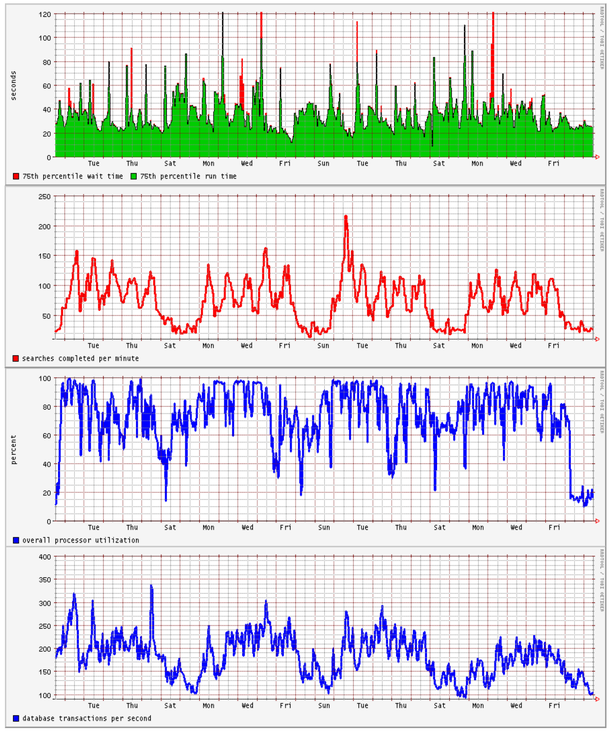

Figure 3. Performance.

Top to bottom: 75th percentile run and wait time in wallclock seconds, searches completed per minute, fraction of overall processor utilization, database transactions per second.

References

1. Altschul SF, Madden TL, Schäffer AA, Zhang J, Zhang Z, Miller W, Lipman DJ. Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. Nucleic Acids Res 1997 25:3389-3402.

2. Kegel, D. The C10K problem. Available from: http://www.kegel.com/c10k.html

3. Schmidt, DC. Patterns for concurrent, parallel, and distributed systems. Available from: http://www.cs.wustl.edu/~schmidt/patterns-ace.html