Research:

- *Automated Deduction:* resolution decision procedures, decidable classes of first-order logic, automated model building, subsumption algorithms.
- *Proof Theory:* analysis of cut elimination in classical first-order logic, cut-elimination by projection, cut-elimination by resolution, cut-normal forms and proof complexity, complexity of first-order inference. Automated transformation and analysis of mathematical proofs by CERES (cut-elimination by resolution).
- *Recursion Theory* (until 1984): enumeration of subrecursive classes, complexity of index sets, self-reproducing automata and self-control.

A general remark on automated deduction:

We believe that substantial improvements in automated deduction can only be achieved by further *foundational research* and thorough *mathematical analysis*. Think e.g. about first-order theorem proving without Robinson's unification principle (but instead with the 150th improvement of Davis-Putnam combined with ground saturation). One of the most serious defects of current automated deduction systems is their weakness in using *lemmas*. Indeed there is no intelligent way of proving and reasoning without using existing knowledge. The highly sophisticated analytic and quasi-analytic inference engines in theorem proving are surely powerful and useful, but only as an infrastructure for inference. We still lack some profound understanding of the role and the use of lemmas within proofs, and thus of the *reuse of mathematical knowledge*. Without such a reuse, we can speak well about intelligent implementations of calculi, but not about intelligent *reasoning*. Concentrating only on improvements of current theorem provers will lead to smaller and smaller steps of progress, potentially leading to a halt.

There is one more basic point which seems to be crucial, in particular concerning applications: methods which are general are bad on specific domains. Mathematicians are typically more pragmatic than logicians (and also than computer scientists) in choosing their methods and algorithms corresponding to the specific structure of the problem. Clearly the most general methods are most ineffective in detail, the most powerful ones are most restricted in their range of applicability. So how specific should theorem proving algorithms be? Writing a new theorem prover for every mathematical theory

(Skylla) or running the same (slow) method on all theories (Charybdis)? This is clearly one of the central questions in automated deduction at all, and can only be answered when we develop something like a *culture of deduction*, i.e. a way to use, interpret proofs and to search for them. This brings us back to the very root of the problem: the conceptual understanding of inference in general.