The general idea that computations and proofs have much more in common than it may appear from the first sight is being developed in at least three sister communities descending from the classical mathematical logic. Complexity Theory contributed to that purpose notions like *interactive proofs*, *probabilistically checkable proofs* and *transparent proofs* challenging such fundamental issues as rigorousness and verifiability. Feasible Proof Theory suggested that even classical proofs *have their own intrinsic complexity*, and it is extremely closely related to the ordinary complexity of algorithms. Finally, the "LICS community" analyses proofs in formal systems whose *semantics* is intended to capture the process of computation happening in the real software.

In several last years we started to see some renewed communication on this topic between the fields once united in the framework of mathematical logic. My prediction is that in upcoming years these relations between computations and proofs will receive even closer attention on the side of Complexity Theory (other parties involved already do it on the professional basis). Our community certainly has something here both to say (e.g., how to prove anything about the complexity of proofs? Or how to prove efficiently the correctness of programs, rigorously defining first what it means?) and to ask (e.g., what makes our own fundamental problems so difficult to solve?)

Coming to another part of the question, "Where *should* Complexity Theory go?", there's a lot of effort now to try to increase the impact of our field on key application areas. But in my opinion it is not quite clear whether Complexity Theory should go anywhere at all or it would be more useful staying where it is. The mission of this discipline is to provide a bridge for the traffic of ideas and concepts (with a handful of exceptions, not the results themselves!) between pure mathematics and Computer Science. Try to pull it to one side (say, for the purpose of making a junction instead), and firstly you will no longer have anything to cross the river upon, and secondly you may discover that the bridge's remnants are less useful on the land than expected as the construction was designed for different purposes. However, in my opinion that part of semi-applied research in Complexity Theory which *develops according to the* **internal** *logic of our field* has to be strongly encouraged (continuing the analogy with the bridge, it is simply our duty to provide as convenient access to the traffic across it as we can manage). The work on efficient program verification mentioned above in quite a different context is one good example of this.