Real-time multi-core PDE-solvers in LabVIEW

Shawn McCaslin, shawn.mccaslin@ni.com
Michael Cerna, michael.cerna@ni.com
Michael Chen, michael.chen@ni.com
Nanxiong Zhang, nanxiong.zhang@ni.com
Bin Wang, bin.wang@ni.com
Lothar Wenzel, lothar.wenzel@ni.com
National Instruments, Austin, TX, U.S.A.

Abstract

The availability of low-cost multi-core systems enables LabVIEW to combine sophisticated data acquisition systems with demanding numerical tasks. In a typical scenario information about a system is based on direct or derived measurements and the acquired data is used to solve linear or even nonlinear elliptic partial differential equations. The results generated by the PDE-solver might be used as feedback to the running process. Such a system can be very demanding from a realtime standpoint and might require loop-times in the 1 ms range.

To comply with such specifications, multi-core architectures and other techniques such as FPGA-based components and high-speed networking are supported by LabVIEW. We provide benchmarks for specific nonlinear elliptic PDE solvers based on 4-, 8- and 16-core machines using standard quad-core processors where multi-board deployments require fast networking. We also report multi-core performance numbers for more elementary operations such as 1D/2D FFT, DST/DCT (sine/cosine transform) and matrix operations. It turns out that carefully designed systems can come reasonably close to ideal speed-ups.

Data Acquisition and Number Crunching

Control theory is a well-established field with applications that can be found virtually everywhere in engineering and science. Over the last 80 years the mathematical background was extensively developed and covers a broad range of topics. Still, the vast majority of these applications relies on ordinary differential equations. One of the most significant contributions of control theory is building a bridge between ODEs and algebraic concepts such as transfer functions and state-space representations. On the other hand, and just to focus on one concrete example, the temperature control of an oven can not always be implemented by a straightforward one sensor (thermocouple) and one actuator (heater) system.

On the contrary, many of the more demanding control examples require a deeper understanding of higher dimensional data structures. It might make perfect sense to reconstruct the entire temperature field of an oven to control the behavior in a more appropriate manner. Another example is the reconstruction of the temperature field of a wafer based on some well-distributed sensor with the final goal to avoid potentially harmful temperature spikes.

Sometimes, the input-output structure might be simple as in the case of an inverted pendulum where two inputs and one output are necessary. But if the control of an inverted pendulum is based on image processing, very high-dimensional data must be considered before the 2-inputs and 1-output structure emerges.

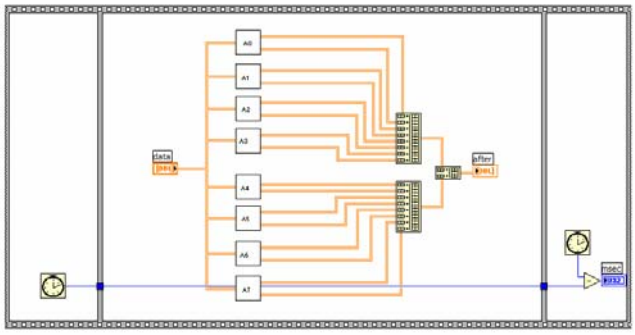

Figure 1: A specific matrix operation is distributed in such a way that 8 cores can work in parallel. The same code would also execute well in LabVIEW using 4, 2 or even 1 core though for low-core systems the code would be sub-optimal because of unnecessary splitting and recombination parts.

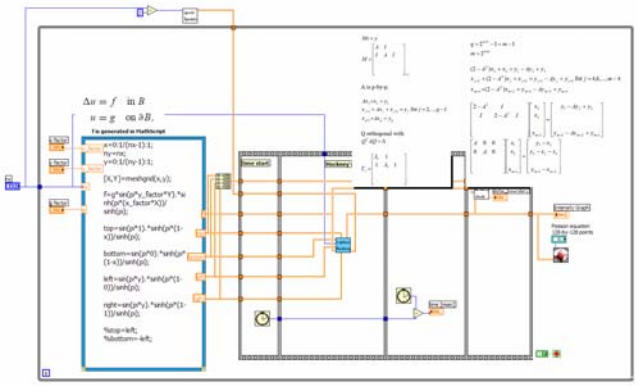

Figure 2: LabVIEW combi nes several aspects of modern programming. The PDE-solver is executed in parallel on n-core systems (center). The PDE problem itself is defined with the aid of a textual language (left). The diagram is annotated with formulas to increase readability.

National Instruments Corporation, 11500 N Mopac Expwy
Austin, TX 78759-3504

The paper focuses on another demanding application - the control of plasma in a nuclear fusion reactor (tokamak). Here the requirements are truly remarkable ([1]). In a still very simplified specification the system has to solve a nonlinear elliptic partial differential equation in 1 ms where the spatial resolution is in the range of 128-by-128 discrete points (see Fig. 3). Moreover, the boundary conditions must be generated on-the-fly by using Green's functions based on sets of specific measurements. The results of the PDE solver are used as feedback for the controller.

Such specifications are in the reach of modern n-core systems in conjunction with integrated data acquisition components. We report performance numbers for 4-, 8- and 16-core PDE solvers using Lab VIEW running under Windows Server 2003 on a DELL PowerEdge 1955 blade server. We also report n-core benchmarks for the following related problems:

- matrix-by-vector multiplication
- matrix-by-matrix multiplication
- 2D FFT, 2D DST (sine transform), 2D DCT (cosine transform)
- highly decoupled multi-channel FFT

Fig. 1 demonstrates how a specific matrix operation can be forced to execute on an 8-core system. The same code would also run on quad- or dual core machines. It is also possible to add a case-structure that contains code for all expected core-situations. The valid branch is chosen at runtime. In Fig. 2 some other components are shown that connect n-core computation with more commonly used programming paradigms.

For PDE solvers we report excellent speed-ups for 8-16 cores and expect acceptable speed-ups for the range of up to 80 cores. Larger numbers of cores might require a shift to new classes of algorithms that decompose the problem at a higher level ([2]). In general, one of the challenges for higher numbers of cores is the fact that inter-board communication can become a serious bottleneck which can be overcome by using very fast and deterministic networks. We report benchmarks for these processes as well.

A convenient method to communicate between two or more 8-core systems is the use of Shared Variables. Shared Variables are implemented as part of a very high abstraction layer. This is a definitive advantage when large systems consisting of separated layers of numerical and DSP routines have to communicate their results from stage to stage. Such variables and their properties are defined globally and can be written by one module (including automatic timestamp) and consumed by one or several other modules that are located on another blade and on a specific core.

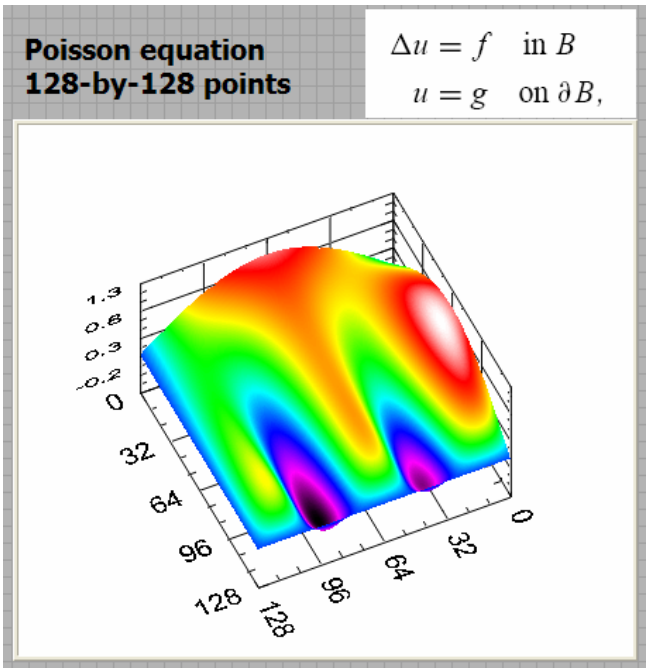

Figure 3: To control a tokamak nuclear fusion reactor - among many other things - problems like the depicted one must be solved in much faster than 1 ms. Modern n-core machines can deliver such performance numbers and scale very well with problem size.

References.

[1] J. Huang and Jon Menard, "Development Of An Auto-Convergent Free-Boundary Axisymmetric Equilibrium Solver" *U.S. Department of Energy Journal of Undergraduate Research*, 2006.

[2] E. Gallopoulos and Y. Saad "A Parallel Block Cyclic Reduction Algorithm For The Fast Solution Of Elliptic Equations" *Paralel Computing*, 10 (1989), p. 143-159.