Randomized Computation Networks

R. D. Hangartner
Tera Computer Corp., Seattle, WA.

For the past 50 years the dominant computing paradigm has been sequential, deterministic, polynomial time, von Neumann programming. There have been attempts to challenge this paradigm, e.g. functional, nondeterministic, and parallel, but none have been overwhelmingly successful. The recent history of theoretical computer science, however, suggests a quite striking alternative. While trying to understand the P=NP question, theorists have extensively studied randomized algorithms and the corresponding complexity class BPP (bounded probability polynomial time). Although randomized algorithms have been insufficient to separate P from NP, they have been shown to be the fastest and most elegant techniques for a large number of problems from graph theory, number theory, and combinatorics. (The crux of these methods is that for most pairs of problems and algorithms there is a small subset of inputs which will force the algorithm to take an unreasonably long time. By randomizing one can almost always make sure that algorithm will NOT take a long time.) Despite these results, randomized machines have not been seriously investigated --- clearly the idea of a randomized machine challenges traditional computer engineering precepts. This talk describes early work on a practical asynchronous architecture for solving instances of the SAT problem, dubbed a Randomized Computation Network (RCN), which harnesses intrinsic circuit noise to efficiently realize massively parallel randomized computations.

The RCN is an architecture which effectively converts the main memory or mass storage space of any standard computer architecture into a co-processor for BPP algorithms. In the RCN, key operations on a suitable generic data structure are realized by hardware primitives that operate asynchronously and independently of the CPU. To effect a computation, the CPU writes the contents of the data structures just as it would write them to conventional main memory. Rather then perform the actual computation, the CPU simply reads the results of the computation from the same memory space after a suitable delay. To prove the formal soundness of this approach, the generic data structure in the version of the RCN described here encodes instances of the Boolean Satisfiability problem (SAT) represented in Conjunctive Normal Form (CNF) since the circuits for dealing with other data structures are reducible to those for SAT instances.

At the circuit level, the RCN consists of a standard SRAM or DRAM array coupled to a plane of crosspoint switches, much like the routing resources in a FPGA. Loading a SAT instance into the RAM configures the switches into an OR-AND network. The inputs to this network are driven by a set of random bit generators similar in design to sense amps but optimized to be sensitive to intrinsic circuit noise. The single output of the network controls these random bit generators. The entire circuit cycles asynchronously, searching for a satisfying assignment by repetitively driving the bit generators into an unresolved state and then allowing them to resolve under the influence of intrinsic circuit noise to a set of random bit values. If a satisfying assignment is found within an appropriate number of attempts (generally constant or linear in the size of the SAT instant), the cycling action halts; otherwise the instance is deemed to be unsatisfiable. An auxiliary memory is used to store the sequence of intermediate bit choices for possible use as approximations in the latter case. Operating in this manner, the RCN solves, with arbitrarily boundable error, those problems which are polynomially reducible to SAT instances and which have satisfying assignments proportional in number to the size of the SAT instance. The proposed RCN arguably is an optimal architecture for realizing the largest practically computable class (BPP) of randomized algorithms in terms of several reasonable criteria: 1) The asynchronous design yields a computation rate, measured in logical-operations-per-second, near the theoretical maximum for the silicon process in which it is implemented. 2) By serving as a main memory replacement, the architecture is an easily implemented hardware extension of any standard computing technology and is ideally suited for realization in 0.25 and deeper sub-micron processes with embedded DRAM. 3) Finally, while the choice of SAT instances as the generic data structure in this first version was motivated by formal considerations, alternative structures more suited to abstract data structures such as Binary Decision Diagrams (BDDs) are possible. Since BDDs are at the heart of current and next generation EDA tools, the RCN may be an efficient tool for attacking certain problems such as formal theorem proving whose complexity would otherwise preclude efficient solution on q deterministic architectures.