Spark におけるディスクを用いた RDD キャッシングの高速化と効果的な利用に関する検討

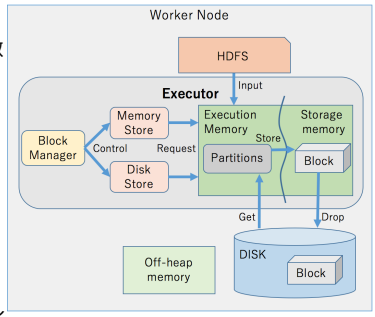
張 凱輝[†],谷村 勇輔[‡ †],中田 秀基[‡ †], 小川 宏高[‡]（† 筑波大, ‡ 産総研）

背景と目的

- Apache Spark（以下Spark）
  - オープンソースの並列データ処理フレームワーク
  - 中間データをメモリに保持
  - 機械学習やデータマイニングなどの反復計算が高効率
- ディスクと合わせて保持することで、より大量なデータを処理できるが、性能が低下する可能性がある
- 中間データをメモリとディスクを併用する場合とディスクのみを利用する場合の性能評価

Spark と RDD(Resilient Distributed Datasets)

- Spark は DriverNode と複数の WorkerNode からなる
- RDD は読み取り専用の分散データ構造、内部はパーティションに分割され、複数のワーカノードに分散配置、データ処理の単位として分散並列実行が可能
- ストレージレベルを指定することで、RDD はメモリやディスクに保存可能



＊RDDキャッシングの内部アルゴリズム
(STORAGE LEVEL: MEMORY_AND_DISK)

評価実験

• 調査項目

1. メモリとディスクの併用による性能評価と改善後の性能評価
2. ストレージデバイスによる性能評価

• 調査方法

1. Spark の機械学習ライブラリ (Mllib) に含まれたベンチマーク (DenseKMeans) と独自のベンチマーク (RDDTest) を実行
2. ストレージレベル、ストレージデバイス、RDDのサイズ、スレッド数、などを変更し、性能を測定

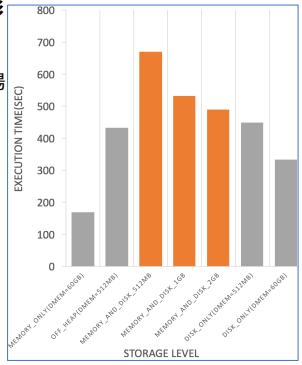
• 実験環境

CPU	Intel Xeon CPU E5-2620v3 2.40GHz, 6 cores x2
Memory	128 GB
Network	10 Gbps (for HDFS connection)
NVMe-SSD	Intel SSD DC P3700
SSD	OCZ Vertex3 (240GB, SATA6G I/F)
HDD	Hitachi Travelstar 7K320 (SATA3G I/F)
OS	Ubuntu 14.04 (Kernel v.3.13)
File System	Ext4

- Spark v2.1.0 を用いローカルモードで各ベンチマークを実行
- DenseKMeansの入力データは HiBench (6.0) 用いて生成、データサイズ：4015 MB

実験結果（１）

- **メモリとディスクの併用による影響 (DenseKMeans)**
- メモリが 512MB、1GB、2GB の場合の実行時間が DISK_ONLY より長い
- ガベージコレクションの頻発とブロックの繰り返しドロップが原因

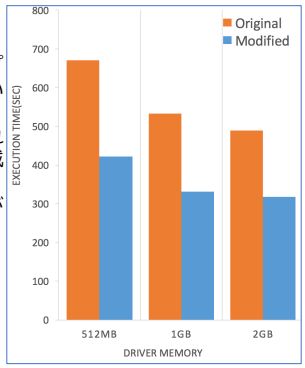


ドロップされたブロック数とサイズ

	#Blocks	Size (MB)
MEMORY_AND_DISK_512MB	600	2,577
MEMORY_AND_DISK_1GB	746	26,793
MEMORY_AND_DISK_2GB	715	20,936
OFF_HEAP_512MB	305	1,332

メモリとディスクを併用する場合のドロップを削減

- 提案手法：一回ディスクへドロップしたブロックをメモリに戻させない
- ドロップの発生回数とドロップされたフロックの合計サイスか大幅削減
- 再ドロップを抑制することで性能が向上することか確認



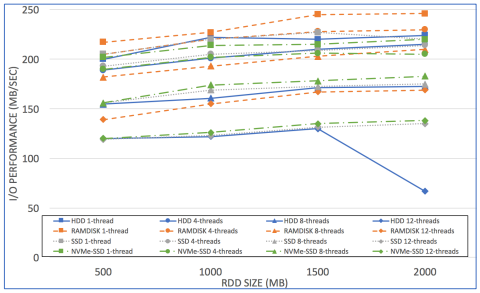
ドロップされたブロック数とサイズ

	Original		Modified	
	#Blocks	Size(MB)	#Blocks	Size(MB)
MEMORY_AND_DISK_512MB	600	2,577	133	610
MEMORY_AND_DISK_1GB	746	26,793	33	486
MEMORY_AND_DISK_2GB	715	20,936	58	473

実験結果（２）

ストレージデバイスによる違い（RDDTest)

• ストレージレベル：DISK_ONLY、Driverメモリ量：64GB



- OS のバッファキャッシュにより、ディスクの性能はボトルネックになりにくい
- HDD ではスレッド数が12、RDD サイズが2000MB でI/O 性能が低下

まとめ

- 再ドロップの繰り返しは性能に影響を与える
- 再ドロップを抑制する修正を施すことで、その問題が解決
- ディスクの性能はボトルネックになりにくいが、RDD サイズおよびスレッド数を増加により、ディスクの性能が重要になる

今後の課題

- シリアライズなど Spark 内部の仕組みの改善が必要
- GCアルゴリズムの選択や調整により、今回提示した指針は変わるのか変わらないのかについて評価

- この成果の一部は、国立研究開発法人新エネルギー・産業技術総合開発機構 （ＮＥＤＯ）の委託業務の結果得られたものです。
- 本研究はJSPS科研費 JP16K00116の助成を受けたものです。