DMR

DMR is a programming language with a focus on automatic or simple distributed computation.

by Matthew Yaspan and Maxwell Bernstein

Goals

DMR aims to speed up computation by fanning out calls like map, filter, foldl, etc to a cluster of threads or even other computers. It does data dependency resolution to ensure that there are no race conditions in the computation.

A stretch goal is to create a parser/lexer for DMR, so the user does not have to manually input an AST. Perhaps even a metacircular evaluator, so that DMR is itself distributed.

Planning Milestones

Create foundation for threaded computation

- Create a protocol in which work computers and main computer communicate
- implement load balancer and reliability features using Erlang
- Create a test environment in which we can implement protocol as well as test different schemes of distributing work

Bridge gap between threaded and distributed computation

- Ensure that there is no difference (except in speed) between sending work to a thread and sending work to another computer entirely
- Finalize thread pool API

Write data-dependency resolver

- Determine if/how it is possible to statically analyze an AST and assign â€œtypesâ€ that allow the program to distribute work
- Create an algorithm that detects data-dependencies and ensures there are no race conditions

Implement `map`, `reduce` in DMR context

- Ensure `map` can distribute work across computers atomically
    - DONE (for local threads)
- Determine if it is possible for `reduce` to be distributed efficiently
- Implement other functions in terms of `map`