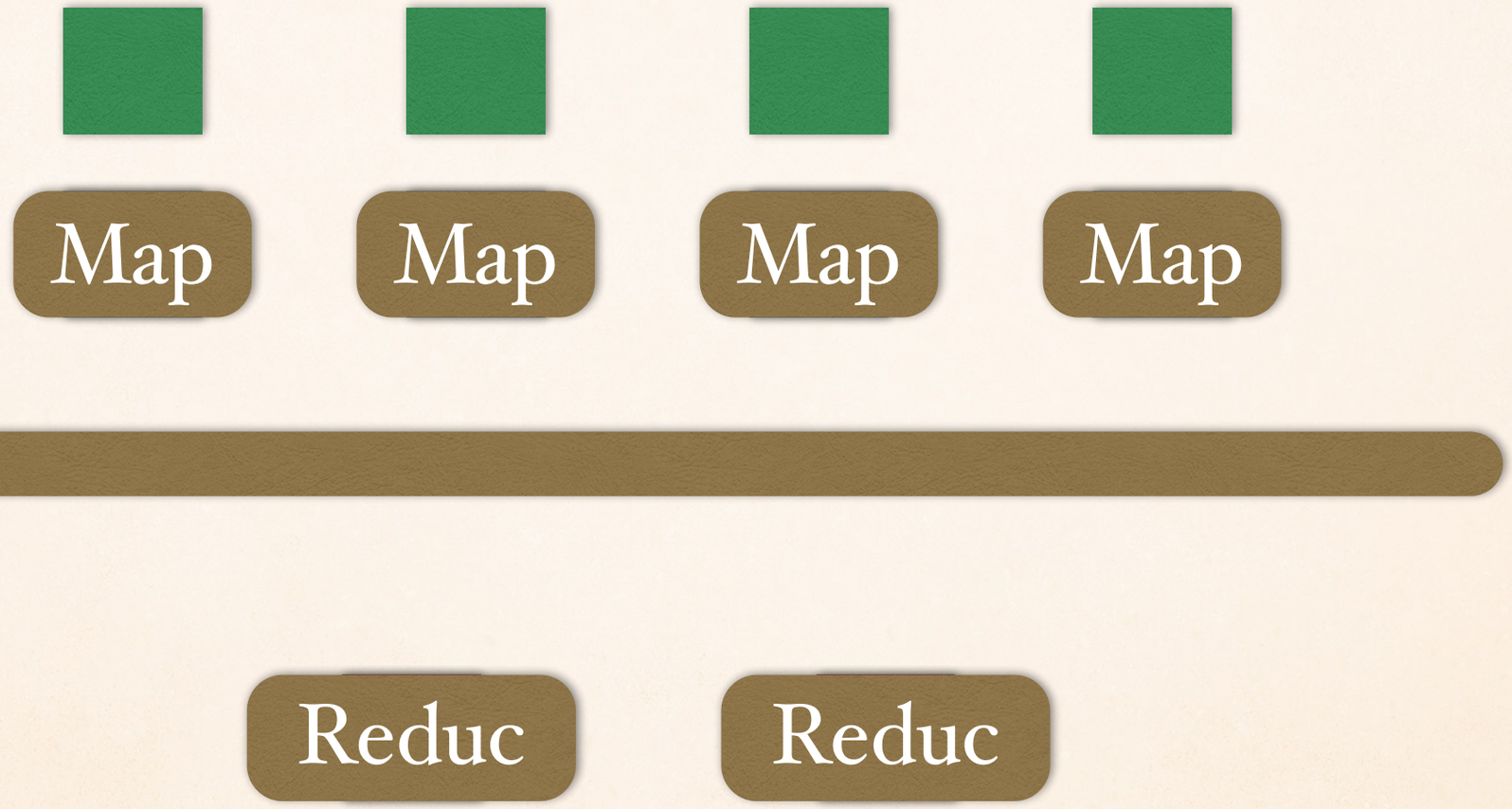
MAP-REDUCE

Distributed Computing Made Easy

KEY IDEAS

- Map(): Apply function f to each item of input list of key-value pairs and emit intermediate key-value pairs
- Reduce(): Aggregate intermediate key-value pairs and emit a smaller (usually) list of values.
- Automatic parallelization and distribution
- Fault tolerance
- Status and monitoring tools

MAP-REDUCE



PROGRAMMERS VIEW

- Expressibility: real world problems can be expressed as MR
- Clean abstraction: only need to provide map() and reduce() functions
- Hides messy details: parallelization, fault tolerance, locality optimizations, load balancing
- Scalability: batch process vast amounts of data on large clusters of machines
- Focus on problem

IMPACT

- Functional programming paradigms to large scale computations
- Moving beyond RPC, threading, shared file access
- Large scale computations (order petabyte)
- Tolerate machine failures gracefully
- Simple execution engine

IMPACT

- Open source implementation (Hadoop)
- Query language on MapReduce (Pig Latin)
- Data warehousing (Hive)
- Comparison with parallel DBMSs (Pavlo et al)
- DBMS on top of Hadoop (HadoopDB)
- Other libraries/platforms - EC2, Cloud9

THANKS

Necessity is Mother of Invention