Yandex

RDDs

Why do we need a new abstraction?

Why do we need a new abstraction?

- Example: iterative computations (K-means, PageRank, …)
  - relation between consequent steps is known only to the user code
  - framework must reliably persist data between steps (even if it is temporary data)

Why do we need a new abstraction?

- Example: iterative computations (K-means, PageRank, ...)
  - relation between consequent steps is known only to the user code
  - framework must reliably persist data between steps (even if it is temporary data)

- Example: joins
  - join operation is used in many MapReduce applications
  - not-so-easy to reuse code

Resilient Distributed Datasets

Resilient Distributed Datasets

- Resilient — able to withstand failures
- Distributed — spanning across multiple machines
- Formally, a read-only, partitioned collection of records

Resilient Distributed Datasets

- Resilient — able to withstand failures
- Distributed — spanning across multiple machines
- Formally, a read-only, partitioned collection of records

- To adhere to RDD interface, a dataset must implement:

Resilient Distributed Datasets

- Resilient — able to withstand failures
- Distributed — spanning across multiple machines
- Formally, a read-only, partitioned collection of records

- To adhere to RDD interface, a dataset must implement:
  - partitions() ➞ Array[Partition]
  - iterator(p: Partition) ➞ Iterator

Resilient Distributed Datasets

- Resilient — able to withstand failures
- Distributed — spanning across multiple machines
- Formally, a read-only, partitioned collection of records

- To adhere to RDD interface, a dataset must implement:
  - partitions() ➞ Array[Partition]
  - iterator(p: Partition, parents: Array[Iterator]) ➞ Iterator
  - dependencies() ➞ Array[Dependency]

Resilient Distributed Datasets

- Resilient — able to withstand failures
- Distributed — spanning across multiple machines
- Formally, a read-only, partitioned collection of records

- To adhere to RDD[T] interface, a dataset must implement:
  - partitions() ➡ Array[Partition]
  - iterator(p: Partition, parents: Array[Iterator[_]]) ➡ Iterator[T]
  - dependencies() ➡ Array[Dependency]
- ...and may implement other helper functions

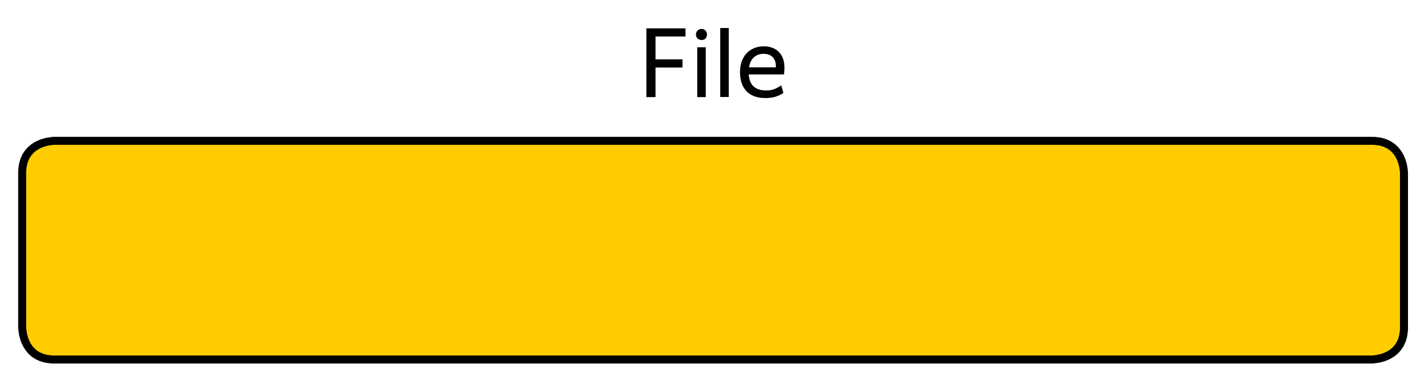
- Typed! RDD[T] — a dataset of items of type T

Example: a binary file in HDFS

- partitions() ➞ *Array[Partition]*

- iterator(p: *Partition*, parents: *Array[Iterator[_]]*) ➞
  ➞ *Iterator[Byte]*

- dependencies*()* ➞ *Array[Dependency]*

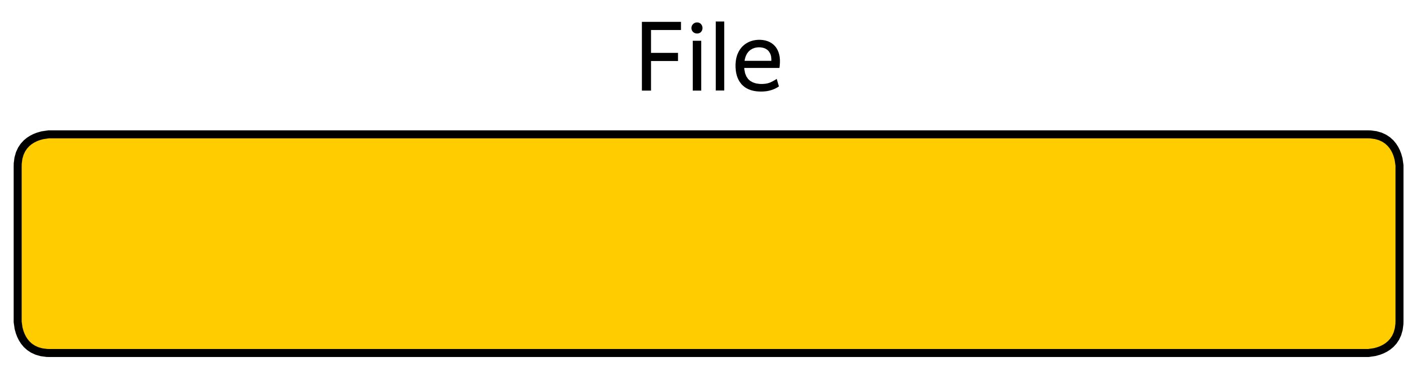


Example: a binary file in HDFS

- partitions() ➞ *Array[Partition]*
  - lookup blocks information from the NameNode
  - make a partition for every block
  - return an array of the partitions

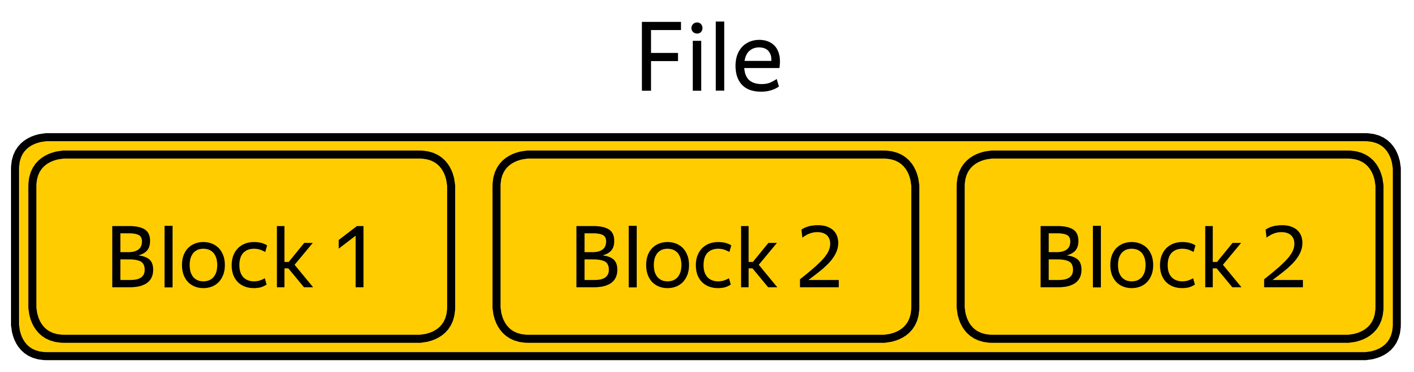
- iterator(p: *Partition*, parents: *Array[Iterator[_]]*) ➞
  ➞ *Iterator[Byte]*

- dependencies*()* ➞ *Array[Dependency]*



Example: a binary file in HDFS

- partitions() ➞ *Array[Partition]*
  - lookup blocks information from the NameNode
  - make a partition for every block
  - return an array of the partitions
- iterator(p*: Partition*, parents*: Array[Iterator[_]]) ➞*
  ➞ Iterator[Byte]
  - parents are not used
- dependencies*()* ➞ *Array[Dependency]*
  - return an empty array

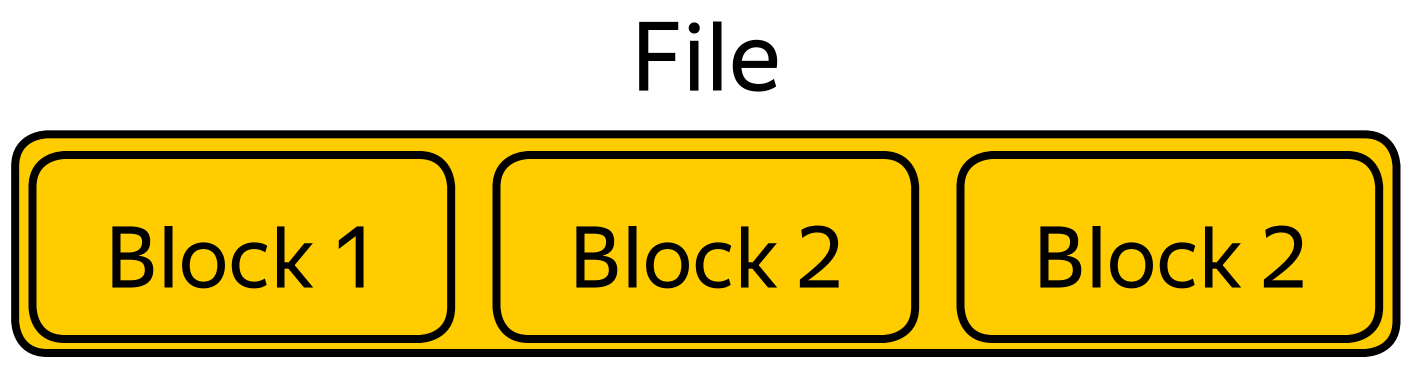


Example: a binary file in HDFS

- partitions() ➞ *Array[Partition]*
  - lookup blocks information from the NameNode
  - make a partition for every block
  - return an array of the partitions

- iterator(p: *Partition*, parents: *Array[Iterator[_]]*) ➞
  ➞ *Iterator[Byte]*
  - parents are not used
  - return a reader for the block of the given partitio

- dependencies*()* ➞ *Array[Dependency]*
  - return an empty array

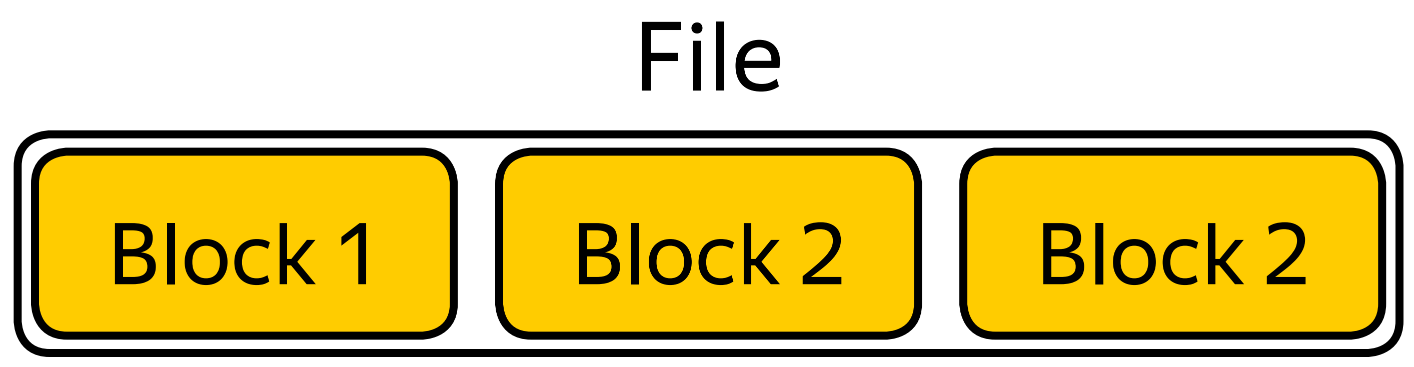


Example: a ~~binary~~ data* file in HDFS

- partitions*()* ➞ *Array[Partition]*
  - lookup blocks information from the NameNode
  - make a partition for every block
  - return an array of the partitions

- iterator*(*p*: Partition*, parents: *Array[Iterator[_]])* ➞ ➞ *Iterator[Byte]*
  - parents are not used
  - return a reader for the block of the given partition

- dependencies*()* ➞ *Array[Dependency]*
  - return an empty array



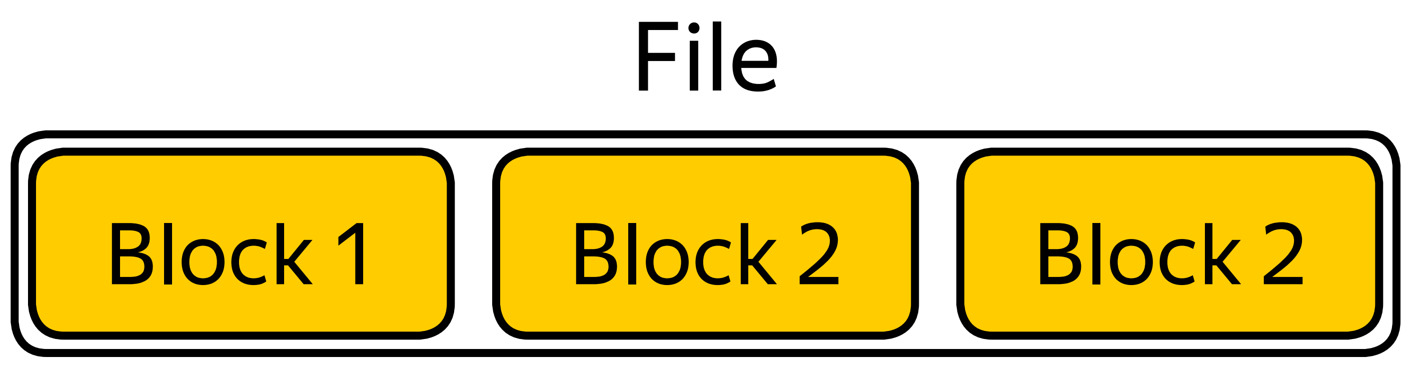
*a file encoded with the file format: see W1; think: text file, SequenceFile, Avro, RCFile

Example: a ~~binary~~ data* file in HDFS

- partitions*() ➞ Array[Partition]*
  - ~~lookup blocks information from the NameNode~~
    use InputFormat to compute InputSplits
  - make a partition for every ~~block~~ InputSplit
  - return an array of the partitions

- iterator(p*: Partition*, parents*: Array[Iterator[_]]) ➞*
  ➞ Iterator[~~Byte~~ InputRecord]
  - parents are not used
  - use InputFormat to create a reader for the InputSplit of the given partition
  - return ~~a reader for the block of the given partition~~ the reader

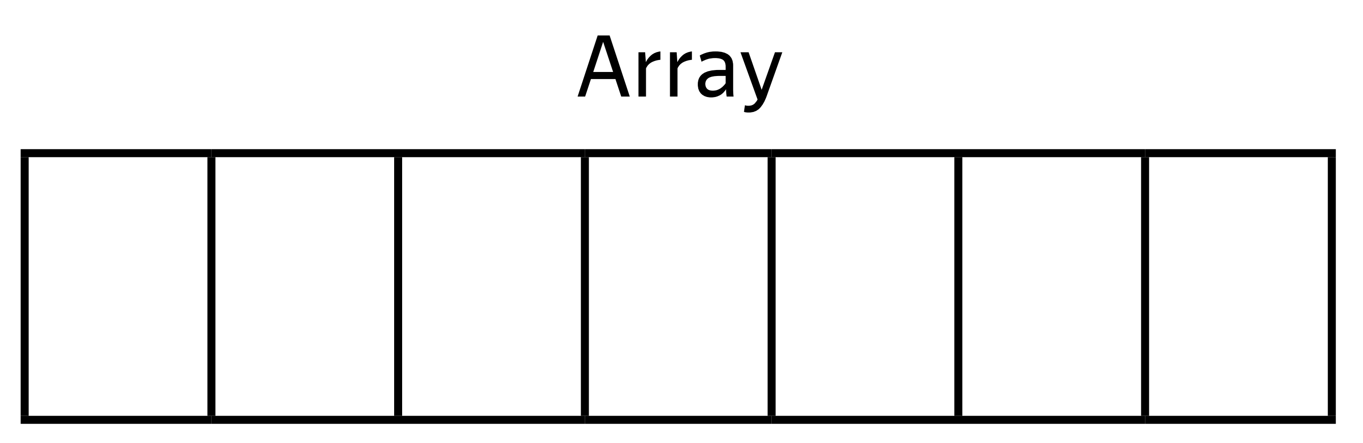
- dependencies*() ➞ Array[Dependency]*
  - return an empty array



*a file encoded with the file format: see W1; think: text file, SequenceFile, Avro, RCFile

Example: an in-memory array

- partitions() ➞ *Array[Partition]*
- iterator(p: *Partition*, parents: *Array[Iterator[_]]*) ➞ *Iterator[T]*
- dependencies() ➞ *Array[Dependency]*

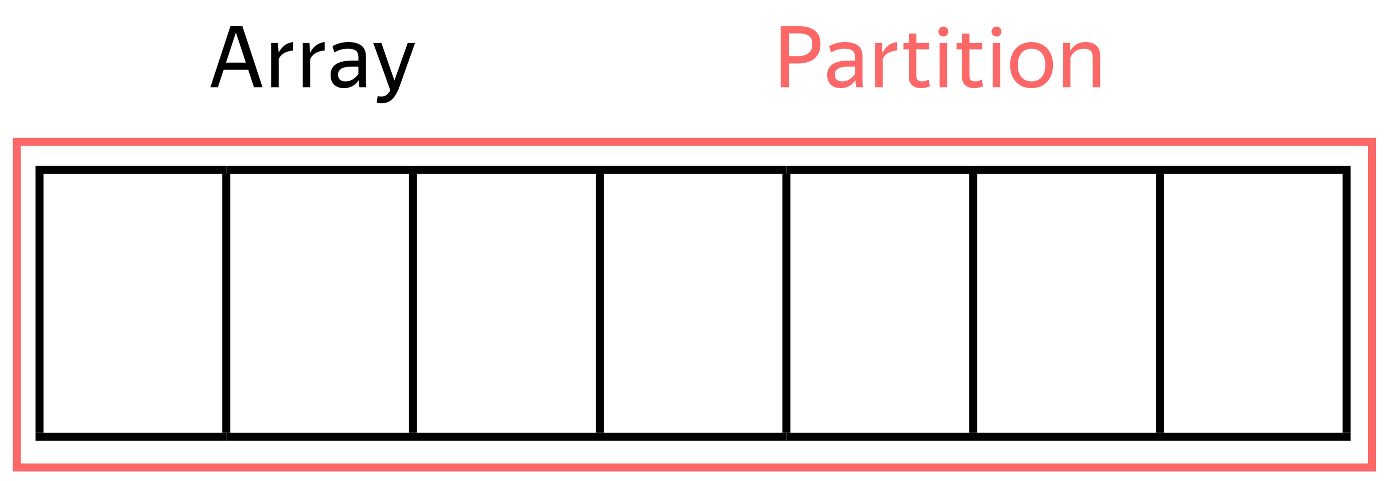


Example: an in-memory array

- partitions*() ➞ Array[Partition]*
  - return an array of a single partition with the source array

- iterator*(*p*: Partition*, parents*: Array[Iterator[_]]) ➞ Iterator[T]*
  - parents are not used
  - return an iterator over the source array in the given partition

- dependencies*() ➞ Array[Dependency]*
  - return an empty array (no dependencies)

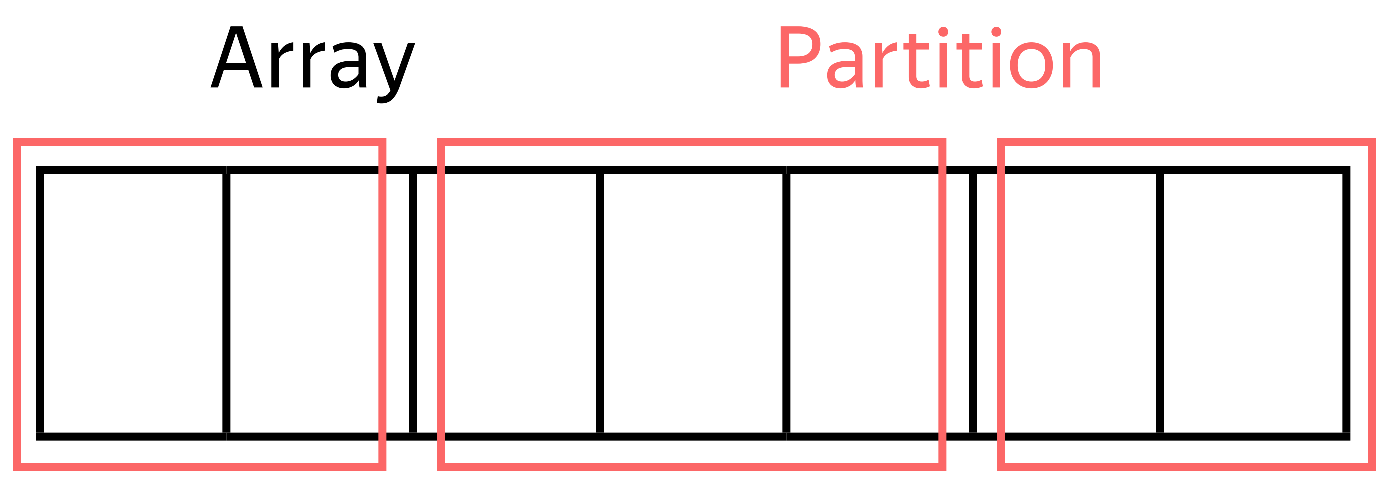


Example: an sliced* in-memory array

- partitions*()* ➞ *Array[Partition]*
  - slice array in chunks of size N
  - make a partition for every chunk
  - return an array ~~of a single partition with the source array~~ of the partitions

- iterator(p*: Partition*, parents*: Array[Iterator[_]])* ➞ *Iterator[T]*
  - parents are not used
  - return an iterator over the source array chunk in the given partition

- dependencies*()* ➞ *Array[Dependency]*
  - return an empty array (no dependencies)



*can be used to parallelize in-memory computations

Quiz

Summary

- RDD is a read-only, partitioned collection of records
  - a developer can access the partitions and create iterators over them
  - RDD tracks dependencies (to be explained in the next video)
- Examples of RDDs
  - Hadoop files with the proper file format
  - In-memory arrays
- Next video: Transformations

BigDATAteam