SHORT DESCRIPTION OF LOSP PROJECT

William Bricken
February 1987

The purpose of the Losp Project is to implement boundary mathematics in a parallel inference engine. Boundary mathematics is an innovative computational technique for manipulating tokens that represent boundaries (such as parentheses and brackets). By placing the semantics of logic onto a boundary notation, we have created a system that is both representationally elegant and computationally efficient. Boundary notation is more concise, requires less pattern-matching to perform inference, and is in general more powerful than traditional approaches to logical deduction. Since the boundary formalism is inherently two-dimensional, it provides a visual language that incorporates logical parallelism by eliminating the conventions of linear notations. The Losp decision procedure converges rapidly and is easily implemented and understood.

We are implementing two versions of boundary mathematics applied to logical deduction. The linear version is written in Pure LISP; it maintains the Boolean structure of a formula in the nesting and concatenation of lists bounded by parentheses. Deduction is accomplished by pattern matching which results in erasure of list structure. Although benchmarking the performance of the linear engine for propositional calculus is just beginning, we have demonstrated a five-fold increase in deductive speed, compared to the Boyer-Moore Theorem Prover. In addition, the algorithm is more powerful, returning contingent expressions which can be used to construct specific counterexamples.

The Losp parallel deductive engine currently under development treats boundaries and terms as system-nodes in a distributed architecture. Communication links are defined by nesting of boundaries and by function-argument dependencies. Problems originally expressed in first-order logic are parsed into a distributed network representation. Deduction is accomplished by message passing which results in the erasure of nodes and links. Each node determines whether or not to alter local network structure based on local information. The network that remains after local parallel reduction represents the solution. The strong parallelism of this engine permits goal-free minimization of premises and obviates the need for forward and backward chaining by distributing deduction opportunistically across the network representation. We will use this system for rule-base optimization, compilation, and partitioning.

In addition to inference, the Losp formalism has been used experimentally for Boolean minimization and for compiling and optimizing the control structure of functional code (Pure LISP) and declarative code (Prolog). We have designed boundary deduction systems for database query management and for planning.