commodity clusters
allows scalable computing to achieve data-parallel scalability for big data
→ cost effective

+

advances in distributed file systems to move computation to data

→ scalable big data analytics

→ how to ~~take~~ ~~advantage~~ of this advances

→ **programming models**

it is
- abstraction / existing machinery / infra structure
  - code → low level (machine language)
  - code → high level (Java)
- set of abstract runtime libraries and programing languages → form a → model of computation

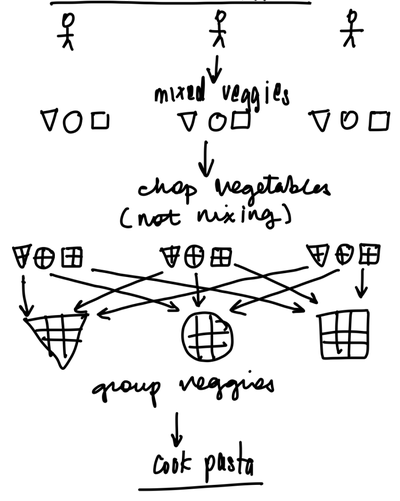
(summary)
programming model = abstractions
runtime libraries + programming languages

FOR BIG DATA : programmability on top of distributed file systems (DFS)
- computer programs that work efficiently
- cope with issues

requirements
1. support big data operations ← split volumes of data, access data fast, distribute computations to nodes — (scheduling many parallel tasks at once)
2. handle fault tolerance ← replicate data partitions, recover files when needed
3. enable adding more racks — (new resources), called scaling out if needed — for ← more or faster data, not losing performance
4. optimized for specific data types
   - types: document, table, key-value, graph, multimedia, stream
   - optimized for at least one type

example (of big data modeling)
- what: cook pasta with tomato sauce
- time: in an hour (1hr)

USE PARALLELIZATION



model: Map Reduce
- big data programming model
- many implementations → Hadoop (framework) implementation
- supports all requirements
  - processing of large data
  - split computations into different parallel tasks
  - make efficient use of large commodity clusters
- abstracts out
  - details of parallelization
  - fault tolerance
  - data distribution
  - monitoring
  - load balancing