(Really brief) thoughts on the feasibility of:

Deterministic, Compile-Time Quality of Result Computation for Approximate Programs

Kendall Stewart
Portland State University

1. Probabilistic / empirical QoR determination at compile time may be infeasible if training data is scarce and hard to compute, and may be unacceptable if the cost of a rare failure from the tail of the quality distribution is incredibly high. These problems may become worse if approximate systems are composed.

2. However, statically computing non-empirical error bounds for approximations of arbitrary code is also infeasible, since the code may express a problem known not to be in APX unless P=NP, such as MAX CLIQUE (c.f. J. Håstad 1996).

3. There have been efforts to provide syntactic characterizations of optimization problems that must admit known-error, efficient approximation schemes (c.f. MAX Φ, A. Malmström 1997; PLANAR MAX-*W*[*h*], J. Chen, et al. 2004), however my search has yet to find work on turning these results into front-end programming language constructs. My imagination leads me to a strongly-typed functional language.

4. All of this could result in a higher-order declarative language for producing efficient, known-quality, deterministic approximation algorithms for certain types of difficult problems. A very open question, as far as I can tell, is whether or not these approximation algorithms would amenable for transformation into approximate hardware.