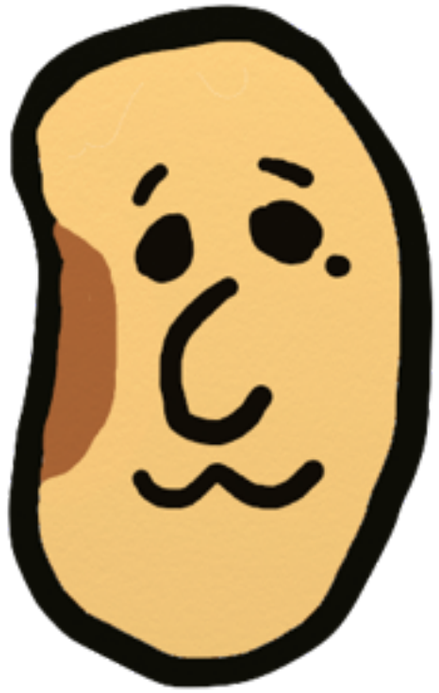
Rubyでデータ分析が出来る未来

軽量Ruby普及・実用化促進フォーラム2017

by 畑中 悠作

自己紹介

- 畑中 悠作
- hatappi hatappi1225
- 株式会社Speee
- 軽量Ruby歴はMItamae



UZOU

UZOUはコンテンツやユーザーを分析した上で最適な広告を配信し、
ユーザーのまだ見ぬ体験を提供する、ネイティブアド配信プラットフォームです。

媒体様向け

広告主様・広告代理店様向け

Rubyは好きですか？

軽量Rubyで集めたビックデータ
どのように分析されてますか？

ビックデータについて

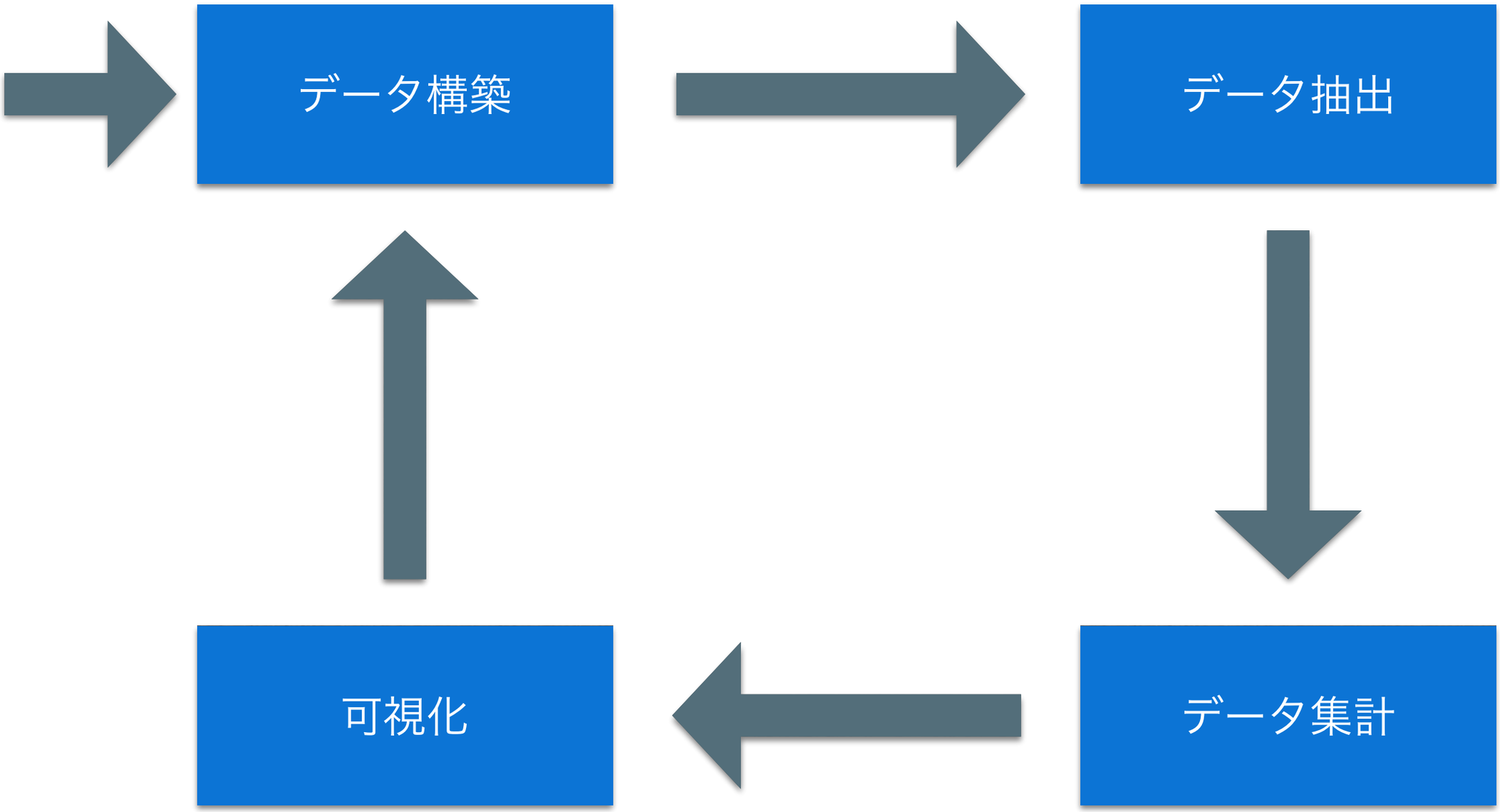
- データ量が多い
- 急速に増える
- 元となるデータの種類が様々

ビックデータの分析が出来ると

- オススメ商品をレコメンドすることが出来る
- 電子機器類の測定されたデータから欠陥品を探すことが出来る
- レントゲンの結果から潜在的な病気を見つけることが出来る

Rubyでデータ分析が出来るよ
うになると良いと思いませんか

データ分析について



データ分析について

- Pandas: Python製のデータ分析用ライブラリ
- Cassandra: 分散データベース管理システム
- Spark: 高速でスケーラブルな汎用分散処理エンジン
- HBase: 列指向、分散データベース

データ分析には複数のシステムを併用して
使用するケースが多い

システム間のデータ連携

それぞれのシステムでは独自のメモリ構造をもっておりデータ交換を行う際にCSVなどのファイルやParquetと呼ばれるディスク上で列指向でデータを扱うフォーマットを使うなどして連携されている

無視の出来ない

データ交換コスト!!

Apache Arrow

- https://arrow.apache.org/
- 2016年の10月に0.1.0がリリース

- メモリ上でカラム型データを扱うためのフォーマットとアルゴリズム
- Apache Software Foundationのトップレベルプロジェクト

ロー型とカラム型

ロー型

行でデータを管理

ID	商品名	価格	発売日
1	ペン	100	2017/1/1
2			
3	定規	30	2010/12/2

カラム型

列でデータを管理

ID	商品名	価格	発売日
1	ペン		2017/1/1
2	消しゴム		2016/11/1
3	定規		2010/12/2

- カラム型は列単位でデータを格納することで格納されたデータは同じ型のため圧縮効率が良い
- 列で集計をしたい場合などに他の列を参照しないのでロー型よりも余分なデータを参照しないため早い

Apache Arrowの特徴

- Fast
- Flexible
- Standard

Apache Arrowの特徴

- Fast
- Flexible
- Standard

Fast

- 最新のプロセッサに含まれるSIMD (Single Instruction/Multiple Data)を使用することが出来る

- カラム型のデータを扱えることでCPUキャッシュを効率よく使用することが出来る

- コピーなしでデータを読み込むZeroCopyをサポート

Apache Arrowの特徴

- Fast
- Flexible
- Standard

Flexible

- Java, C, C++, Pythonなど各種言語から扱うことが出来るのArrow対応したシステムを柔軟に扱うことが出来る
- 他の言語に関しても対応は進められている

Apache Arrowの特徴

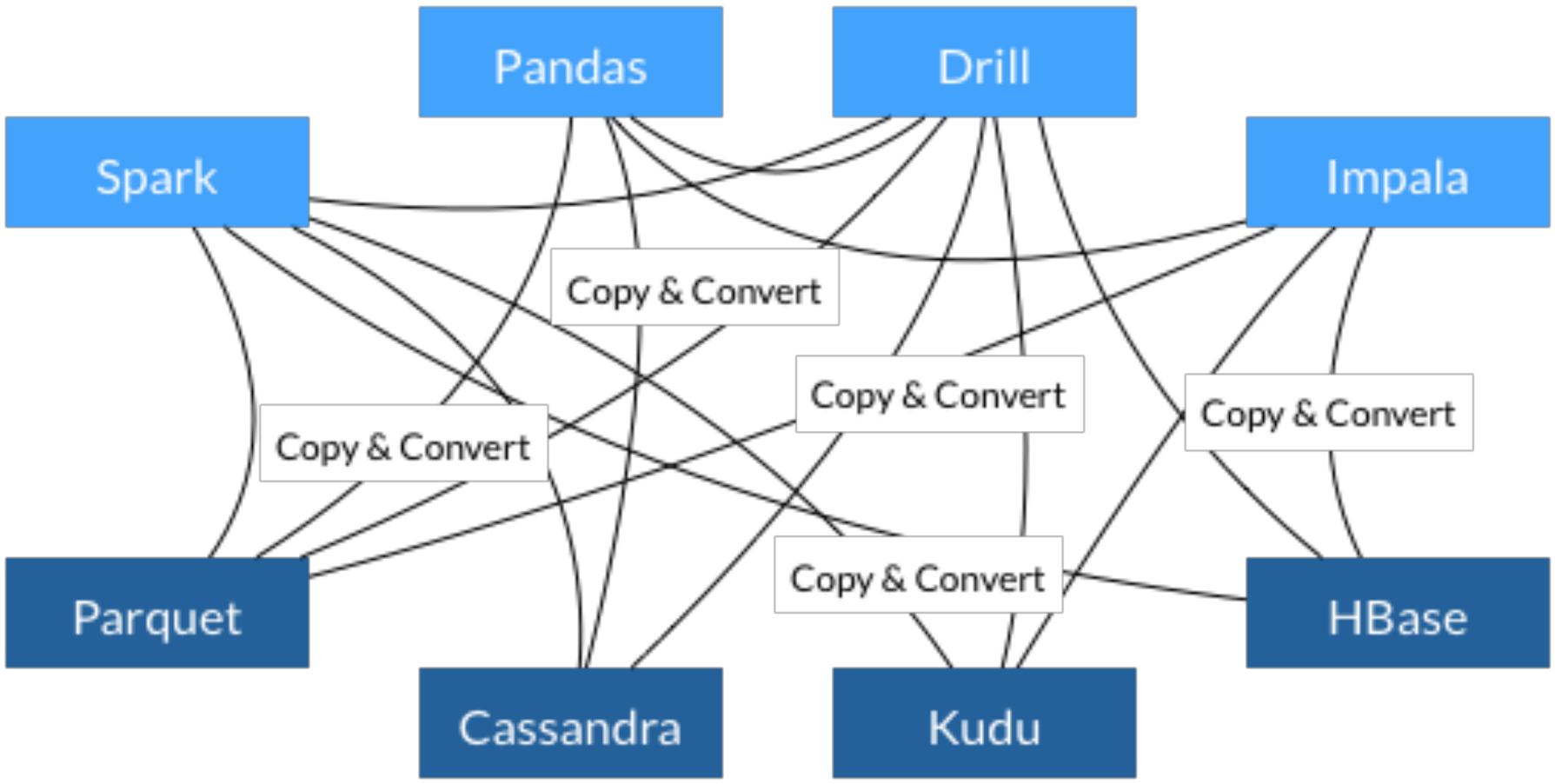
- Fast
- Flexible
- Standard

Standard

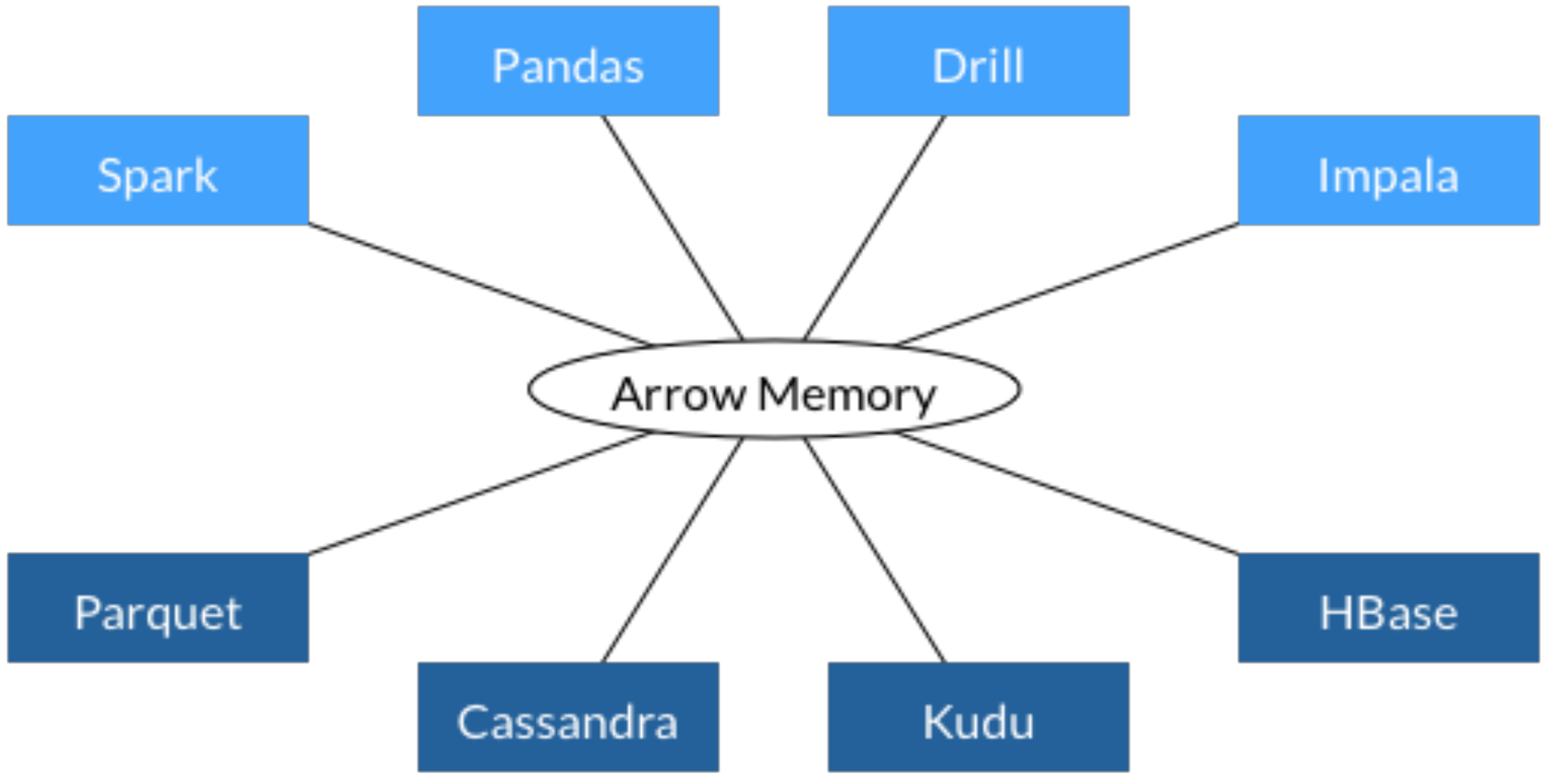
- Calcite, Cassandra, Drill, Hadoop, HBase, Ibis, Impala, Kudu, Pandas, Parquet, Phoenix, Spark, Stormの13のビックデータ関連プロジェクトの開発者が参加

- ApacheIncubatorが省略されたApache Software Foundationの**トップレベルプロジェクト**

Apache Arrowがない時



Apache Arrowがある時



Rubyが対応すれば？！

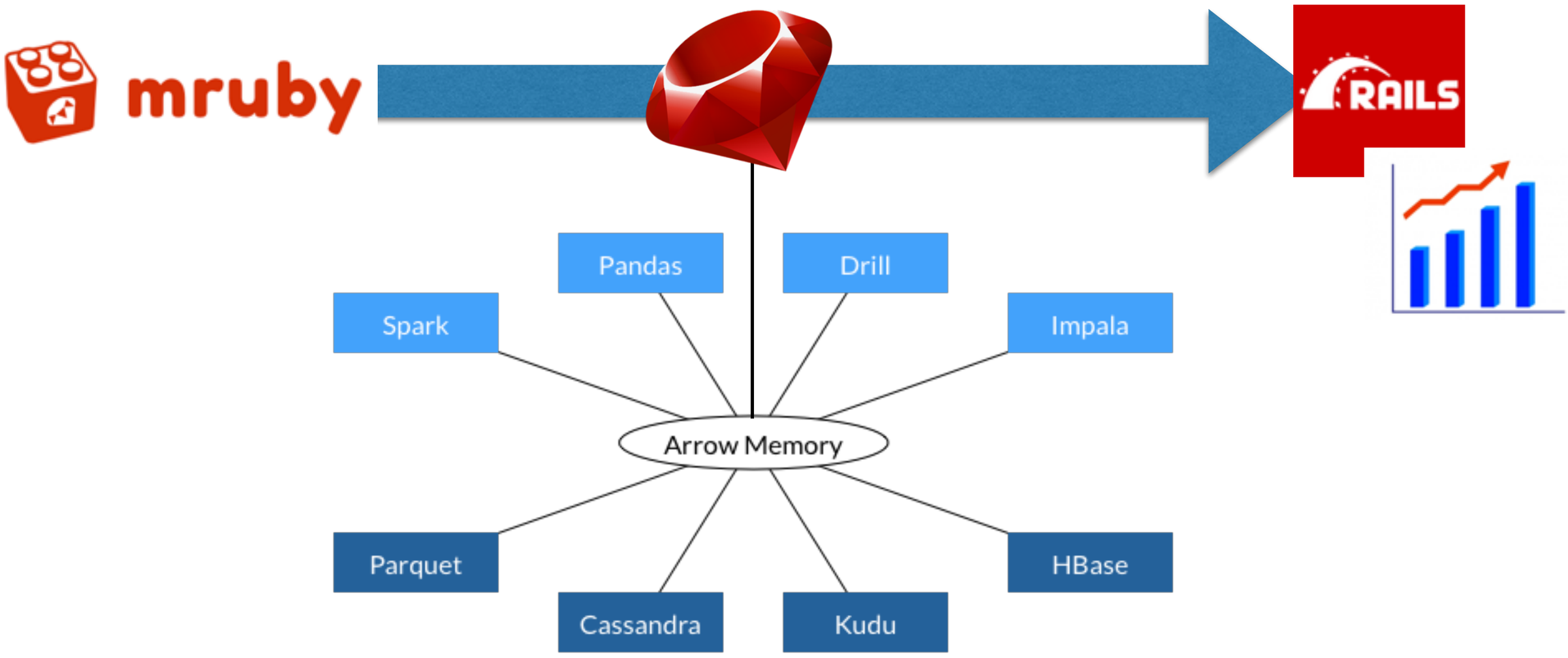
Apache Arrowに対応することで必要な部分から
Rubyを使ったデータ分析をはじめることが出来る

例

- Rubyで集めたデータをArrowに対応している PandasやSparkに連携し分析した結果をRubyで受け取ってRailsを使ったwebアプリで可視化

- PandasやSpark部分を徐々にRubyへ移行することも出来る

イメージ


mruby
RAILS
Pandas
Drill
Spark
Impala
Arrow Memory
Parquet
HBase
Cassandra
Kudu

RubyからApache Arrow
を使うことが出来るのか

Apache Arrow

Powering Columnar In-Memory Analytics

Join Mailing List | Install (0.6.0 Release - August 14, 2017)

Latest News: Apache Arrow 0.6.0 release

Fast

Apache Arrow™ enables execution engines to take advantage of the latest SIM D (Single input multiple data) operations included in modern processors, for native vectorized optimization of analytical data processing. Columnar layout of data also allows for a better use of CPU caches by placing all data relevant to a column operation in as compact of a format as possible.

The Arrow memory format supports **zero-copy reads** for lightning-fast data access without serialization overhead.

Flexible

Arrow acts as a new high-performance interface between various systems. It is also focused on supporting a wide variety of industry-standard programming languages. Java, C, C++, Python, Ruby, and JavaScript implementations are in progress and more languages are welcome.

Standard

Apache Arrow is backed by key developers of 13 major open source projects, including Calcite, Cassandra, Drill, Hadoop, HBase, Ibis, Impala, Kudu, Pandas, Parquet, Phoenix, Spark, and Storm making it the de-facto standard for columnar in-memory analytics.

Apache Arrow

Powering Columnar In-Memory Analytics

Join Mailing List | Install (0.6.0 Release - August 14, 2017)

Latest News: Apache Arrow 0.6.0 release

Fast

Apache A…rines to take adva… input multiple … processo… … analytica… … data also… caches by placing a… data relevant to a column operation in as compact of a format as possible.

The Arrow memory format supports **zero-copy reads** for lightning-fast data access without serialization overhead.

Flexible

Arrow acts as a new high-performance interface between various systems. It is also focused on supporting a wide variety of industry-standard programming languages. Java, C, C++, Python, Ruby, and JavaScript implementations are in progress and more languages are welcome.

Standard

Apache Arrow is backed by key developers of 13 major open source projects, including Calcite, Cassandra, Drill, Hadoop, HBase, Ibis, Impala, Kudu, Pandas, Parquet, Phoenix, Spark, and Storm making it the de-facto standard for columnar in-memory analytics.

前日にPRを出してマージ

apache / arrow
mirrored from git://git.apache.org/arrow.git

Watch 161

Code | Pull requests 17 | Projects 0 | Insights

ARROW-1441: [Site] Add Ruby to Flexible section #1021

Closed **hatappi** wants to merge 2 commits into `apache:master` from `hatappi:ARROW-1441`

Conversation 3 | Commits 2 | Files changed 1

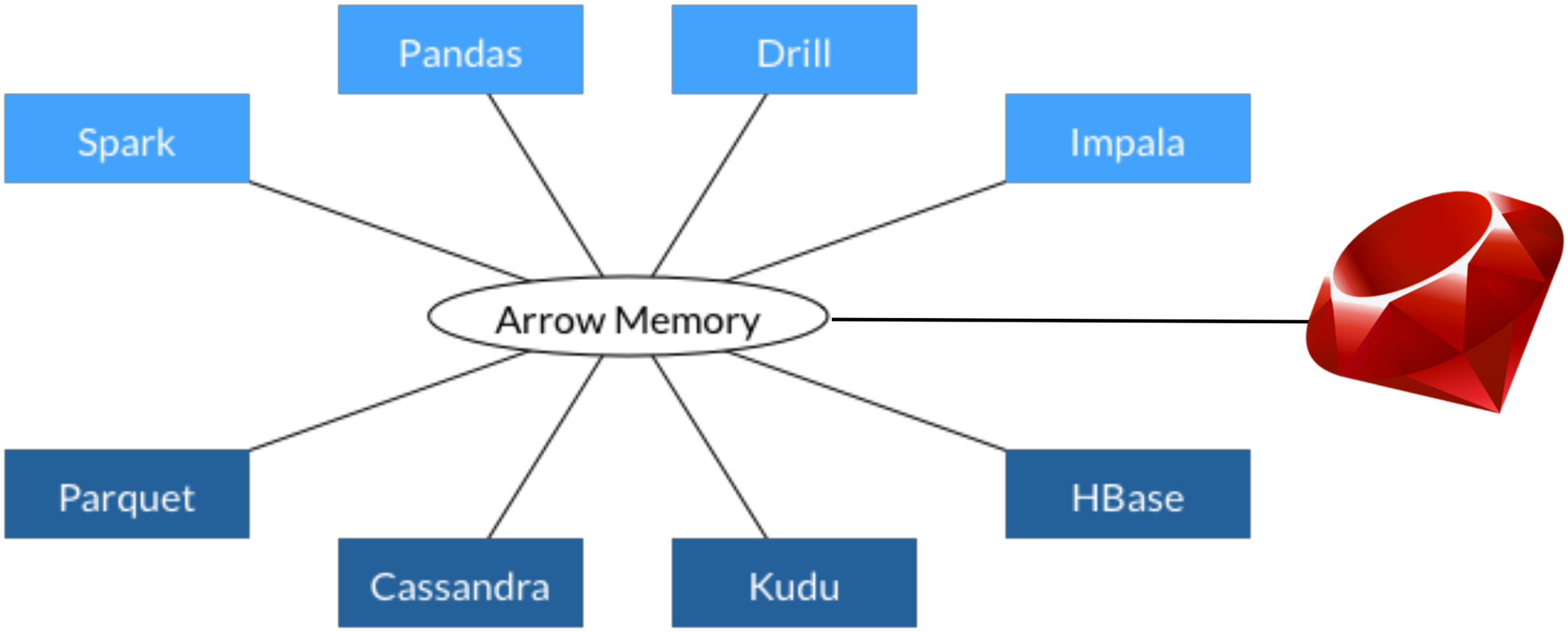
hatappi commented 18 hours ago — Contributor

2017-05-08 C (GLib) Bindings, with support for Ruby, Lua, and more release.
There is a Ruby binding called red-arrow.
How about adding Ruby to the top page Flexible section?
https://arrow.apache.org/

issue link is here

red-data-tools/red-arrow

Rubyもデータ分析の仲間入り🎉



Apache Arrowを体験

検証内容

- Pythonで約2GのデータをCSV, Apache Arrowで書き込みRubyでそれぞれを読み込む
- 各フォーマットにおける書き込みと読み込みの時間を計測して比較を行う

検証環境

- AWSのEC2インスタンスのt2.large
  - vCPU: 2, メモリ: 8G
    - Ubuntu: 16.04.2 LTS
- Python: 3.6.2
- Ruby: 2.4.1
- Apache Arrow: 0.6.0

CSV (書き込み)

```python
#-*- using:utf-8 -*-
import pandas as pd
import pyarrow as pa
from time import time

df = pd.DataFrame({"a": ["a" * i for i in list(range(40000))],
                   "b": ["b" * i for i in list(range(40000))],
                   "c": ["c" * i for i in list(range(40000))]})

start_time = time()
df.to_csv("test.csv", header=False)
print("write : {0}s".format(time() - start_time))
```

CSV (読み込み)

```ruby
require "time"
require "csv"

start_time = Time.now
CSV.open("test.csv") do |row|
    puts "batch size : #{row.count}"
end
puts "read : #{Time.now - start_time}s"
```

Apache Arrow (書き込み)

```python
#-*- using:utf-8 -*-
import pandas as pd
import pyarrow as pa
from time import time

df = pd.DataFrame({"a": ["a" * i for i in list(range(40000))],
                   "b": ["b" * i for i in list(range(40000))],
                   "c": ["c" * i for i in list(range(40000))]})
start_time = time()
record_batch = pa.RecordBatch.from_pandas(df)
with pa.OSFile("/dev/shm/pandas.arrow", "wb") as sink:
    schema = record_batch.schema
    writer = pa.RecordBatchFileWriter(sink, schema)
    writer.write_batch(record_batch)
    writer.close()
print("write : {0}s".format(time() - start_time))
```

Apache Arrow (読み込み)

```
require "arrow"
require "time"
require "csv"

Input = Arrow::MemoryMappedInputStream

start_time = Time.now
Input.open("/dev/shm/pandas.arrow") do |input|
  reader = Arrow::RecordBatchFileReader.new(input)
  puts "batch size : #{reader.get_record_batch(0).count}"
end
puts "read : #{Time.now - start_time}s"
```

DEMO

結果

	書き込み (秒)	読み込み (秒)
CSV	51.40	4.87
Apache Arrow	3.31	0.01

※2.3Gのデータをそれぞれ5回計測した平均を記載

http://hatappi.hateblo.jp/entry/2017/09/01/002458

Red Data Tools

Red Data Tools

- 株式会社クリアコードの須藤さんが2017年2月にプロジェクトを設立

- Ruby用のデータ処理ツールを提供することを目的としたプロジェクト

- 多くの言語が共通して使用できるApache Arrowを使用することで**Rubyコミュニティーを超えて協力する**

活動内容

活動内容

- Red Arrow
- 既存gemのArrow化
- 新しいツールを提供

活動内容

- Red Arrow
- 既存gemのArrow化
- 新しいツールを提供

Red Arrow

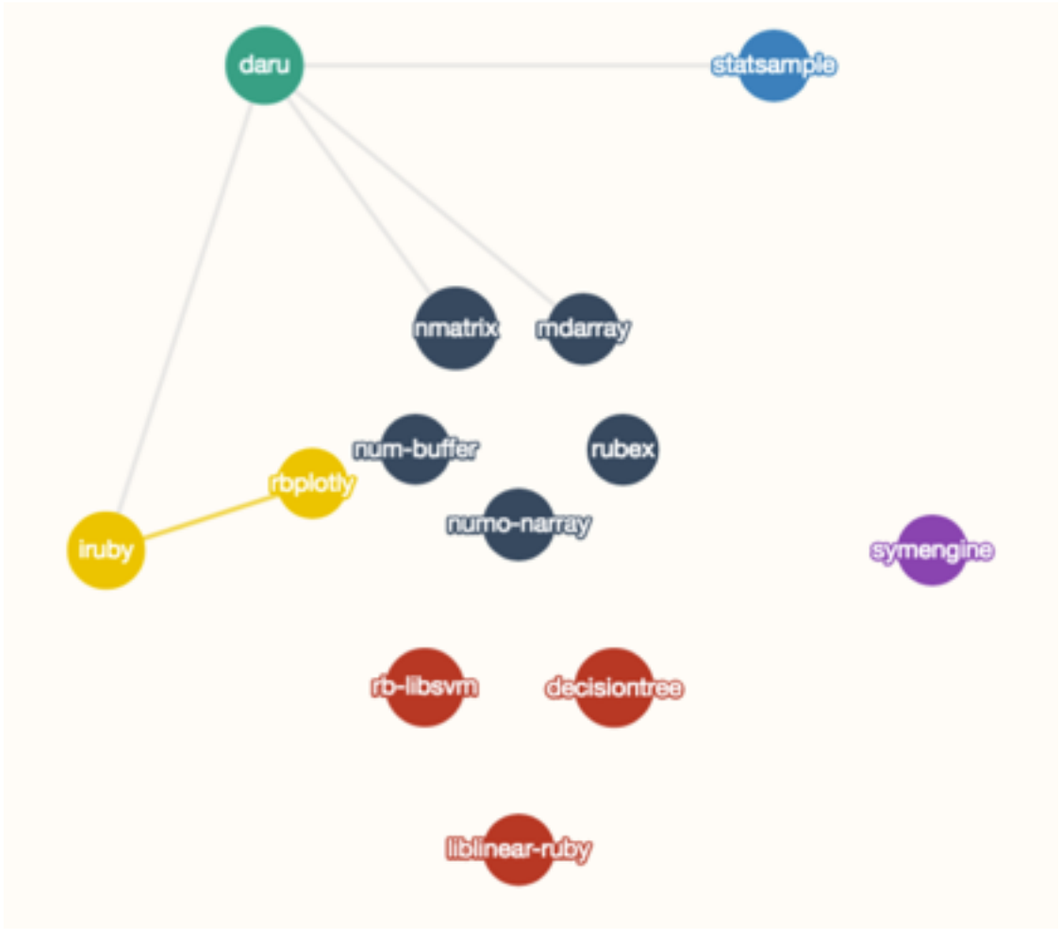
- Apache ArrowのRubyバインディング
- Rubyバインディングの開発だけでなくApache Arrow本体の開発も行っている

活動内容

- Red Arrow
- 既存gemのArrow化
- 新しいツールを提供

SciRuby

- 科学技術計算、データ可視化用途のGem群の総称
- Pythonでいうpandasにあたるdaruやnumpyにあたるnumo-narryなどがある
- Gem間が独立していて連携が難しい

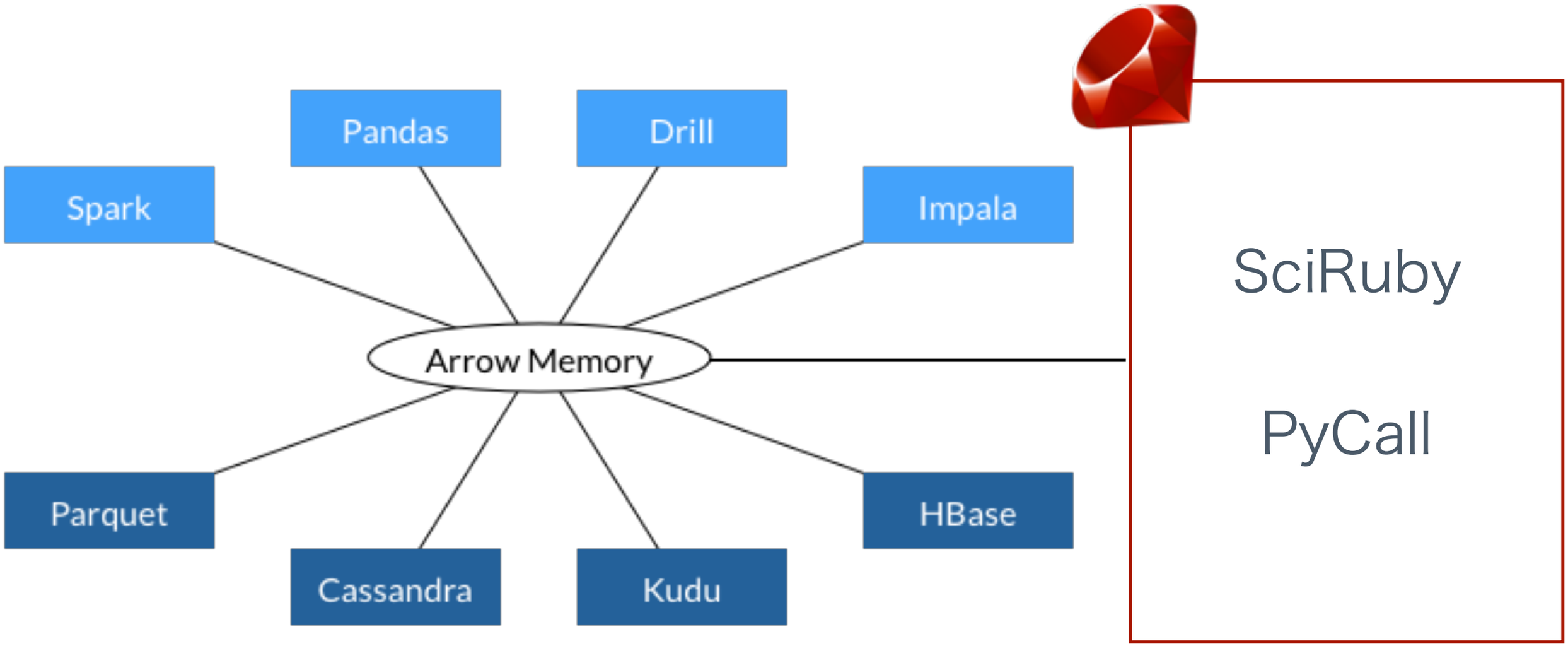


PyCall

- @mrknさんが作成されているRubyとPythonのブリッジライブラリ
- Pythonで作成された既存の資産を使うことが出来るのでRubyでPythonのオブジェクトを使用することが出来る

```ruby
require 'pycall/import'
include PyCall::Import

pyimport 'numpy', as: :np

x = np.matrix.([[1,2], [3, 4]])
# => matrix([[1, 2],
#            [3, 4]])
y = np.matrix.([[5,6], [7, 8]])
# => matrix([[5, 6],
#            [7, 8]])

z = x * y
# => matrix([[19, 22],
#            [43, 50]])
z.max.()
# => 50
```

既存gemのArrow化



Apache Arrowに対応させることで
既存のgemを使用してデータ分析をはじめられる

活動内容

- Red Arrow
- 既存gemのArrow化
- 新しいツールを提供

新しいツールを提供

- red-data-tools/xtensor-arrow-glib
  - xtensorと呼ばれるC++で実装された多次元配列を扱うライブラリのApace Arrow対応
- red-data-tools/red-chainer
  - Python製の深層学習フレームワークのChainerをRubyへポーティングしたもの

Chainerとは

- 株式会社Preferred Networks(PFI)が開発した深層学習フレームワーク

- 国内産!!

- Python製で柔軟性、直感的、高機能といった3つの特徴をもっている

- インストールが楽 pip install chainer

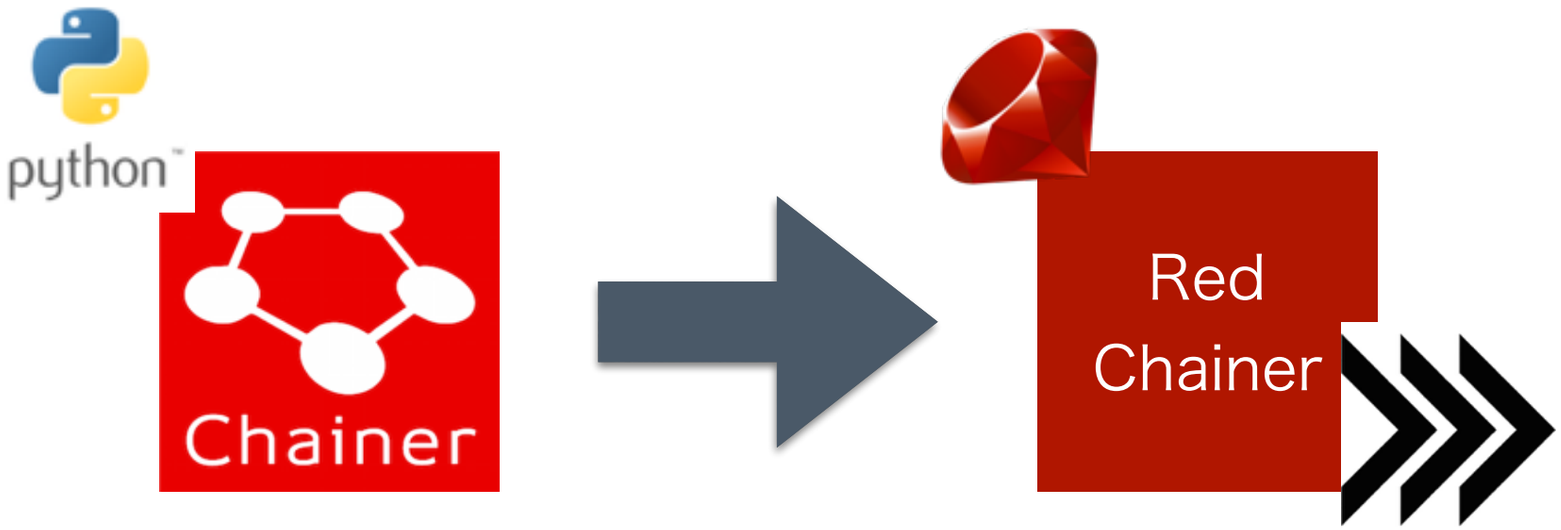
- GPUも対応してる
  http://hatappi.hateblo.jp/entry/2017/07/29/135053

red-data-tools/red-chainer

- Chainerのクラスやパラメータの持ち方を参考にRubyで書くことで**0から作るよりもスムーズに作成することが出来る**

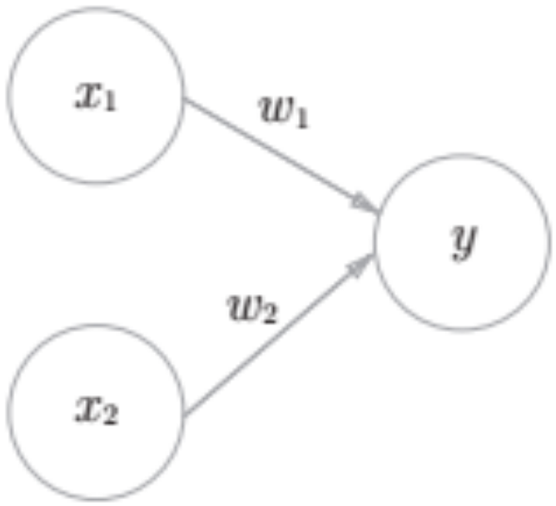
- Chainer本体へのフィードバックもしていく

- Apache Arrowに対応したNumo::NArrayを使用することでApache Arrowに対応した深層学習フレームワークを実現することが出来る

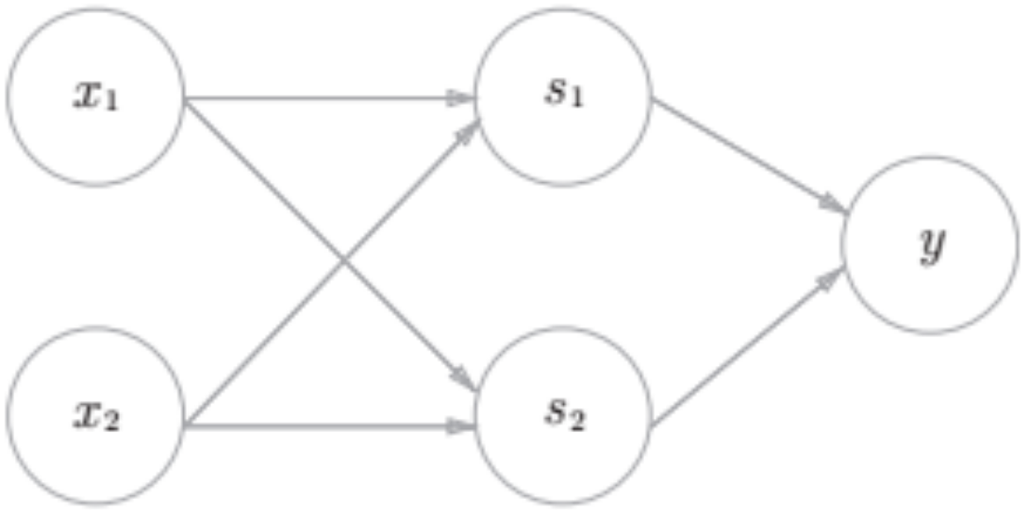


Multi Layer Perceptron (MLP)

- パーセプトロンは複数の信号を受け取り、ひとつの信号を出力する



- パーセプトロンはいくつも重ねることが出来る。これを多層パーセプトロンという



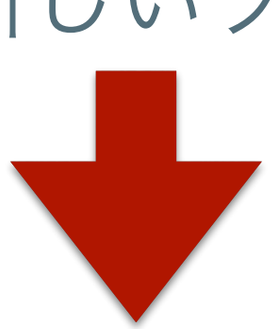
Red ChainerのMLP実装

```ruby
require 'chainer'

class MLP < Chainer::Chain
  L = Chainer::Links::Connection::Linear
  R = Chainer::Functions::Activation::Relu

  def initialize(n_units, n_out)
    super()
    init_scope do
      @l1 = L.new(nil, out_size: n_units)
      @l2 = L.new(nil, out_size: n_units)
      @l3 = L.new(nil, out_size: n_out)
    end
  end

  def call(x)
    h1 = R.relu(@l1.(x))
    h2 = R.relu(@l2.(h1))
    @l3.(h2)
  end
end
```

Red Data Toolsの今後

- 引き続き既存のgemのAapche Arrowへの対応は行う

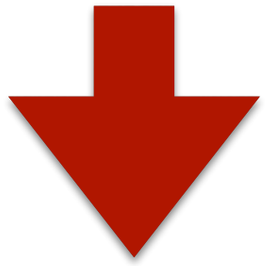
- red-chainerのような新しいツールの作成



Ruby間でデータ分析が
出来るようになりたい

Red Data Toolsの今後

- Apache Arrow本体の開発の追従
- Rubyバインディングの作成



Ruby間だけでなく言語も超えて協力して
データ分析が出来るようになりたい

Rubyが対応すれば？！

Apache Arrowに対応することで必要な部分から
Rubyを使ったデータ分析をはじめることが出来る

例

- Rubyで集めたデータをArrowに対応している PandasやSparkに連携し分析した結果をRubyで受け取ってRailsを使ったwebアプリで可視化

- PandasやSpark部分を徐々にRubyへ移行することも出来る

まとめ

- 軽量Rubyで集めたデータをRubyで分析が出来るようになればRubyだけで完結することが出来る
- Apache Arrowに対応することで必要な部分からRubyでデータ分析をはじめることが出来る
- Rubyでもデータ分析が出来る未来へ！