LUCATA PROGRAMMING OPTIONS

Lucata Pathfinder is a next-generation computing platform that dramatically improves processing speed and scalability for sparse Big Data. Adopters can attain deeper graph analytics insights and more accurate and faster artificial intelligence model training than ever before possible by processing massive datasets without database sharding or data pruning. The Lucata programming environment empowers the Lucata Migrating Thread technology, enabling commercial and internally developed applications to leverage the full power of the Lucata platform.

PATHFINDER USAGE OPTIONS

The Lucata Pathfinder hardware platform can power open source, commercial, or custom-developed graph databases using one of three approaches:

- Library calls to Lucata'a optimized algorithm library that runs natively on Lucata, including Breadth-First Search (BFS), PageRank, Connected Component, Triangle Count, and K-truss Subgraph
- API calls for GraphBLAS queries supported on the Lucata platform, such as Sparse Matrix Multiply. The public domain LAGRAPH library provides an extensive collection of functions constructed with this approach.
- Running RedisGraph queries on Lucata using the optimized data loader and RedisGraph application code which have been ported to Lucata and optimized for use with Lucata Migratory Thread technology

The first two options provide alternatives for organizations that will use Lucata with their own internally developed graph applications or will modify an open source or commercial graph application to leverage the power of Lucata. The third option allows existing Redis or RedisGraph users to easily leverage Lucata to run high performance queries on massive graphs using RedisGraph.

PATHFINDER PROGRAMMING: QUICK, EASY AND EFFICIENT

As a developer, you can quickly build and test code. You can use IDEs and debuggers with libraries based on Intel x86 or use RedisGraph's graphical interface or the Redis command-line interface. The toolchain and libraries for the Pathfinder system support program development in C, C++, and Cilk. The emu-cc program in the OpenCilk toolchain manages the compilation of programs for the Pathfinder system. The OpenCilk toolchain, an open-source cross-compiler managed by the Massachusetts Institute of Technology (cilk.mit.edu), runs on x86 systems and generates executable programs for the Pathfinder system.

Pathfinder can spawn millions of parallel threads, significantly reducing runtime for enabled applications. You can write parallel programs with ease using Cilk. You only need to know a few additional simple commands and Pathfinder will handle the complexities behind the scenes.

CONTACT US

Contact Lucata now to learn more about the Pathfinder-S for high performance graph analytics.
Please email us at **info@lucata.com** or call us at **646 661-5252**.

Examples of Cilk Parallel Programming Keywords	
Attribute	Identifies functionality
cilk spawn	Indicates that the function can run in parallel with the caller. Typically, a new thread is spawned to execute the function.
cilk sync	Causes the parent thread to synchronize with its children by waiting for all of the child threads it created to return
cilk for	Replaces the traditional FOR loop so that loop iterations can execute in parallel.

To reduce the overhead involved in managing threads, the Pathfinder architecture supports thread spawning and thread synchronization directly in hardware. It does not use the Cilk runtime software.

MEMORY ACCESS: SIMPLIFYING THE PROGRAMMING PROCESS

The Pathfinder architecture implements a Partitioned Global Address Space (PGAS) that simplifies programming. Each memory location has a unique address that is visible anywhere in the system. Furthermore, the range of addresses need not be contiguous and the memory may be distributed across various partitions. This memory technology eliminates the need for data pruning and database sharding and reduces programming complexity by enabling developers to process and analyze data as one monolithic dataset.

PATHFINDER-S SOFTWARE SPECIFICATIONS

Operating System	Red Hat Enterprise Linux 8.x
System Compiler	Lucata C / C++ / OpenCilk LLVM ~~6~~ compiler
Optimizing Languages	Cilk, C, C++, CilkPlus
Front-end Languages	Python 2/3, Java, SQL, any code running on x86