Zeitschrift: L'Enseignement Mathématique

Herausgeber: Commission Internationale de l'Enseignement Mathématique

Band: 27 (1981)

Heft: 1-2: L'ENSEIGNEMENT MATHÉMATIQUE

Artikel: TOWARDS A COMPLEXITY THEORY OF SYNCHRONOUS PARALLEL COMPUTATION

Autor: Cook, Stephen A.

Kapitel: 8. Open Questions

DOI: https://doi.org/10.5169/seals-51742

8. Open Questions

Among the basic open questions in computational complexity are the problems of finding lower bounds for various resources for any simple interesting problem. In particular, for sequential complexity, we don't have any nonlinear time lower bounds nor any nonlogarithmic (i.e. $\omega(\log n)$) space lower bounds on any natural problem in the class NP. For parallel complexity, the same state of ignorance applies to nonlinear circuit size, and nonlogarithmic depth and hardware. Theorems 2.2 and 2.4 indicate that a nonlogarithmic lower bound on circuit depth may be weaker (and therefore easier to obtain) than such a bound on space, so the depth question deserves more attention. (There are already results which show the depth complexity of some simple problems cannot be $\log_2 n + O(1)$: see Neciporuk [N1] and Hodes and Specker [HS2].)

For simultaneous resource bounds, the situation is almost as wide open, although Borodin and Cook [BC] have recently shown that sorting cannot be done *simultaneously* in linear time and logarithmic space. It would be interesting to get similar tradeoff results for other resource pairs, such as size versus depth and aggregate time versus hardware. Another problem is whether there exists any set whose minimum time complexity is at least, say, the square of its minimum space complexity (assuming the latter is at least $\Omega(n)$). Similarly for uniform size versus uniform depth and aggregate time versus hardware size. (We do know by Lupanov's result [S3] that most sets have (nonuniform) size exponential in depth.

Finally, the questions concerning SC and NC mentioned in section 7 are worth emphasizing. In particular, it would be nice to know whether one class is included in the other, and whether they are proper subsets of their (common) intersection class.

Acknowledgement. While forming my ideas on parallel computation, and in particular while writing this manuscript I had frequent conversations with my colleagues Allan Borodin and Patrick Dymond, and their ideas as much as mine have contributed to whatever insights appear here. My thanks also to Nicholas Pippenger for some very helpful conversations.