Computer Systems Research

Kexin Rong

CS197 09/26/19

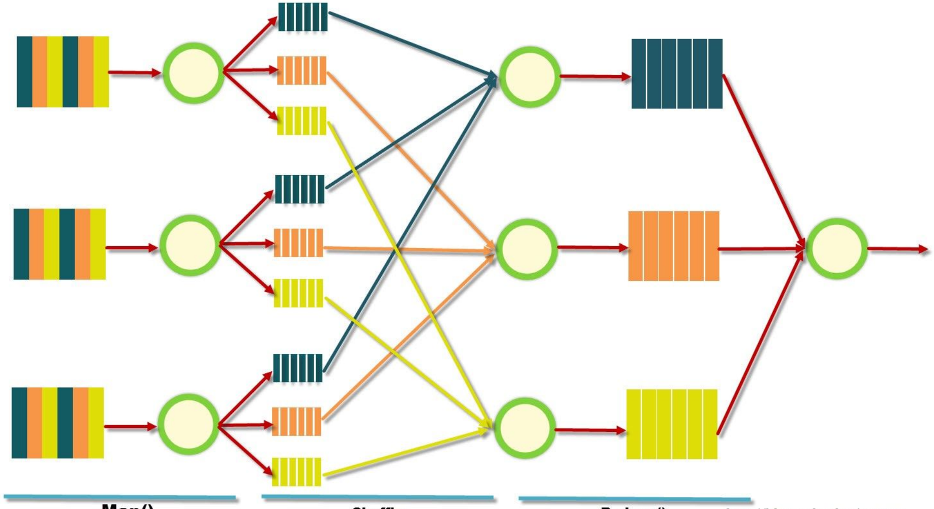
Agenda

- Area overview
- Introductions
- Project overview
- (maybe git tutorial)

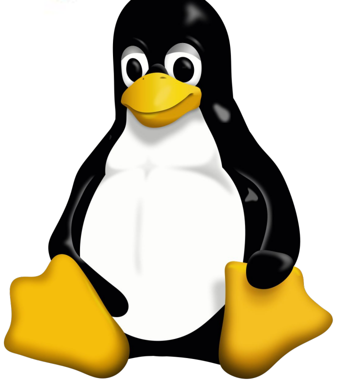
What is a computer system?

- Software and hardware systems
- A system comprises of many components
  - Components need to interact and cooperate well to provide the overall behaviour
  - Components typically have well specified interfaces
- Key goals in systems:
  - Performance/Scalability
  - Reliability/Availability
  - Usability/Generality
  - Security

Some famous systems contributions



http://blog.sqlauthority.com



Systems Area Overview

A non-exhaustive list of the subareas in systems:

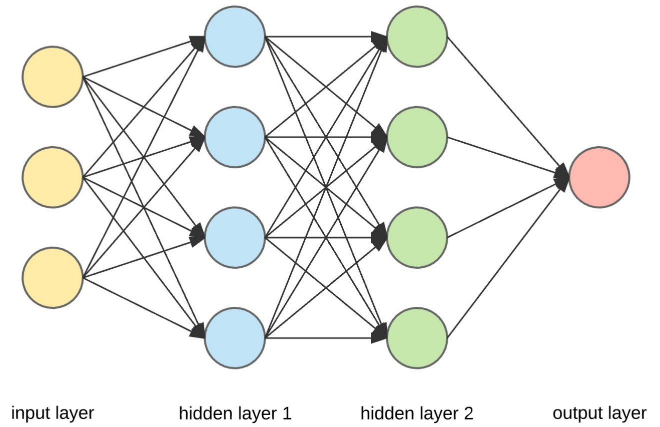
- Architecture
- Networking
- Security
- Distributed Systems
- Databases
- Operating Systems

Distributed Systems

- **Example**: Resilient Distributed Datasets: A Fault-Tolerant Abstraction for In-Memory Cluster Computing
- **Problem**: Frameworks such as MapReduce do not handle applications like iterative algorithms and interactive data mining tools efficiently, which *reuse* intermediate results across multiple computations.
- **Idea**: Keeping data in memory can greatly improve performances of such applications. RDD is an abstraction that is general enough to support a range of applications and can also provide fault tolerance efficiently.
- **Evaluation**:
  - Speedups on K-means, Logistics Regression, PageRank versus Hadoop:
  - Fault recovery
  - User applications

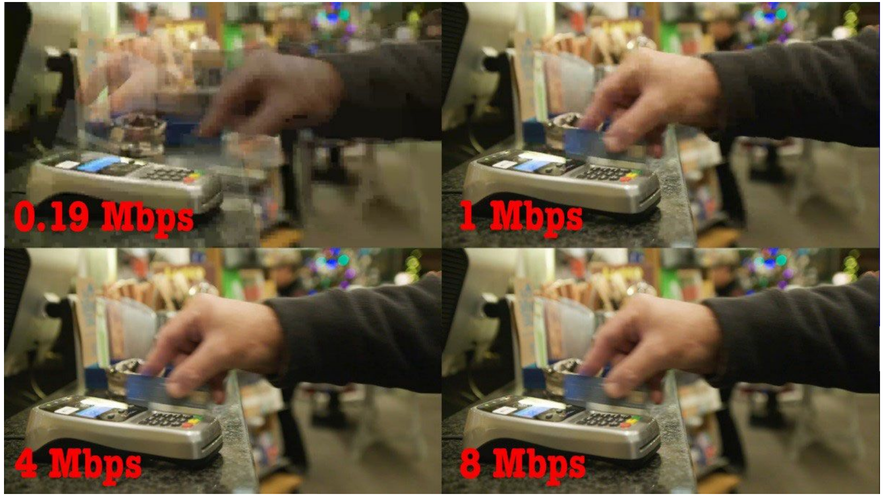
Architecture

- **Example**: In-Datacenter Performance Analysis of a Tensor Processing Unit
- **Problem**: How to design a specialized hardware to improve the cost-energy-performance of neural network inferences?
- **Idea**: Matrix Multiply Unit designed for dense matrices. The philosophy of the TPU microarchitecture is to keep the matrix unit busy.
- **Evaluation**:
  - Roofline analysis against CPUs and GPUs
  - Alternative TPU designs



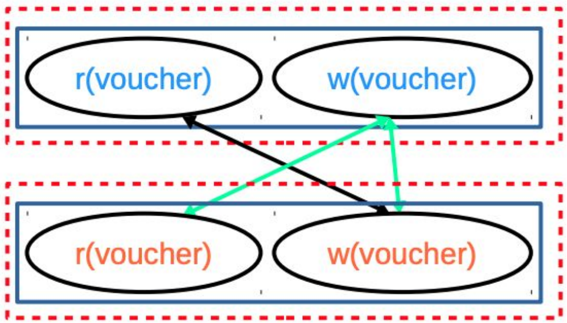
Networking

- **Example**: A Buffer-Based Approach to Rate Adaptation: Evidence from a Large Video Streaming Service
- **Problem**: How to dynamically choose the video bit rates to:
  - 1) maximizes the video quality by picking the highest video rate the network can support
  - 2) minimize rebuffering events which halts the video if the client's playback buffer goes empty.
- **Idea**: Choose the video rate based *only* on the playback buffer occupancy.
- **Evaluation**: Reduced the rebuffer rate by 10–20% compared to Netflix's then-default ABR algorithm.



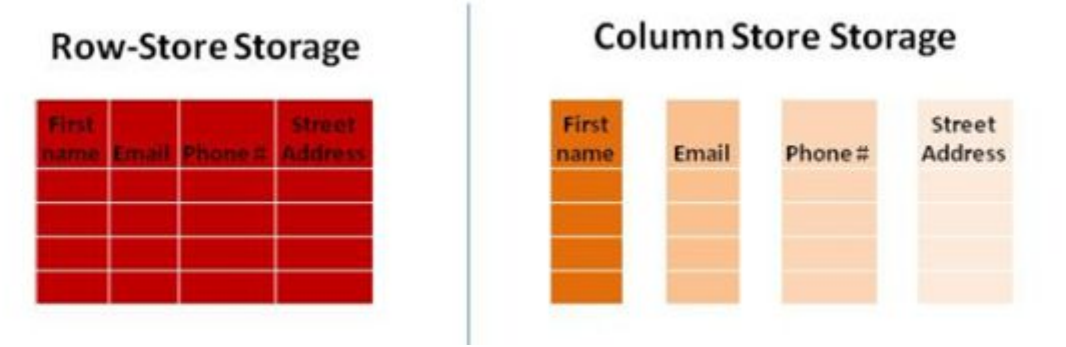
Security/Database

- **Example**: ACIDRain: Concurrency-Related Attacks on Database-Backed Web Applications
- **Attack**: Adversaries can exploit race condition to e.g. double spend vouchers.
- **Defense:** Use database logs to reconstruct transaction history, and detect cycles as potential anomaly
- **Evaluation**: Demonstrated vulnerabilities in 50% eCommerce site

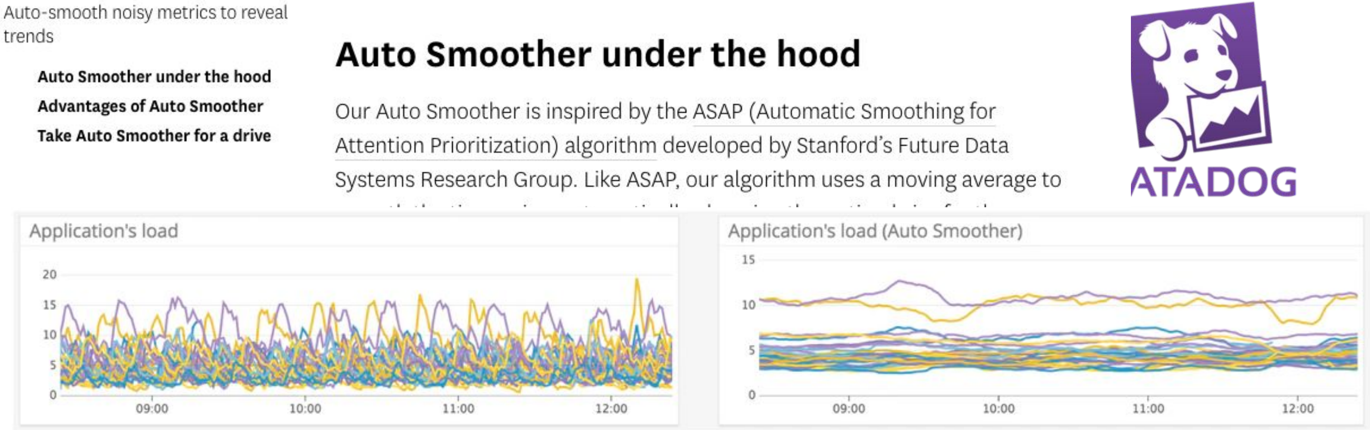


Database

- **Example**: C-Store: A Column-oriented DBMS
- **Problem**: Row-oriented databases are optimized for writes but not for reads
- **Idea:** Storage of data by column rather than by row
- **Evaluation**: Performance comparison on a number of queries



Introductions!



Auto-smooth noisy metrics to reveal trends

- Auto Smoother under the hood
- Advantages of Auto Smoother
- Take Auto Smoother for a drive

Auto Smoother under the hood

Our Auto Smoother is inspired by the ASAP (Automatic Smoothing for Attention Prioritization) algorithm developed by Stanford's Future Data Systems Research Group. Like ASAP, our algorithm uses a moving average to

It’s your turn!

Name

Year

Fun fact

What brings you here?

Anything else you’d like to share

Assignment 1 - due next Wednesday!

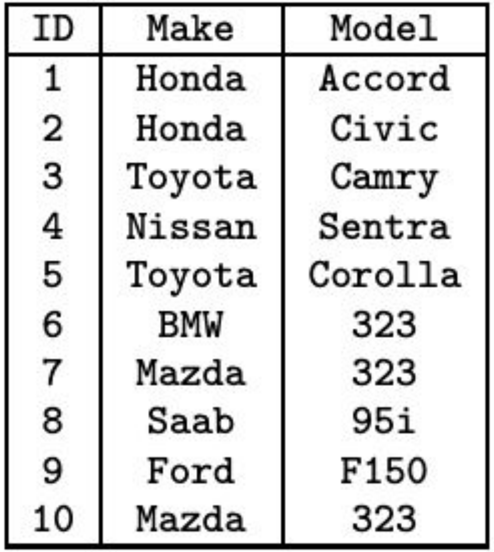
- Part 1: Read a paper and write an outline
- Part 2: Starter Task
  - Set up a Google cloud instance
    - Email instructions on how to request credits to follow
  - Play with git
  - Reproduce a benchmark
  - Produce a plot

Please enroll in the correct session!!

(My OH: Monday 9-10am @ Gates 433)

#1 Independence Assumption in Real Life

[CORDS: Automatic Discovery of Correlations and Soft Functional Dependencies]()



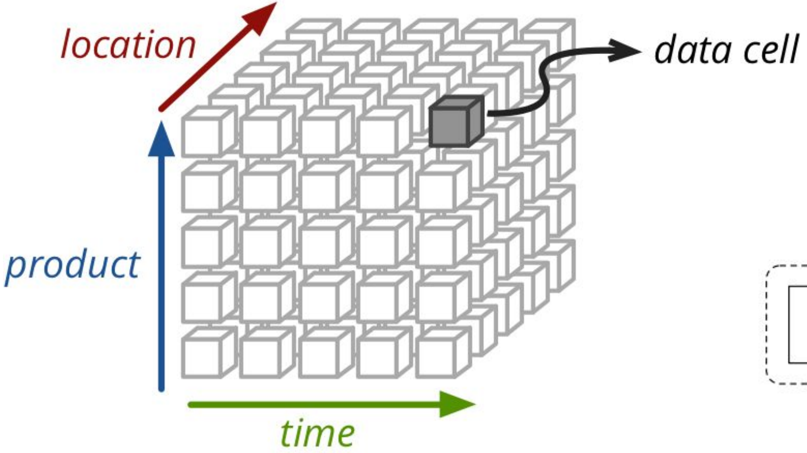
ID	Make	Model
1	Honda	Accord
2	Honda	Civic
3	Toyota	Camry
4	Nissan	Sentra
5	Toyota	Corolla
6	BMW	323
7	Mazda	323
8	Saab	95i
9	Ford	F150
10	Mazda	323

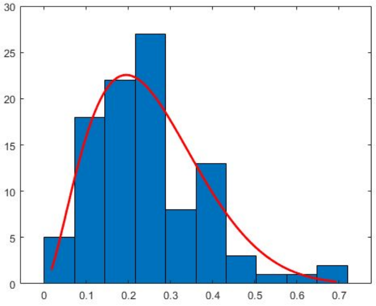
P[Make = “Honda”] = 1/7
P[Model = “Accord”] = 1/8

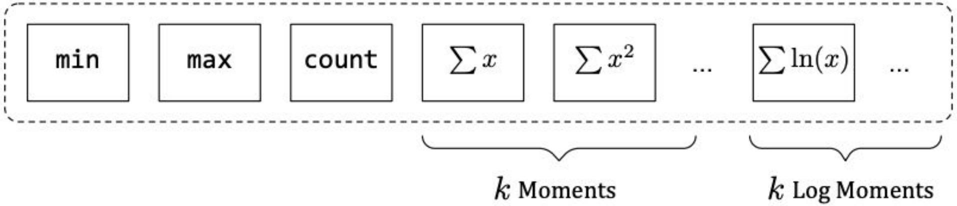
P[Make = “Honda” & Model = “Accord”] = ?

#2 Answering Queries with Metadata

Implementing Data Cubes Efficiently



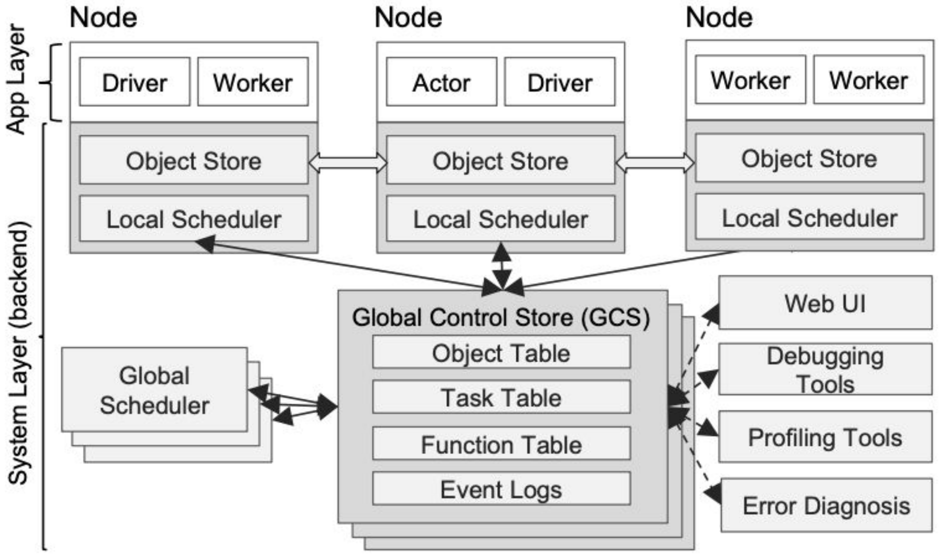




*Focus on main ideas, you don't need to understand the proofs.

#3 Designing Sketches in End-to-end Systems

Ray: A Distributed Framework for Emerging AI Applications

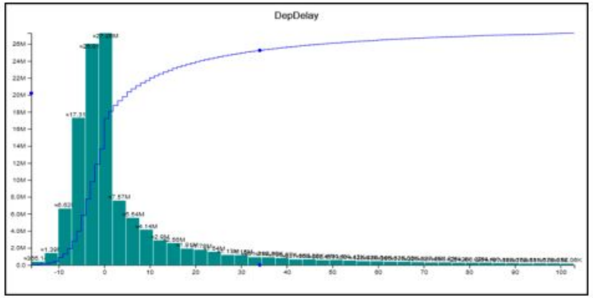


Also check out their project website for resources:

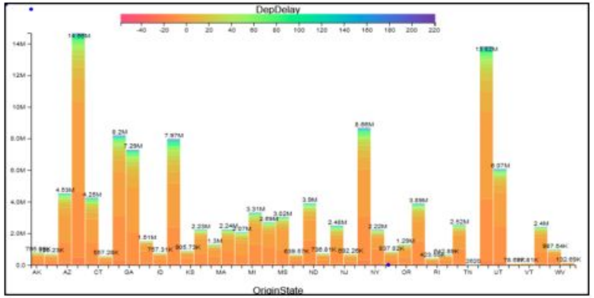
Code: https://github.com/ray-project/ray
Documentation:
http://ray.readthedocs.io/en/latest/index.html
Tutorial:
https://github.com/ray-project/tutorial
Blog: https://ray-project.github.io

#4 Sketches for Interactive Visualization Systems

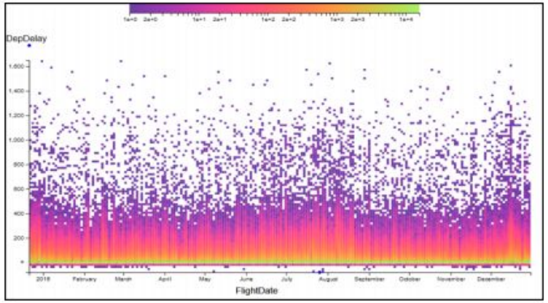
[Hillview: A trillion-cell spreadsheet for big data]()



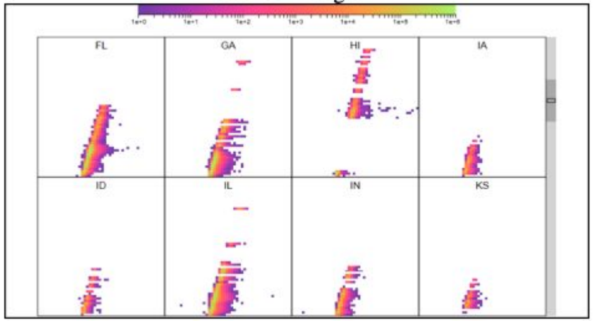
Histogram and CDF



Stacked histogram



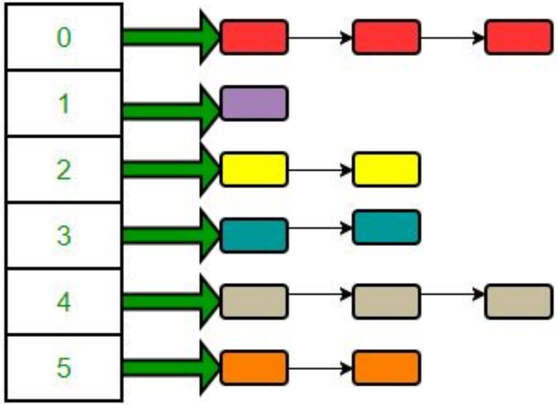
Heat map

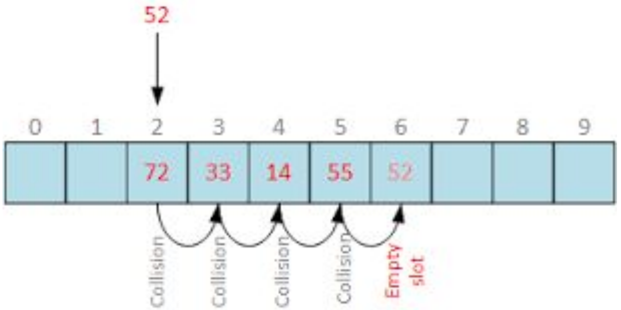


Trellis plot with heat maps

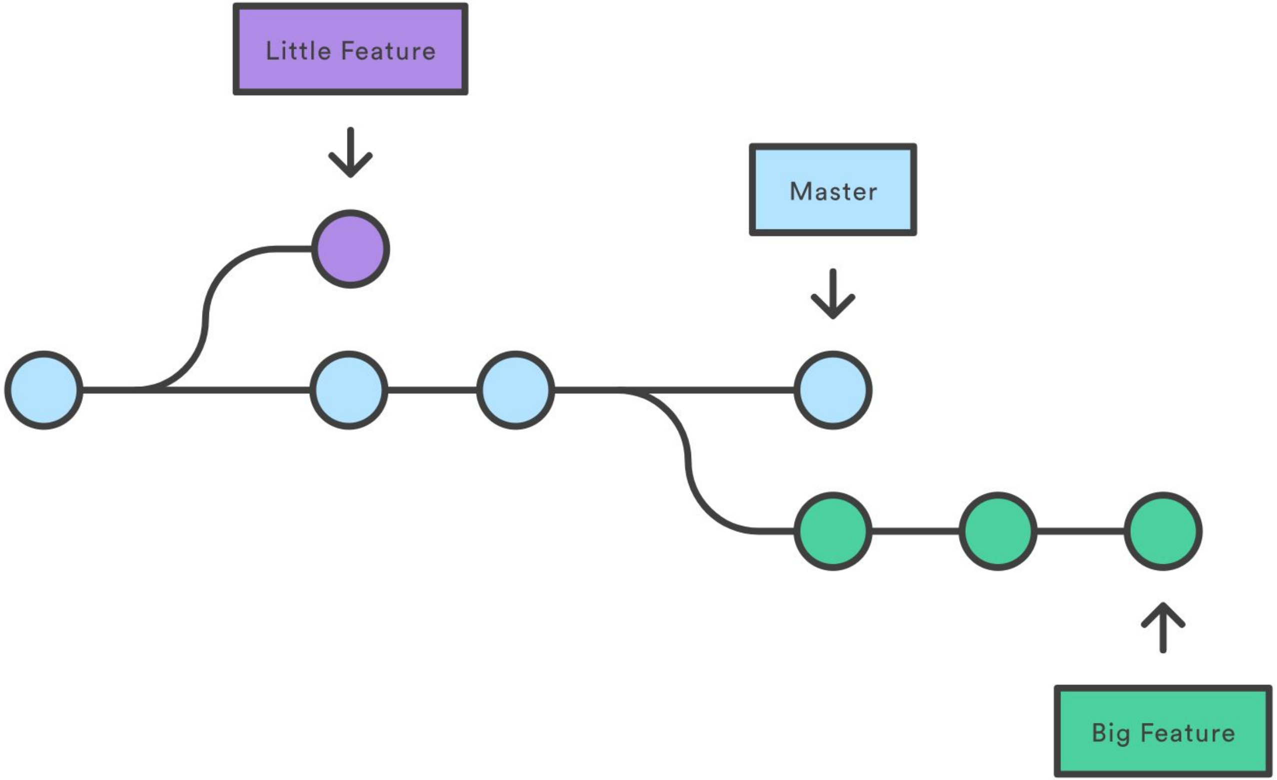
#5 Hash Table Bake off

A Seven-Dimensional Analysis of Hashing Methods and its Implications on Query Processing

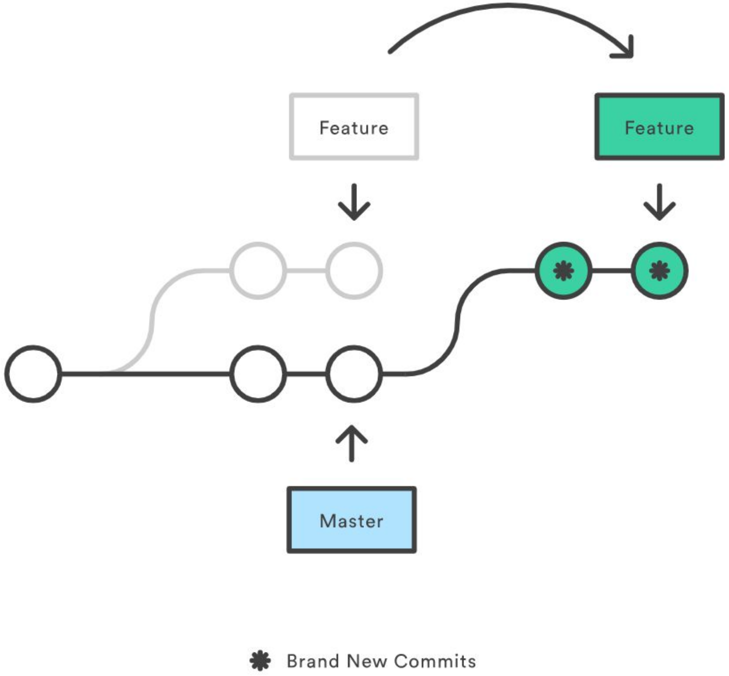


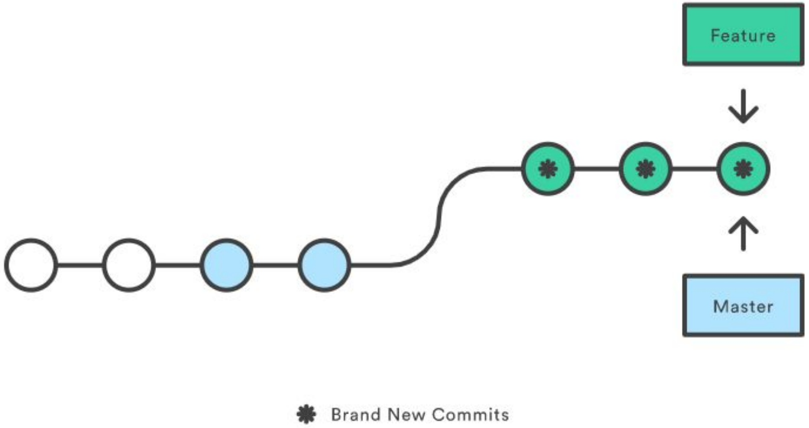


git branching

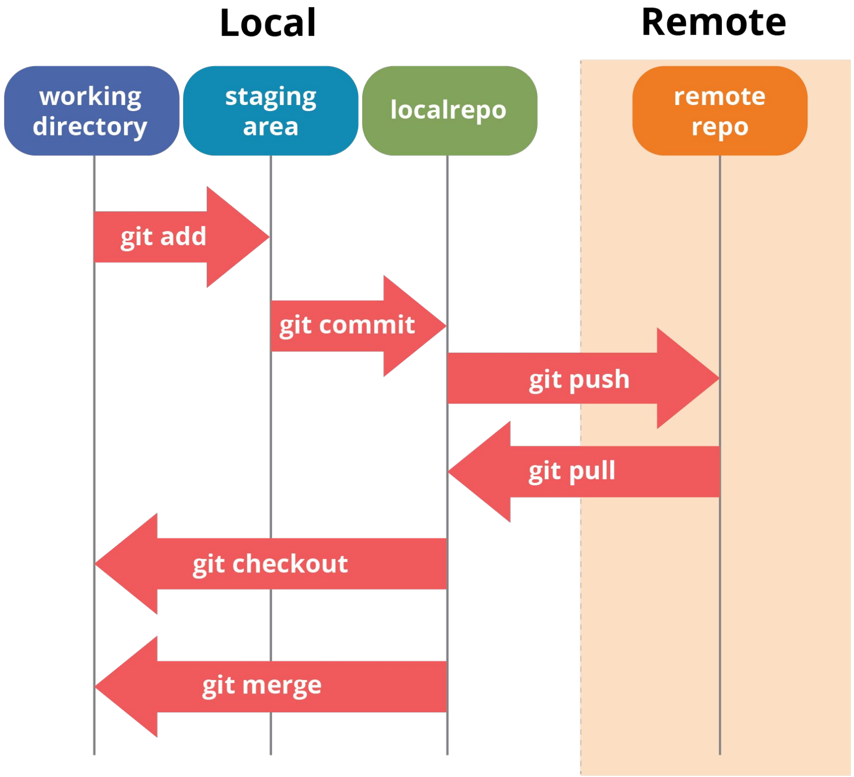


git rebase


Feature
Feature
Master
✱ Brand New Commits


Feature
Master
✱ Brand New Commits

Local versus remote


Local
Remote
working directory
staging area
localrepo
remote repo
git add
git commit
git push
git pull
git checkout
git merge