Data Mining - GitHub

Bruno Gonçalves
www.bgoncalves.com

Snowball Sampling

- Commonly used in Social Science and Computer Science

    1. Start with a single node (or small number of nodes)

    2. Get friends list

    3. For each friend get the friend list

    4. Repeat for a fixed number of layers or until enough users have been connected

- Generates a connected component from each seed

- Quickly generates a *lot* of data/API calls

Snowball Sampling

```
def get_followers(user_id, twitter_api=twitter_api):
    cursor = -1
    followers = []

    while cursor != 0:
        result = twitter_api.followers.ids(user_id=user_id, cursor=cursor)

        followers += result["ids"]
        cursor = result["next_cursor"]

    return followers
```

Snowball Sampling

```
def snowball(user_id, max_calls=10, twitter_api=twitter_api):
    seen = set()
    queue = set()

    queue.add(user_id)

    G = NX.Graph()
    call_count = 0

    while queue:
        user_id = queue.pop()
        seen.add(user_id)

        followers = get_followers(user_id)
        call_count += 1

        for follower in followers:
            if follower not in seen:
                queue.add(follower)

            G.add_edge(user_id, follower)

        if call_count > max_calls:
            break

    return G
```

Co-Occurrence Networks

- Connect two items whenever they co-occur
- Patterns on the resulting network structure reveals correlations
- Communities reveal clusters in data

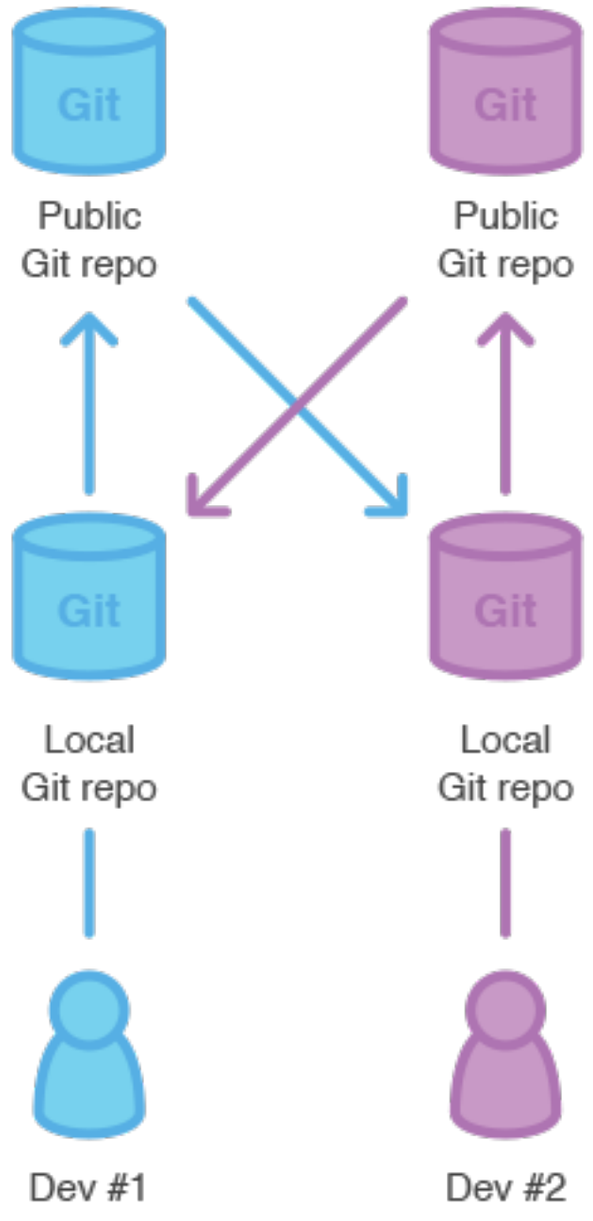
- **Challenge** - Generate the #tag co-occurence network from a stream search for "music"

Co-Occurrence Networks

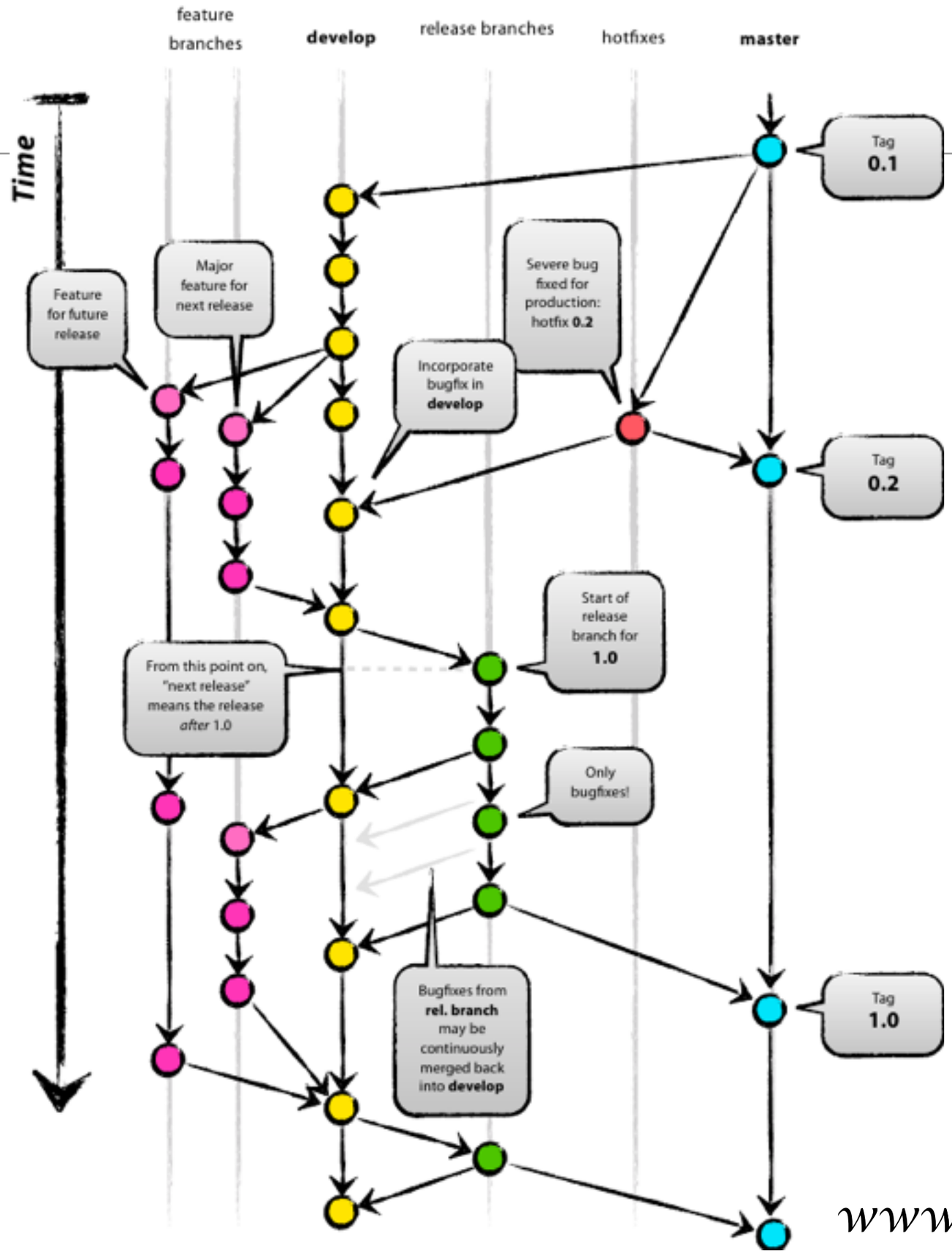
```
import twitter
from twitter_accounts import accounts
import networkx as NX
import itertools

app = accounts["social"]

auth = twitter.oauth.OAuth(app["token"],app["token_secret"], app["api_key"],
app["api_secret"])

stream_api = twitter.TwitterStream(auth=auth)
stream_results = stream_api.statuses.filter(track = “music”)
G = NX.Graph()

try:
    for tweet in stream_results:
        if len(tweet["entities"]["hashtags"]) > 1:
            tags = set([tag["text"].lower() for tag in tweet["entities"]["hashtags"]])
            G.add_edges_from(itertools.combinations(tags, 2))
            print tags
except KeyboardInterrupt:
    pass

print "Found", G.number_of_nodes(), "#tags"
print "\n".join(G.nodes())
```

git

- Created by Linus Torvalds in 2005
- Distributed Source Code Management System
- "the stupid content tracker"
- Extremely efficient and popular
- Self hosted in 3 days
- Built to handle linux kernel development
- Designed for remote (offline) collaboration



Git branches



torvalds / linux

Linux kernel source tree

495,037 commits **1** branch **402** releases **4,528** contributors

branch: **master** linux / +

Merge branch 'leds-fixes-for-3.19' of git://git.kernel.org/pub/scm/li… ···

torvalds authored 21 hours ago latest commit 188c901941

Documentation	Merge git://git.kernel.org/pub/scm/linux/kernel/git/nab/target-pending	2 days ago
arch	Merge tag 'stable/for-linus-3.19-rc4-tag' of git://git.kernel.org/pub…	a day ago
block	Revert "blk-mq: Micro-optimize bt_get()"	a month ago
crypto	crypto: af_alg - fix backlog handling	23 days ago
drivers	Merge branch 'leds-fixes-for-3.19' of git://git.kernel.org/pub/scm/li…	21 hours ago
firmware	kbuild: remove obj-n and lib-n handling	3 months ago
fs	Merge branch 'sched-urgent-for-linus' of git://git.kernel.org/pub/scm…	3 days ago
include	Merge tag 'for-linus' of git://git.kernel.org/pub/scm/linux/kernel/gi…	21 hours ago
init	init: fix read-write root mount	28 days ago
ipc	Merge branch 'for-linus' of git://git.kernel.org/pub/scm/linux/kernel…	29 days ago
kernel	Merge branch 'sched-urgent-for-linus' of git://git.kernel.org/pub/scm…	3 days ago
lib	Merge tag 'for_linus-3.19-rc4' of git://git.kernel.org/pub/scm/linux/…	5 days ago
mm	mm: mmu_gather: use tlb->end != 0 only for TLB invalidation	2 days ago

Code
Pull Requests 61
Pulse
Graphs

HTTPS clone URL

https://github.com

You can clone with HTTPS or Subversion.

Download ZIP

GitHub

- Most popular Git repository hosting service
- Web based
- Launched April 2008
- Adds functionality on top of Git
- "Gists" - Version controlled paste-bins
- Users create accounts and are able to comment, fork, clone, etc…

GitHub Developer

https://developer.github.com

GitHub Developer

API Developers Blog Support Search

Leverage the power of GitHub in your app.

Get started with one of our guides, or jump straight into the API documentation.

Browse the documentation

Join the GitHub Developer Program.

The best way to integrate with GitHub. Learn more.

Get Started

New to the GitHub API? With these guides you'll be up and running in a snap.

Libraries

We've got you covered. Use the GitHub API in your favorite language.

Support

Are you stuck? Already tried our troubleshooting guide? Talk to a supportocat.

Authorized applications

https://github.com/settings/applications

Search GitHub | Explore | Gist | Blog | Help | bmtgoncalves

Personal settings

- Profile
- Account settings
- Emails
- Notification center
- Billing
- SSH keys
- Security
- **Applications**
- Repositories
- Organizations

Developer applications

Register new application

These are applications you have registered to use the GitHub API.

Data Mining Application http://www.bgoncalves.com — **0** users

Personal access tokens

Generate new token

Tokens you have generated that can be used to access the GitHub API.

Testing — *gist, repo, user* — Last used on Jan 20, 2015 — Edit | Delete

Personal access tokens function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to authenticate to the API over Basic Authentication.

Authorized applications

You have no applications authorized to access your account.

GitHub applications

These are applications developed and owned by GitHub, Inc. They have full access to your GitHub account.

Gist — Last used on Jun 16, 2014 — Revoke

New personal access toker ×

https://github.com/settings/tokens/new

Search GitHub | Explore | Gist | Blog | Help | bmtgoncalves

Personal settings

- Profile
- Account settings
- Emails
- Notification center
- Billing
- SSH keys
- Security
- **Applications**
- Repositories
- Organizations

Applications / **New personal access token**

Token description

Testing

What's this token for?

Select scopes

Scopes *limit* access for personal tokens. Read more about OAuth scopes.

- [x] **repo**
- [x] **public_repo**
- [ ] **user:email**
- [ ] **write:org**
- [ ] **write:public_key**
- [ ] **write:repo_hook**
- [x] **gist**
- [ ] **repo:status**
- [ ] **delete_repo**
- [ ] **user:follow**
- [ ] **read:org**
- [ ] **read:public_key**
- [ ] **read:repo_hook**
- [ ] **notifications**
- [ ] **repo_deployment**
- [x] **user**
- [ ] **admin:org**
- [ ] **admin:public_key**
- [ ] **admin:repo_hook**
- [ ] **admin:org_hook**

Generate token

Personal access tokens function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to authenticate to the API over Basic Authentication.

Authorized applications

https://github.com/settings/applications#personal-access-tokens

Search GitHub

Explore Gist Blog Help

bmtgoncalves

Personal settings

- Profile
- Account settings
- Emails
- Notification center
- Billing
- SSH keys
- Security
- **Applications**
- Repositories
- Organizations

Developer applications

Register new application

These are applications you have registered to use the GitHub API.

Data Mining Application http://www.bgoncalves.com — **0** users

Personal access tokens

Generate new token

Tokens you have generated that can be used to access the GitHub API.

Make sure to copy your new personal access token now. You won't be able to see it again!

Edit Delete

Personal access tokens function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to authenticate to the API over Basic Authentication.

Authorized applications

You have no applications authorized to access your account.

GitHub applications

These are applications developed and owned by GitHub, Inc. They have full access to your GitHub account.

Gist — Last used on Jun 16, 2014 — Revoke

API Basics

https://developer.github.com/v3/

- The github module provides all the necessary interface. We just need to provide the right client token.
- As before, best to keep the credentials in a dict and parametrize our calls with the dict key. This way we can switch between different accounts easily.
- .Github(token, per_page=100) takes a client token as argument and returns a Github object that we can use to interact with the API
- 3 basic types of objects:
  - Users
  - Repositories
  - Commits
- The python library we are using works a bit differently from the Twitter one we saw before.
- API calls return objects
- Data is stored as properties instead of dictionary fields

Authenticating with the API

https://developer.github.com/v3/

```
import github

accounts = {
    "social" : TOKEN,
}

token = accounts["social"]

client = github.Github(token, per_page=100)
```

Users

https://developer.github.com/v3/users/

- .get_user(user) returns the NamedUser object corresponding to user
- NamedUser objects contain many properties with information about the user:
  - .id - User ID
  - .name - User name
  - .email - Email address (not always visible)
  - .followers / .following - Number of people follow user / are followed by user
  - .repos - Number of public repositories
- Some properties have an associated get_*() function that returns an iterator over its elements:
  - .get_followers() / .get_following(), .get_repos()

Challenge

- Print the name of the first 10 followers of user:

torvalds

```
import github
from github_accounts import accounts

token = accounts["social"]

client = github.Github(token, per_page=100)

screen_name = "torvalds"

user = client.get_user(screen_name)

follow_count = 0

for follow in user.get_followers():
    print follow_count, follow.name

    follow_count += 1

    if follow_count == 10:
        break
```

Repos

https://developer.github.com/v3/repos/

- Multiple users can have repositories with the same name
- Must ask for a repository belonging to a specific NamedUser instance.
- .get_repo(repo) returns a Repository object with a similar structure to NamedUser objects (properties and methods)
  - .fork - Whether or not it is a fork
  - .forks - How many times it was forked (.get_forks())
  - .get_commits() - returns iterator over all Commit objects
  - .get_commit(commit_sha) - return a specific Commit object
  - .stargazers_count - How many times it was stared (.get_stargazers())

Challenge

- Get the name of the first 10 stargazers for repo

linux

- of user

torvalds

Challenge

- Get the name of the first 10 stargazers for repo

linux

- of user

```python
import github
from github_accounts import accounts

token = accounts["social"]

client = github.Github(token, per_page=100)

screen_name = "torvalds"
repository_name = "linux"

user = client.get_user(screen_name)
repo = user.get_repo(repository_name)

user_count = 0

for user in repo.get_stargazers():
    print user_count, user.name

    user_count += 1

    if user_count == 10:
        break
```

Commits

https://developer.github.com/v3/repos/commits/

- Commit objects correspond to one of the most fundamental concepts, a change in a piece of code
- Commits belong to one branch and are logically part of a DAG with one or more parents, siblings and children
    - More than one parent indicates that this commit is a merge of multiple previous commits
- Each commit is identified by a cryptographic sha id and refers to its parents

Challenge

- Build the commit DAG of repository:

Mining-the-Social-Web

- belonging to user:

ptwobrussell

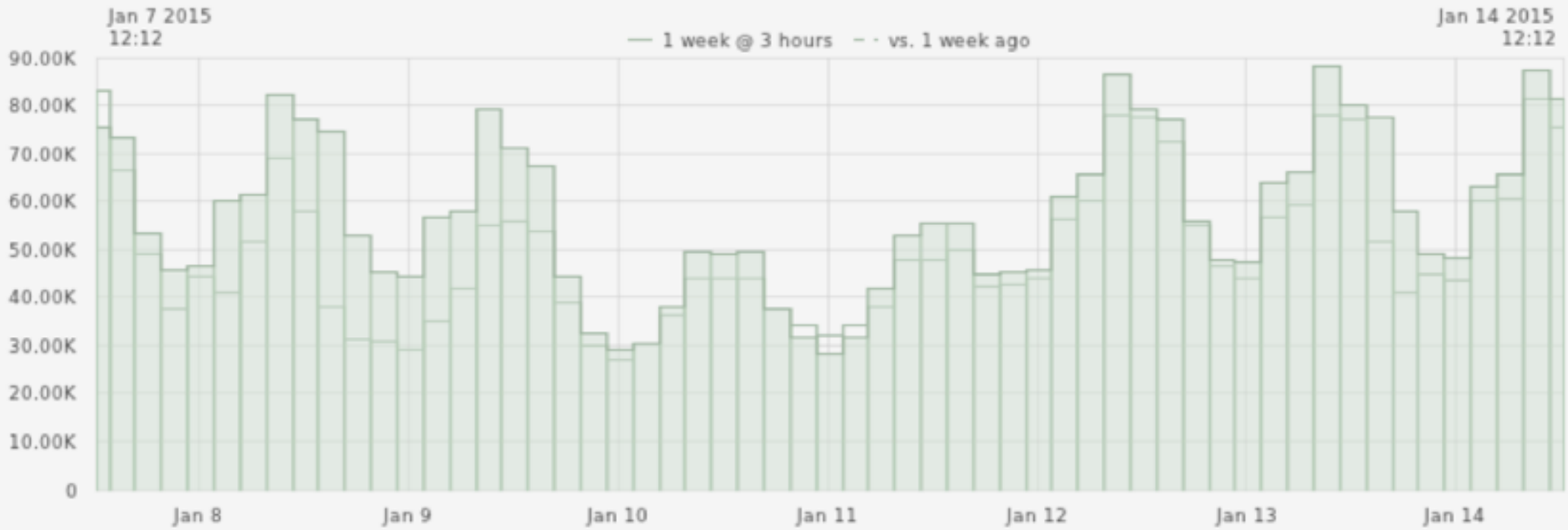
- and print the total number of nodes and edges

Challenge

- Build the commit DAG of repository:
- belonging to user:
- and print the total number

```
import github
from github_accounts import accounts
import networkx as NX

token = accounts["social"]

client = github.Github(token, per_page=100)

screen_name = "ptwobrussell"
repository_name = "Mining-the-Social-Web"

user = client.get_user(screen_name)
repo = user.get_repo(repository_name)

G = NX.DiGraph()

for commit in repo.get_commits():
    for parent in commit.parents:
        G.add_edge(parent.sha, commit.sha)

print G.number_of_nodes(), G.number_of_edges()
```

GitHub Archive



Open-source developers all over the world are working on millions of projects: writing code & documentation, fixing & submitting bugs, and so forth. GitHub Archive is a project to **record** the public GitHub timeline, **archive it**, and **make it easily accessible** for further analysis.

GitHub provides 20+ event types, which range from new commits and fork events, to opening new tickets, commenting, and adding members to a project. These events are aggregated into hourly archives, which you can access with any HTTP client:

Query	Command
Activity for 1/1/2015 @ 3PM UTC	`wget http://data.githubarchive.org/2015-01-01-15.json.gz`
Activity for 1/1/2015	`wget http://data.githubarchive.org/2015-01-01-{0..23}.json.gz`

Event Types

i. CommitCommentEvent
ii. CreateEvent
iii. DeleteEvent
iv. DeploymentEvent
v. DeploymentStatusEvent
vi. DownloadEvent
vii. FollowEvent
viii. ForkEvent
ix. ForkApplyEvent
x. GistEvent
xi. GollumEvent
xii. IssueCommentEvent
xiii. IssuesEvent
xiv. MemberEvent
xv. MembershipEvent
xvi. PageBuildEvent
xvii. PublicEvent
xviii. PullRequestEvent
xix. PullRequestReviewCommentEvent
xx. PushEvent
xxi. ReleaseEvent
xxii. RepositoryEvent
xxiii. StatusEvent
xxiv. TeamAddEvent
xxv. WatchEvent

Event Structure

https://developer.github.com/v3/activity/events/types

- Events Contain information about:
  - Repository ("repo")
  - Actor
  - Type
  - ID
  - Payload - Event specific information

Create Event

https://developer.github.com/v3/activity/events/types

- Triggered when a repo, branch or tag is created

- Payload includes:

  - "ref_type" - type of structure created (repository, branch, tag)

  - "description"

ForkEvent

https://developer.github.com/v3/activity/events/types

- Triggered when a repository is forked

- Payload contains information about repository that was created (“forkee”)

Challenge

- From the file

http://data.githubarchive.org/2015-01-01-15.json.gz

- that you downloaded in the first class, list the "full_name" of all "forkee" and created repos

```
import gzip
import json

filename = "2015-01-01-15.json.gz"

for line in gzip.open(filename):
    event = json.loads(line.strip())

    if event["type"] == "CreateEvent":
        print "Create", event["repo"]["name"]
    elif event["type"] == "ForkEvent":
        print "Fork", event["payload"]["forkee"]["full_name"]
```

PushEvent

https://developer.github.com/v3/activity/events/types

- Triggered when commits are pushed to a branch
- Payload includes:
- List of “commits” with “sha” ids, “message” and “author” information
    - ids can then be used with the regular API to obtain more information about the commit or author
- as “head” the sha reference of the head of the branch that is being pushed to

Challenge

- From the file

http://data.githubarchive.org/2015-01-01-15.json.gz

- list all the commit ids ("sha") that were pushed along with it parent (you can get information on the parent by querying the API)

Challenge

- From th
- list all th
  informat

```
import gzip
import json
import github
from github_accounts import accounts
import sys

token = accounts["social"]

client = github.Github(token, per_page=100)

filename = "2015-01-01-15.json.gz"

for line in gzip.open(filename):
    event = json.loads(line.strip())

    if event["type"] == "PushEvent":
        user_name, repo_name = event["repo"]["name"].split('/')

        try:
            user = client.get_user(user_name)
            repo = user.get_repo(repo_name)

            for commit in event["payload"]["commits"]:
                commit_id = commit["sha"]

                commit = repo.get_commit(commit_id)

                for parent in commit.parents:
                    print commit_id, parent.sha
        except Exception, e:
            print >> sys.stderr, "Error processing", event["repo"]["name"], e.status
```

MemberEvent

https://developer.github.com/v3/activity/events/types

- Triggered when a new member is added to a repository
- There is no event for when a developer is removed from a repository
- Payload includes basic information about member added
- Actor is the developer who is adding the new member to the team

Challenge

- Build a social network based on "MemberEvent"s

```
import gzip
import json
import networkx as NX

filename = "2015-01-01-15.json.gz"

G = NX.DiGraph()

for line in gzip.open(filename):
    event = json.loads(line.strip())

    if event["type"] == "MemberEvent":
        actor = event["actor"]["login"]
        member = event["payload"]["member"]["login"]

        G.add_edge(actor, member)

print G.number_of_nodes(), G.number_of_edges()
```