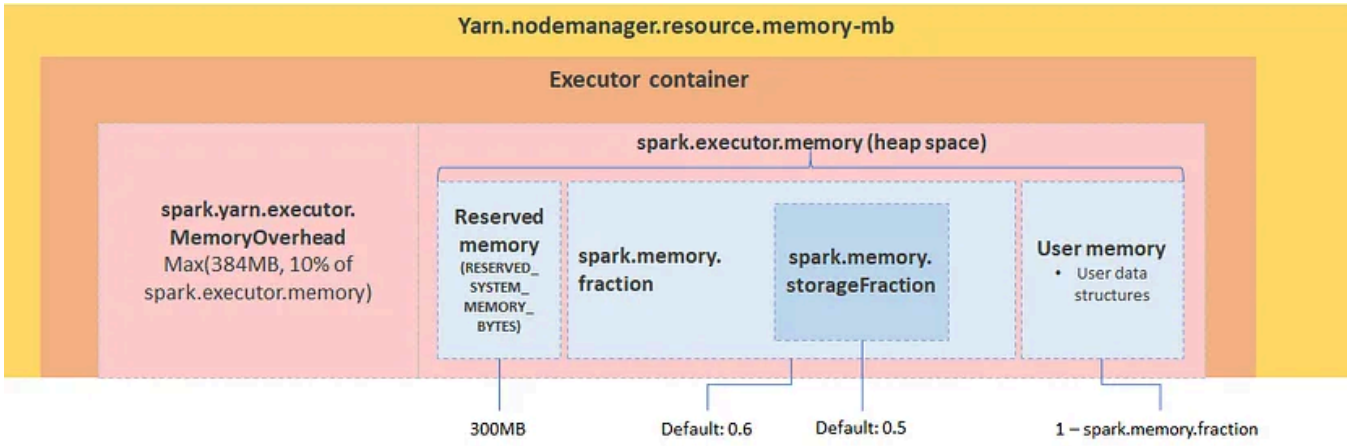
Spark Memory Management

Tharun Kumar Sekar · Follow
Published in Analytics Vidhya
3 min read · Jan 3, 2020

Let's try to understand how memory is distributed inside a spark executor.



Spark executor memory decomposition

In each executor, Spark allocates a minimum of 384 MB for the memory overhead and the rest is allocated for the actual workload.

The formula for calculating the memory overhead — max(**Executor Memory** * 0.1, 384 MB).

- 1st scenario, if your executor memory is 5 GB, then memory overhead = max(5 (GB) * 1024 (MB) * 0.1, 384 MB), which will lead to max(512 MB, 384 MB) and finally 512 MB.
  This will leave you with 4.5 GB in each executor for spark processing.

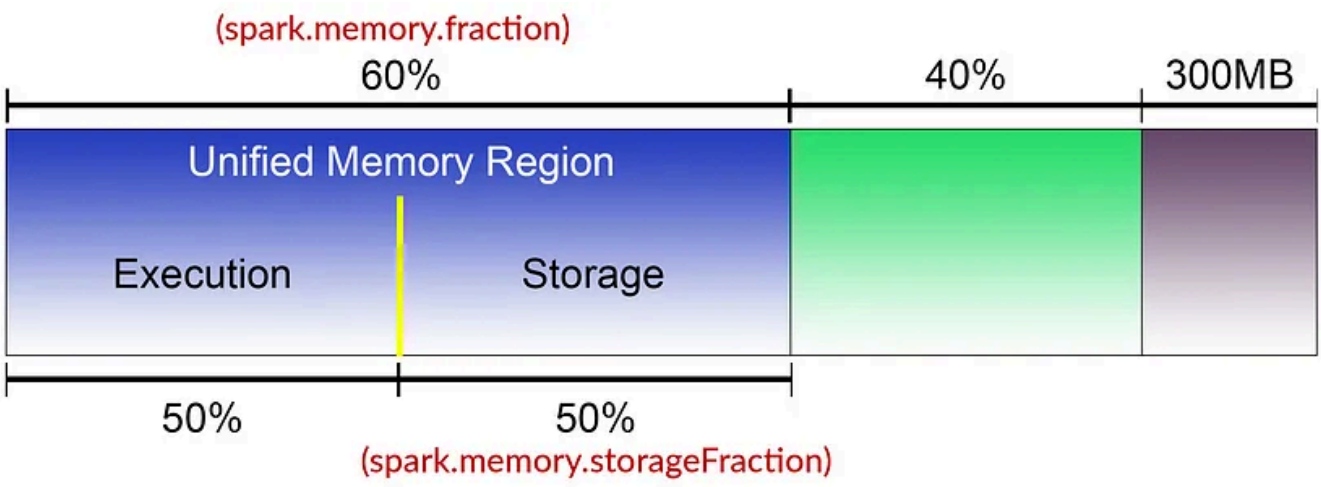
- 2nd scenario, if your executor memory is 1 GB, then memory overhead = max(1(GB) * 1024 (MB) * 0.1, 384 MB), which will lead to max(102 MB, 384 MB) and

finally 384 MB.
This will leave you with 640 MB in each executor for spark processing.

On Heap Memory

By default, Spark uses On-memory heap only. The On-heap **memory** area in the Executor can be roughly divided into the following four blocks:

- Storage Memory: It's mainly used to store Spark cache data, such as RDD cache, Unroll data, and so on.
- Execution Memory: It's mainly used to store temporary data in the calculation process of Shuffle, Join, Sort, Aggregation, etc.
- User Memory: It's mainly used to store the data needed for RDD conversion operations, such as the information for RDD dependency.
- Reserved Memory: The memory is reserved for the system and is used to store Spark's internal object

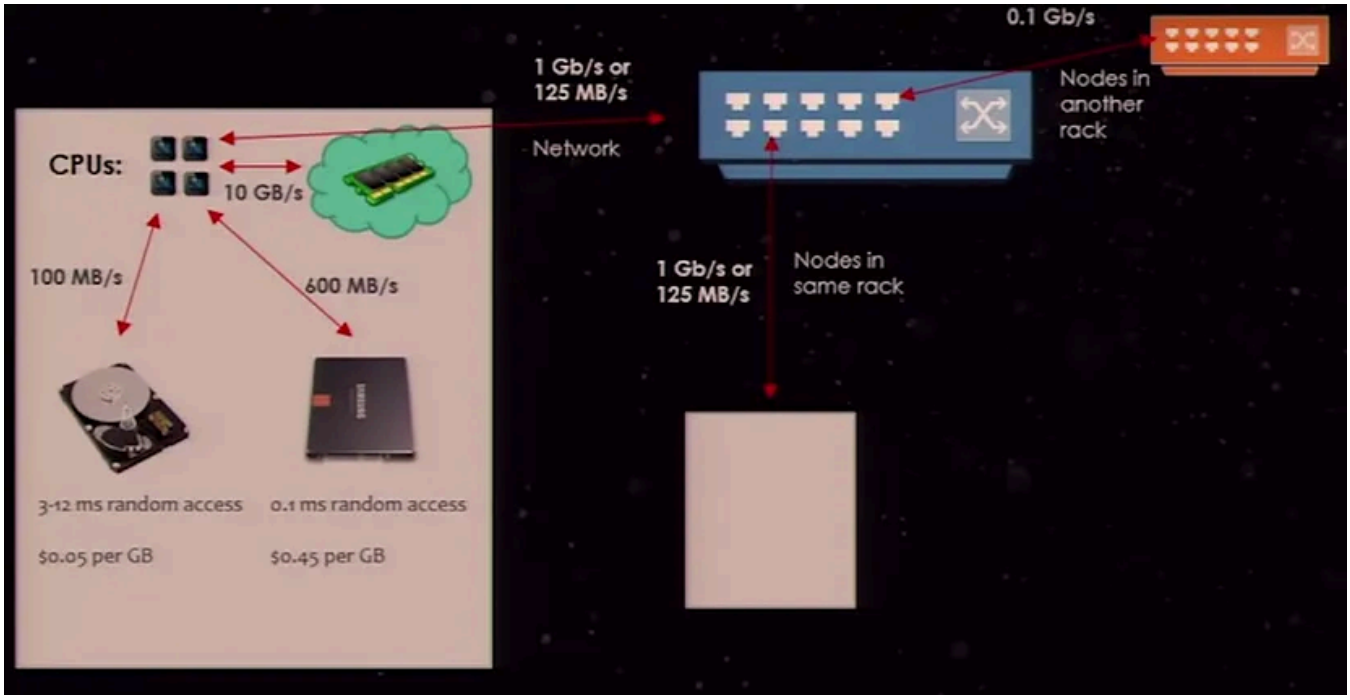


Memory Distribution

You have to consider two default parameters by Spark to understand this.

- spark.memory.fraction — to identify memory shared between Unified Memory Region and User Memory.
  In this case, it is 60%. Say, if we have 1 GB spark executor memory 600MB will be allocated for the Unified Memory Region and 400MB for User Memory.
- spark.memory.storageFraction — to identify memory shared between Execution Memory and Storage Memory. The default value provided by Spark is 50%. But

according to the load on the execution memory, the storage memory will be reduced to complete the task.



Spark Network Speed

One of the reasons Spark leverages memory heavily is because the CPU can read data from memory at a speed of 10 GB/s. Whereas if Spark reads from memory disks, the speed drops to about 100 MB/s and SSD reads will be in the range of 600 MB/s.

If CPU has to read data over the network the speed will drop to about 125 MB/s.

Common Issue CheckList

- Enough partitions for concurrency.
  If you have 20 cores, make sure you have at least 20 partitions or more than that.
- Minimize memory consumption by filtering the data you need.
- Minimize the amount of data shuffled. Shuffle is expensive.
- Know the standard library and use the right functions in the right place.

Spark | Spark Memory | Spark Executor | Spark Heap | Spark Internals

Follow

Written by Tharun Kumar Sekar

189 Followers · Writer for Analytics Vidhya

In case it helps others...

More from Tharun Kumar Sekar and Analytics Vidhya

Tharun Kumar Sekar in Analytics Vidhya

Understanding Resource Allocation configurations for a Spark application

Resource Allocation is an important aspect during the execution of any spark job. If not configured correctly, a spark job can consume...

Dec 23, 2019 1

Kia Eisinga in Analytics Vidhya

How to create a Python library

Ever wanted to create a Python library, albeit for your team at work or for some open source project online? In this blog you will learn...

Jan 27, 2020 29

Harikrishnan N B in Analytics Vidhya

Confusion Matrix, Accuracy, Precision, Recall, F1 Score

Binary Classification Metric

Dec 10, 2019 6

Tharun Kumar Sekar in Analytics Vidhya

Simple Method to choose Number of Partitions in Spark

At the end of this article, you will able to analyze your Spark Job and identify whether you have the right configurations settings for...

Dec 27, 2019 3

See all from Tharun Kumar Sekar

See all from Analytics Vidhya

Recommended from Medium

Prem Vishnoi(cloudvala) in Dev Genius

Apache-Spark : Resource Management (RM) and Scheduling within a Apache Spark Application

Managing resources and scheduling efficiently in a PySpark/Spark application is crucial to ensuring optimal performance, minimizing costs...

Aug 16

Nethaji Kamalapuram

Spark Memory allocation & Management

Lets talk about how memory allocation works for spark driver and executors. Understanding driver and executor memory allocation is crucial...

May 13 1

Lists

Natural Language Processing

1741 stories · 1333 saves

Yash Kothari

Coalesce in Spark, the internal working

Coalesce in spark is mainly used to reduce the number of partitions. Why is coalesce not as expensive as repartition?

Apr 30 1

Deepa Vasanthkumar

Spark Logical and Physical Plan Generation

In Spark, when you submit a SQL query or DataFrame transformation, it goes through several stages of processing before execution. Let us...

Apr 9

Kevin Wong

Intensive Spark Optimization Course

The Express Lane to Spark Mastery High-Speed Learning Guide for Spark Optimization

Apr 16 7

Shashwath Shenoy

Understanding Resource and Memory Management in Apache Spark!

Apache Spark is an open-source, distributed computing system that provides an interface for programming entire clusters with implicit data...

Jul 20

See more recommendations