Combinatorics of Git Graphs

Julien COURTIEL
(Université de Caen Normandie)

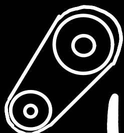
Work 1
with Paul DORBEC
Romain LECOQ
(Université de Caen Normandie)

Work 2
with Martin PEPIN
(Université de Caen Normandie)

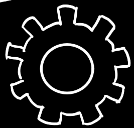
Git de France

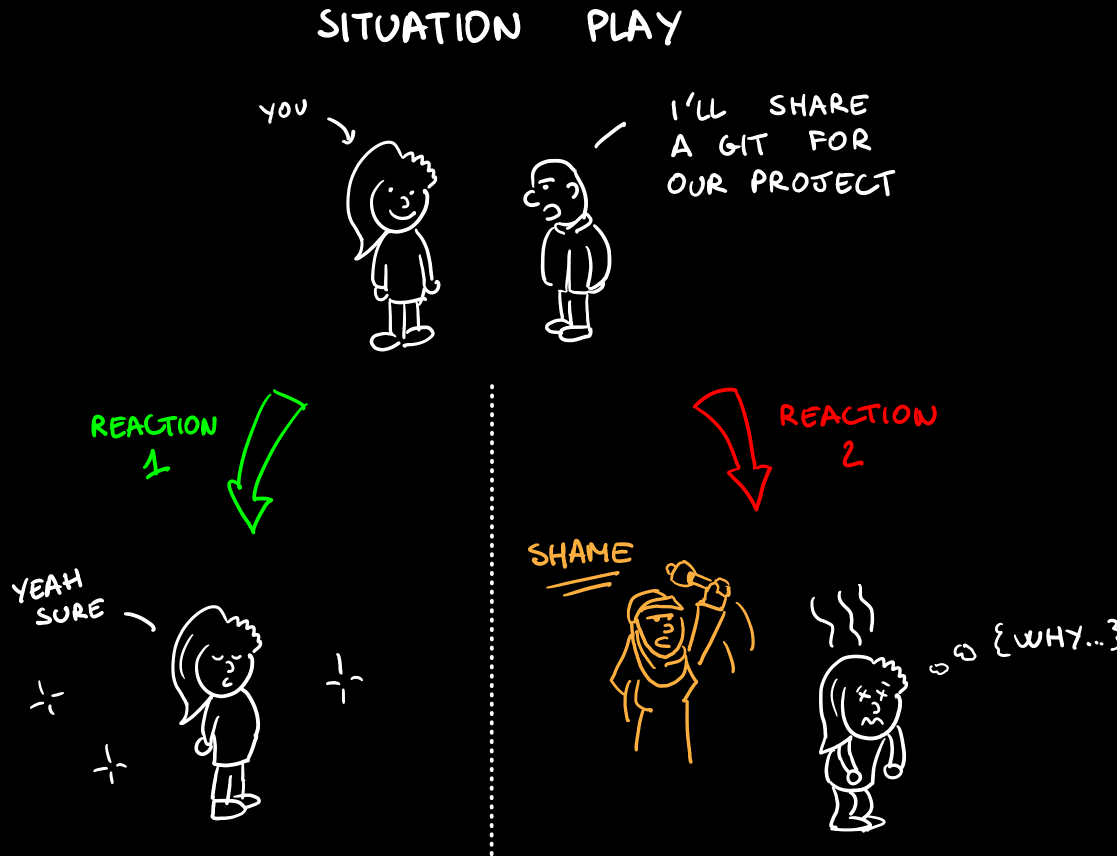
ALEA Days 2025

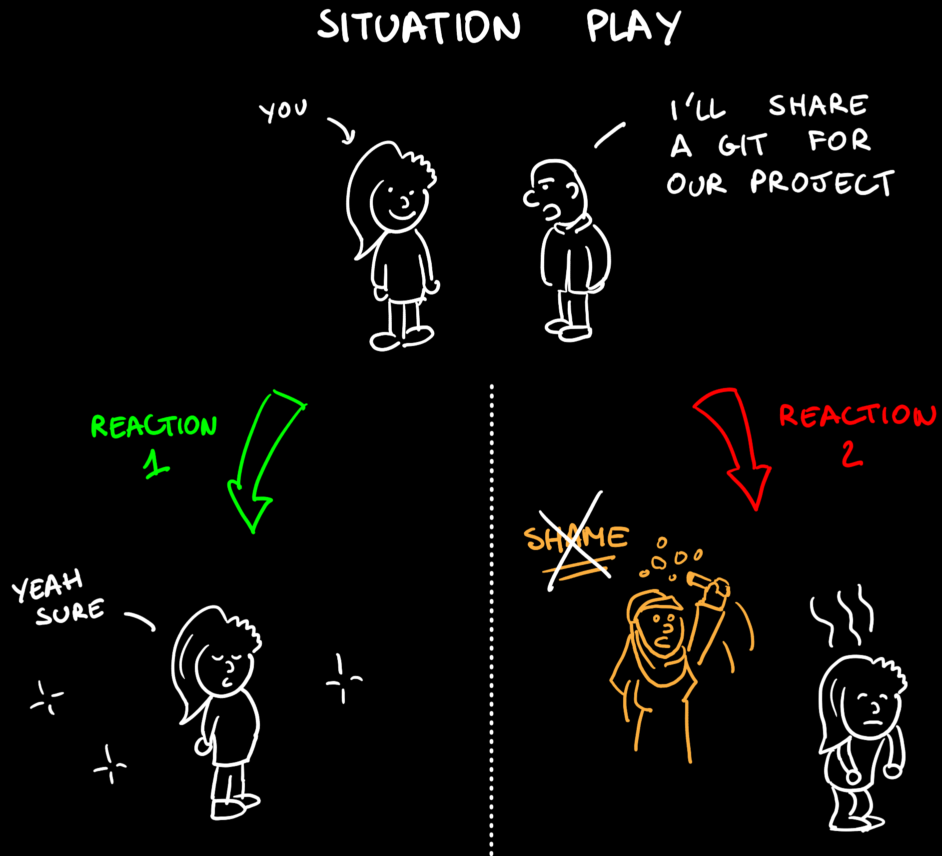
INTRODUCTION



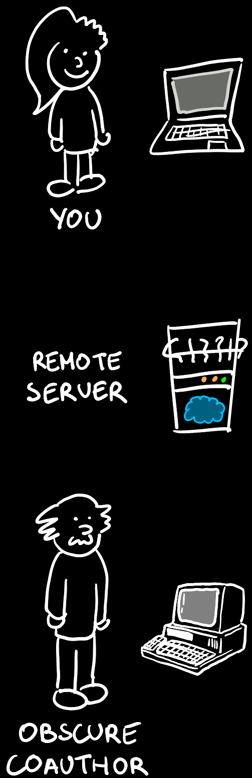
HOW GIT WORKS




SITUATION PLAY
YOU
I'LL SHARE A GIT FOR OUR PROJECT
REACTION 1
YEAH SURE
REACTION 2
SHAME
{WHY...}


SITUATION PLAY
YOU
I'LL SHARE A GIT FOR OUR PROJECT
REACTION 1
REACTION 2
YEAH SURE
SHAME

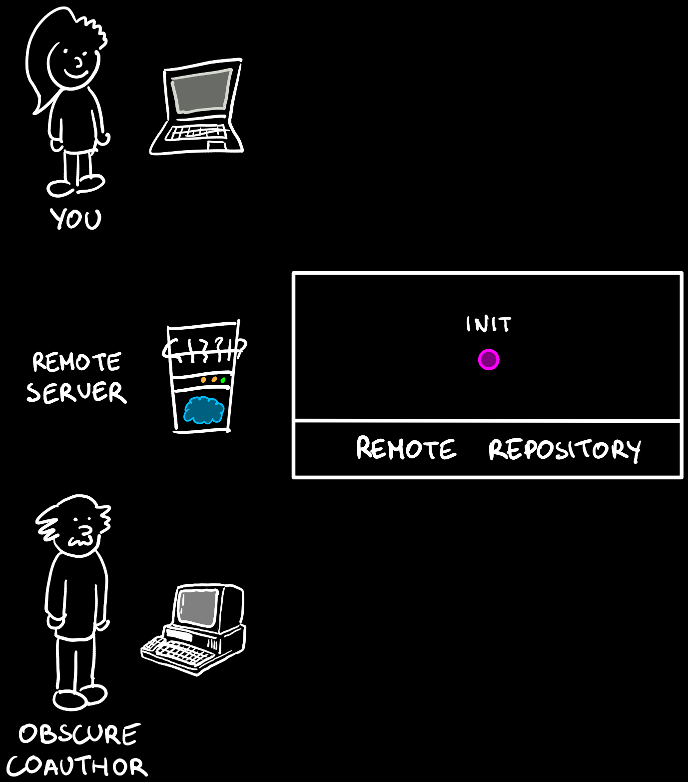
WORKING TOGETHER ON AN ARTICLE WITH GIT



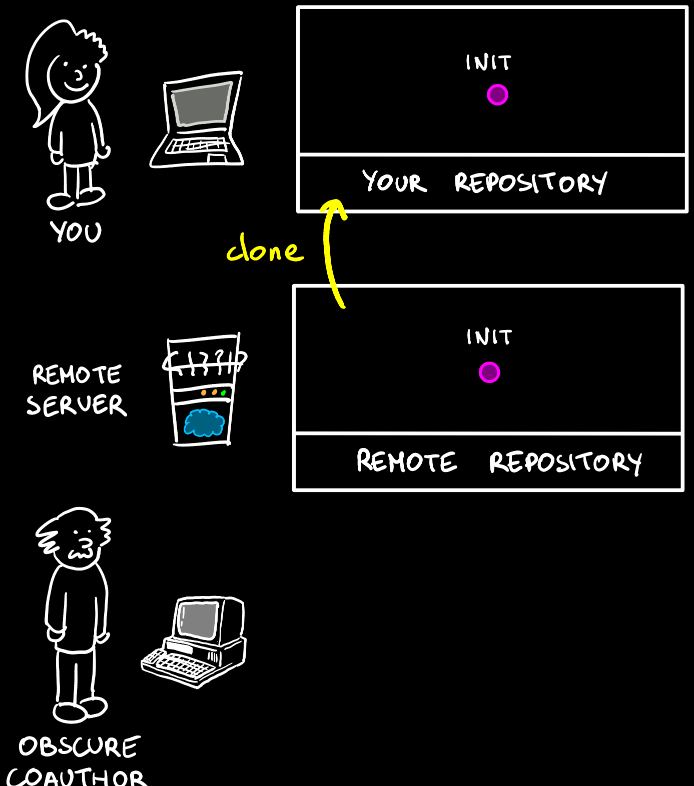
Situation play 2

You write an article with an obscure coauthor

WORKING TOGETHER ON AN ARTICLE WITH GIT

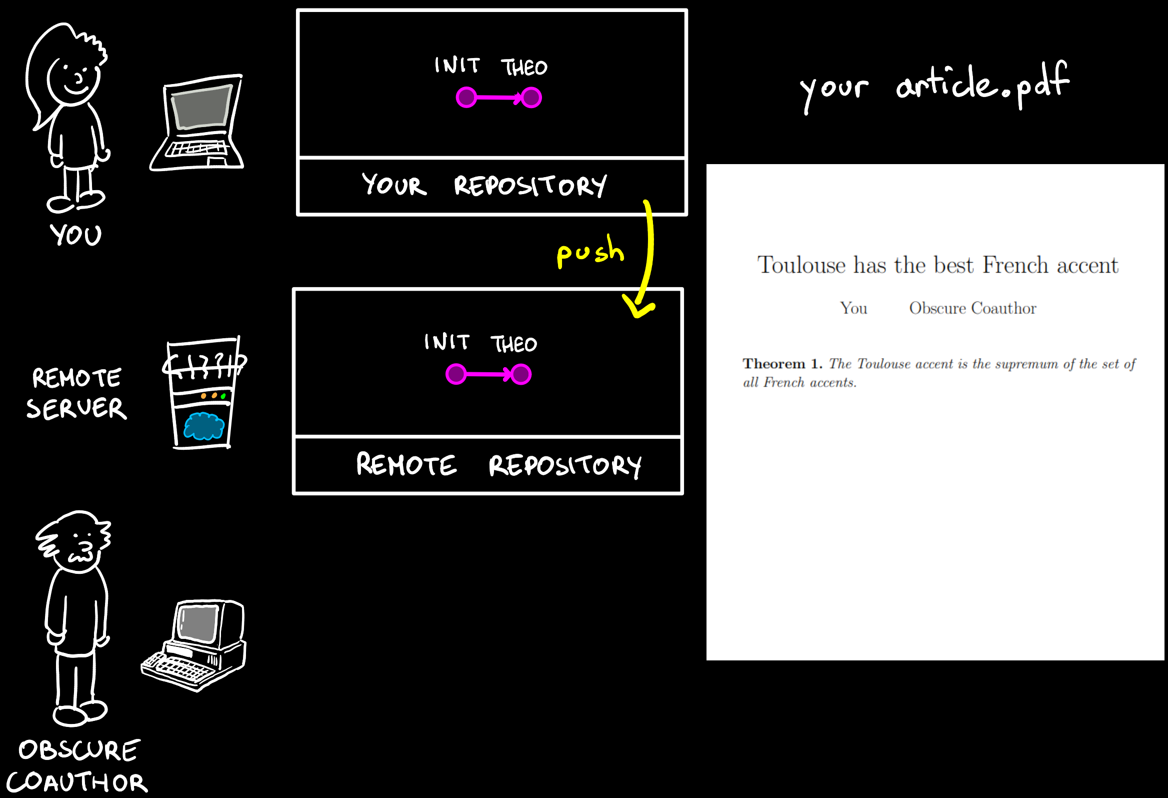


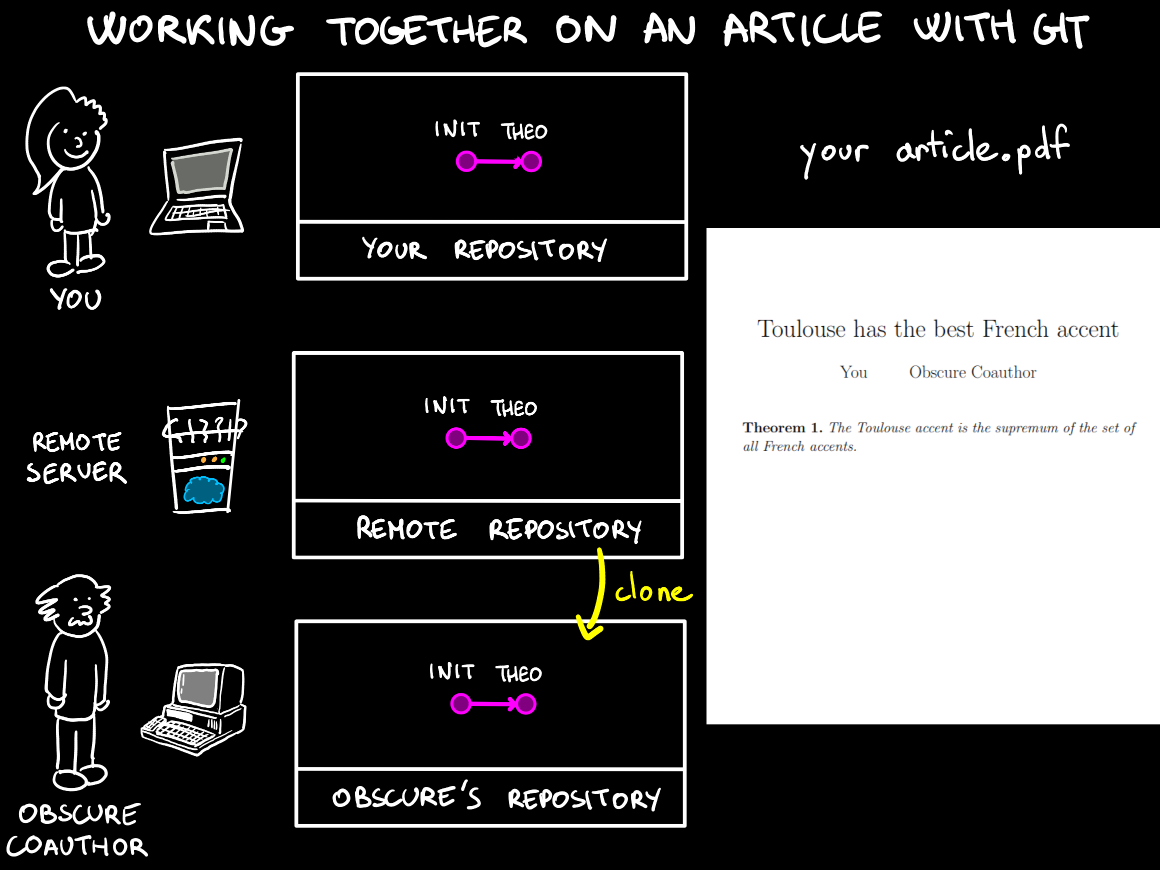
WORKING TOGETHER ON AN ARTICLE WITH GIT

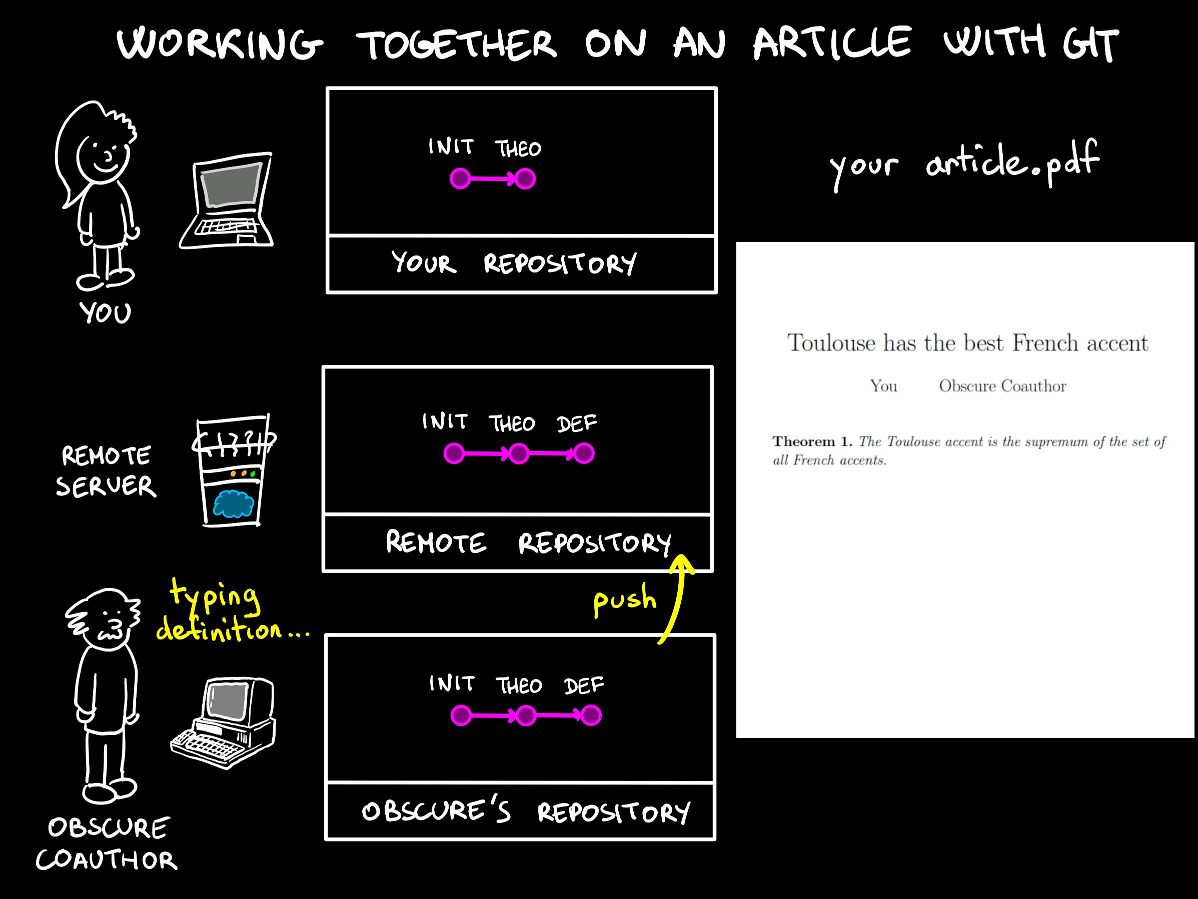


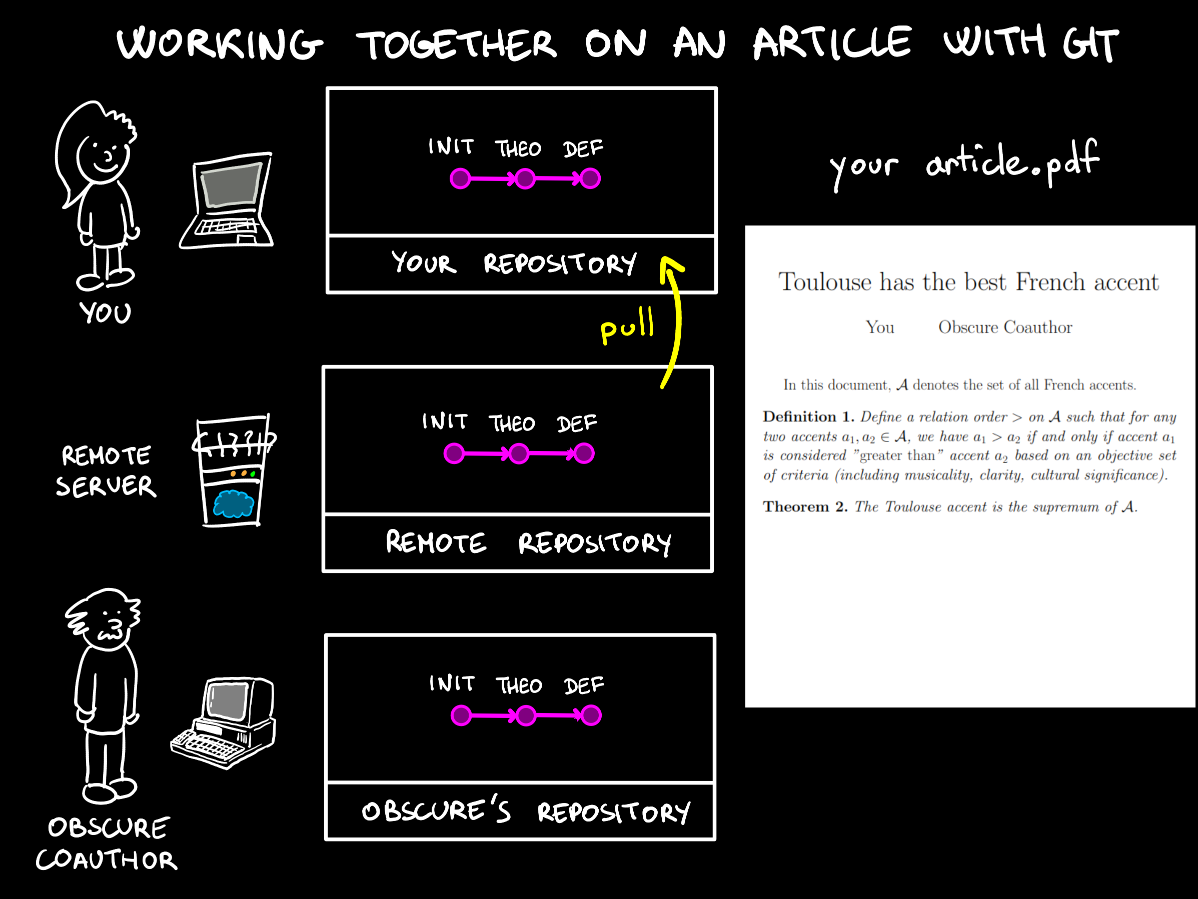
WORKING TOGETHER ON AN ARTICLE WITH GIT

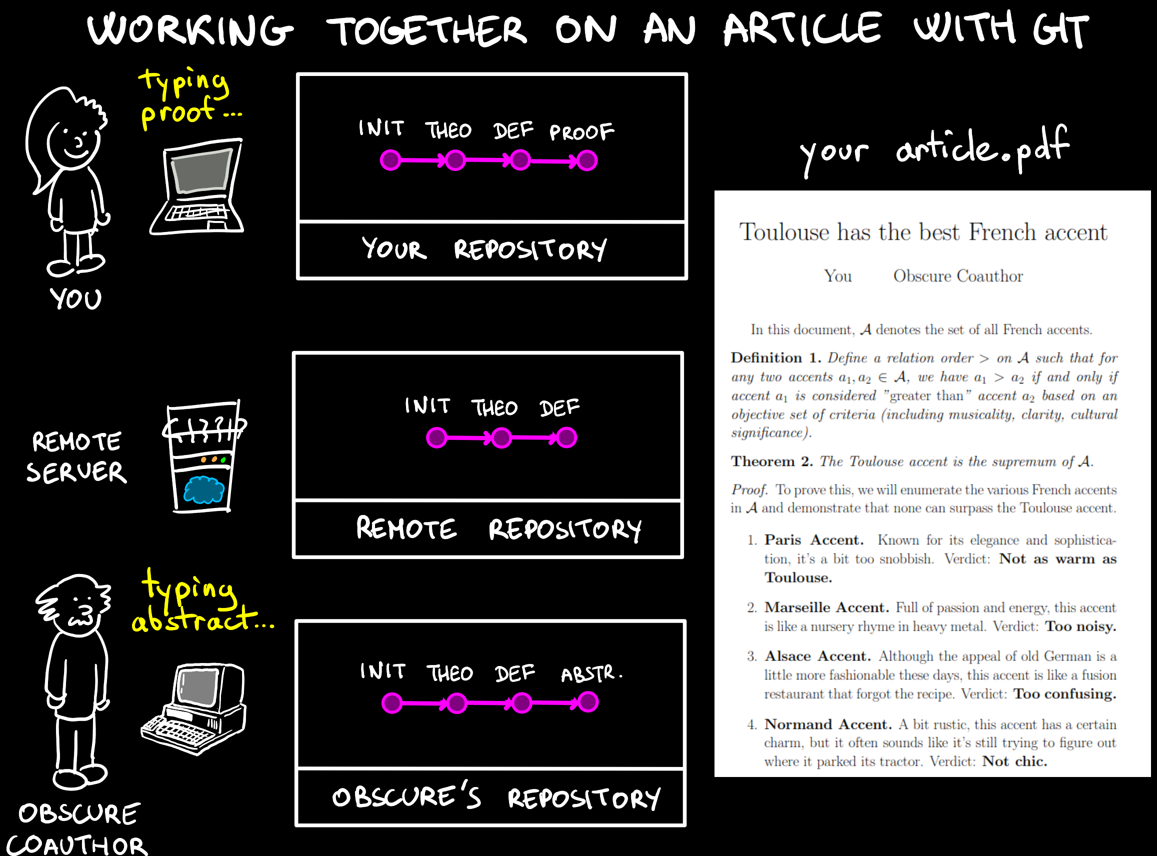
WORKING TOGETHER ON AN ARTICLE WITH GIT




WORKING TOGETHER ON AN ARTICLE WITH GIT
YOU
INIT
THEO
YOUR REPOSITORY
REMOTE SERVER
INIT
THEO
REMOTE REPOSITORY
clone
OBSCURE COAUTHOR
INIT
THEO
OBSCURE'S REPOSITORY
your article.pdf
Toulouse has the best French accent
You
Obscure Coauthor
Theorem 1. The Toulouse accent is the supremum of the set of all French accents.


WORKING TOGETHER ON AN ARTICLE WITH GIT
YOU
INIT THEO
YOUR REPOSITORY
your article.pdf
Toulouse has the best French accent
You Obscure Coauthor
Theorem 1. *The Toulouse accent is the supremum of the set of all French accents.*
REMOTE SERVER
INIT THEO DEF
REMOTE REPOSITORY
push
typing definition...
INIT THEO DEF
OBSCURE'S REPOSITORY
OBSCURE COAUTHOR


WORKING TOGETHER ON AN ARTICLE WITH GIT
YOU
REMOTE SERVER
OBSCURE COAUTHOR
INIT THEO DEF
YOUR REPOSITORY
pull
INIT THEO DEF
REMOTE REPOSITORY
INIT THEO DEF
OBSCURE'S REPOSITORY
your article.pdf
Toulouse has the best French accent
You Obscure Coauthor
In this document, $\mathcal{A}$ denotes the set of all French accents.
Definition 1. *Define a relation order $>$ on $\mathcal{A}$ such that for any two accents $a_1, a_2 \in \mathcal{A}$, we have $a_1 > a_2$ if and only if accent a_1 is considered* "greater than" *accent a_2 based on an objective set of criteria (including musicality, clarity, cultural significance).*
Theorem 2. *The Toulouse accent is the supremum of $\mathcal{A}$.*



Toulouse has the best French accent

You Obscure Coauthor

In this document, $\mathcal{A}$ denotes the set of all French accents.

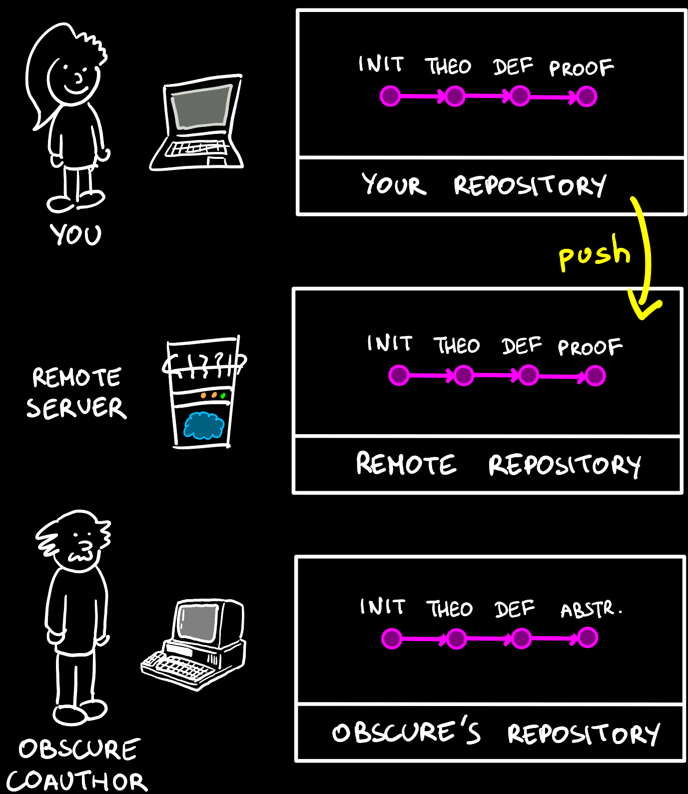
Definition 1. *Define a relation order $>$ on $\mathcal{A}$ such that for any two accents $a_1, a_2 \in \mathcal{A}$, we have $a_1 > a_2$ if and only if accent a_1 is considered* "greater than" *accent a_2 based on an objective set of criteria (including musicality, clarity, cultural significance).*

Theorem 2. *The Toulouse accent is the supremum of $\mathcal{A}$.*

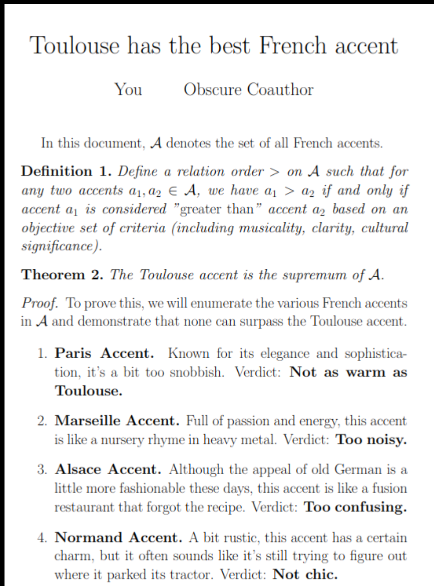
Proof. To prove this, we will enumerate the various French accents in $\mathcal{A}$ and demonstrate that none can surpass the Toulouse accent.

1. **Paris Accent.** Known for its elegance and sophistication, it's a bit too snobbish. Verdict: **Not as warm as Toulouse.**
2. **Marseille Accent.** Full of passion and energy, this accent is like a nursery rhyme in heavy metal. Verdict: **Too noisy.**
3. **Alsace Accent.** Although the appeal of old German is a little more fashionable these days, this accent is like a fusion restaurant that forgot the recipe. Verdict: **Too confusing.**
4. **Normand Accent.** A bit rustic, this accent has a certain charm, but it often sounds like it's still trying to figure out where it parked its tractor. Verdict: **Not chic.**

WORKING TOGETHER ON AN ARTICLE WITH GIT



your article.pdf



Toulouse has the best French accent

You Obscure Coauthor

In this document, $\mathcal{A}$ denotes the set of all French accents.

Definition 1. *Define a relation order $>$ on $\mathcal{A}$ such that for any two accents $a_1, a_2 \in \mathcal{A}$, we have $a_1 > a_2$ if and only if accent a_1 is considered* "greater than" *accent a_2 based on an objective set of criteria (including musicality, clarity, cultural significance).*

Theorem 2. *The Toulouse accent is the supremum of $\mathcal{A}$.*

Proof. To prove this, we will enumerate the various French accents in $\mathcal{A}$ and demonstrate that none can surpass the Toulouse accent.

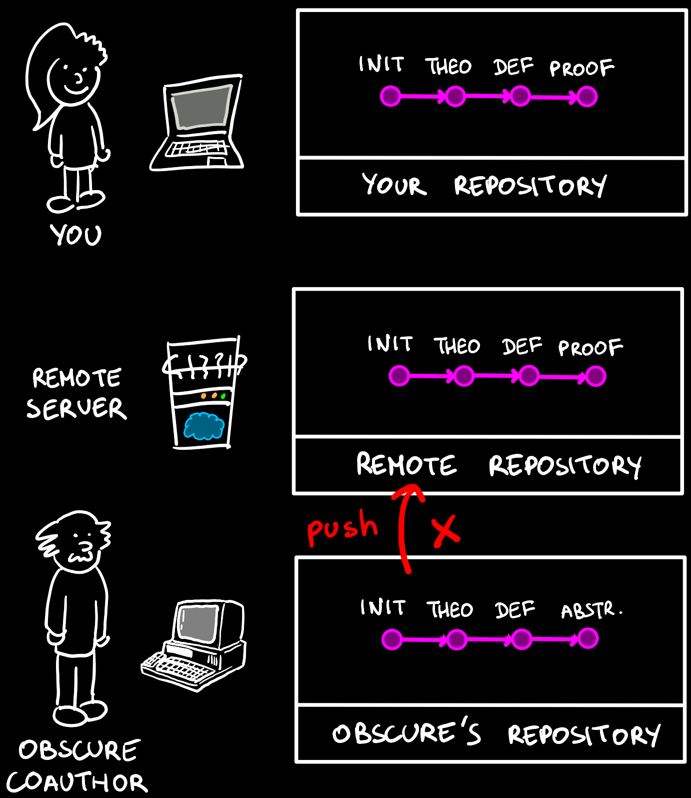
1. **Paris Accent.** Known for its elegance and sophistication, it's a bit too snobbish. Verdict: **Not as warm as Toulouse.**

2. **Marseille Accent.** Full of passion and energy, this accent is like a nursery rhyme in heavy metal. Verdict: **Too noisy.**

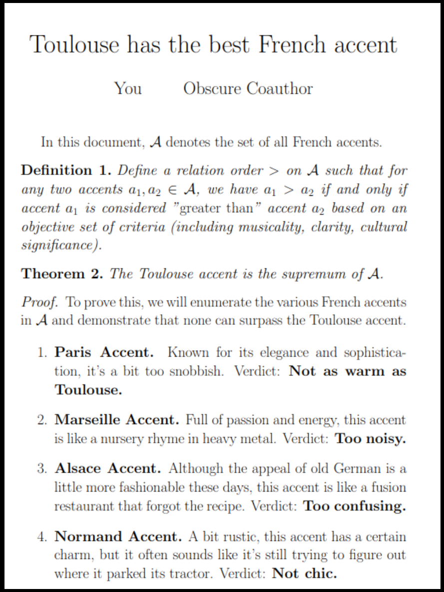
3. **Alsace Accent.** Although the appeal of old German is a little more fashionable these days, this accent is like a fusion restaurant that forgot the recipe. Verdict: **Too confusing.**

4. **Normand Accent.** A bit rustic, this accent has a certain charm, but it often sounds like it's still trying to figure out where it parked its tractor. Verdict: **Not chic.**

WORKING TOGETHER ON AN ARTICLE WITH GIT



your article.pdf



Toulouse has the best French accent

You Obscure Coauthor

In this document, $\mathcal{A}$ denotes the set of all French accents.

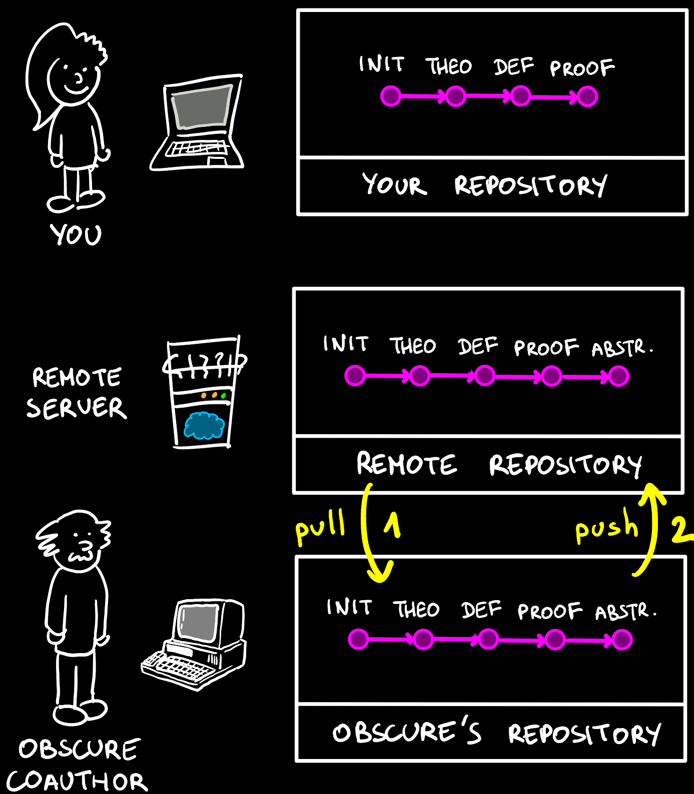
Definition 1. *Define a relation order > on* $\mathcal{A}$ *such that for any two accents* $a_1, a_2 \in \mathcal{A}$*, we have* $a_1 > a_2$ *if and only if accent* a_1 *is considered* "greater than" *accent* a_2 *based on an objective set of criteria (including musicality, clarity, cultural significance).*

Theorem 2. *The Toulouse accent is the supremum of* $\mathcal{A}$*.*

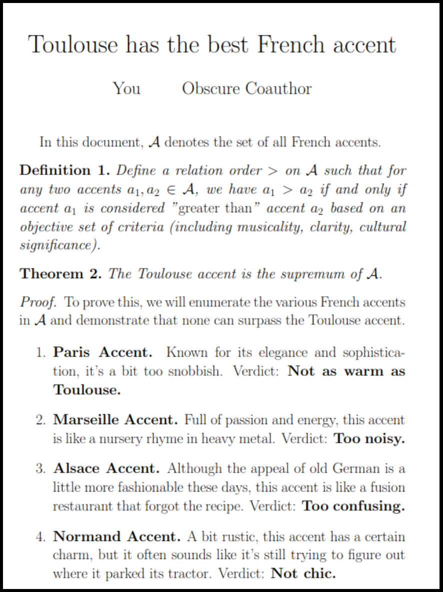
Proof. To prove this, we will enumerate the various French accents in $\mathcal{A}$ and demonstrate that none can surpass the Toulouse accent.

1. **Paris Accent.** Known for its elegance and sophistication, it's a bit too snobbish. Verdict: **Not as warm as Toulouse.**
2. **Marseille Accent.** Full of passion and energy, this accent is like a nursery rhyme in heavy metal. Verdict: **Too noisy.**
3. **Alsace Accent.** Although the appeal of old German is a little more fashionable these days, this accent is like a fusion restaurant that forgot the recipe. Verdict: **Too confusing.**
4. **Normand Accent.** A bit rustic, this accent has a certain charm, but it often sounds like it's still trying to figure out where it parked its tractor. Verdict: **Not chic.**

WORKING TOGETHER ON AN ARTICLE WITH GIT



your article.pdf



Toulouse has the best French accent

You Obscure Coauthor

In this document, $\mathcal{A}$ denotes the set of all French accents.

Definition 1. *Define a relation order $>$ on $\mathcal{A}$ such that for any two accents $a_1, a_2 \in \mathcal{A}$, we have $a_1 > a_2$ if and only if accent a_1 is considered* "greater than" *accent a_2 based on an objective set of criteria (including musicality, clarity, cultural significance).*

Theorem 2. *The Toulouse accent is the supremum of $\mathcal{A}$.*

Proof. To prove this, we will enumerate the various French accents in $\mathcal{A}$ and demonstrate that none can surpass the Toulouse accent.

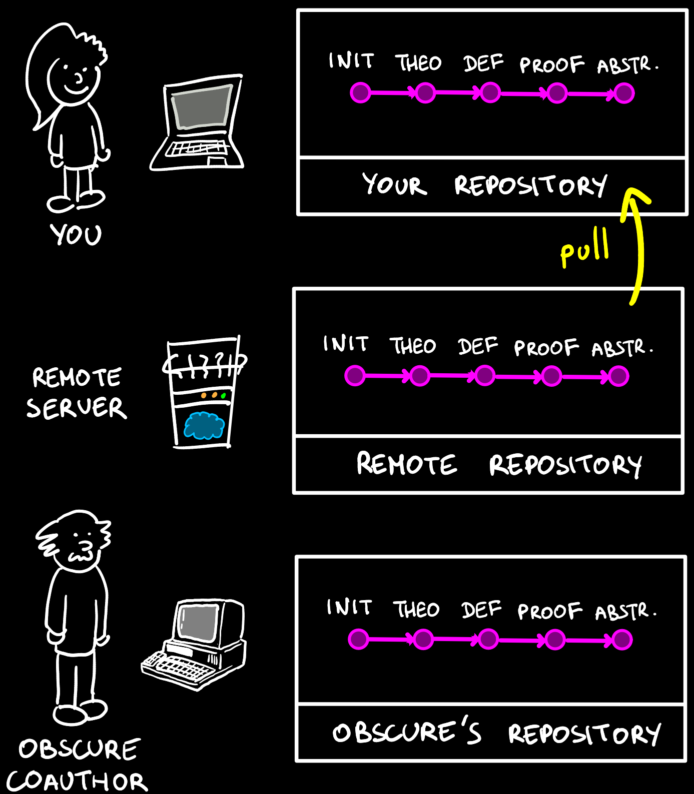
1. **Paris Accent.** Known for its elegance and sophistication, it's a bit too snobbish. Verdict: **Not as warm as Toulouse.**

2. **Marseille Accent.** Full of passion and energy, this accent is like a nursery rhyme in heavy metal. Verdict: **Too noisy.**

3. **Alsace Accent.** Although the appeal of old German is a little more fashionable these days, this accent is like a fusion restaurant that forgot the recipe. Verdict: **Too confusing.**

4. **Normand Accent.** A bit rustic, this accent has a certain charm, but it often sounds like it's still trying to figure out where it parked its tractor. Verdict: **Not chic.**

WORKING TOGETHER ON AN ARTICLE WITH GIT



your article.pdf

Toulouse has the best French accent

You Obscure Coauthor

Abstract

This article demonstrates that the Toulouse accent is the supremum of the set of French accents $\mathcal{A}$ by exhaustively enumerating its competitors, ultimately revealing that none can match its warmth, charm, and delightful character.

In this document, $\mathcal{A}$ denotes the set of all French accents.

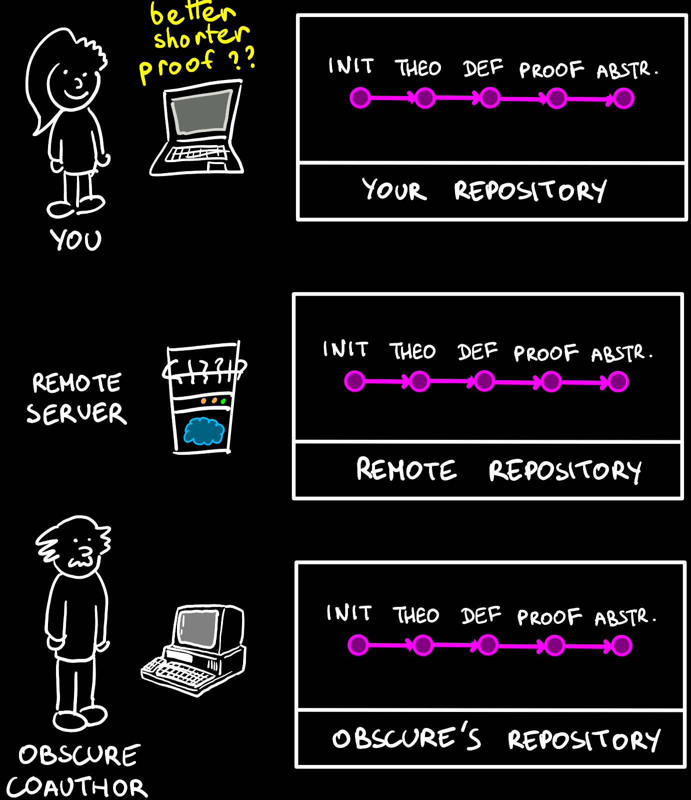
Definition 1. *Define a relation order $>$ on $\mathcal{A}$ such that for any two accents $a_1, a_2 \in \mathcal{A}$, we have $a_1 > a_2$ if and only if accent a_1 is considered* "greater than" *accent a_2 based on an objective set of criteria (including musicality, clarity, cultural significance).*

Theorem 2. *The Toulouse accent is the supremum of $\mathcal{A}$.*

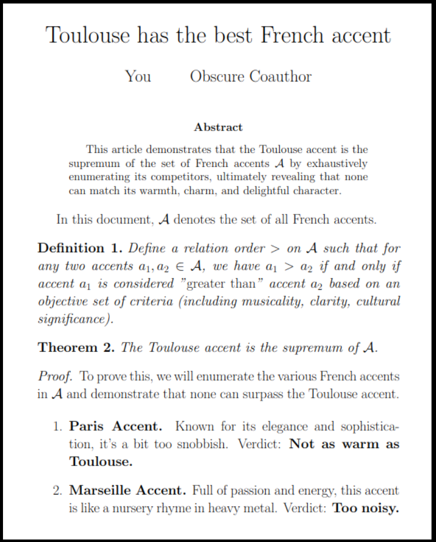
Proof. To prove this, we will enumerate the various French accents in $\mathcal{A}$ and demonstrate that none can surpass the Toulouse accent.

1. **Paris Accent.** Known for its elegance and sophistication, it's a bit too snobbish. Verdict: **Not as warm as Toulouse.**
2. **Marseille Accent.** Full of passion and energy, this accent is like a nursery rhyme in heavy metal. Verdict: **Too noisy.**

WORKING TOGETHER ON AN ARTICLE WITH GIT



your article.pdf



Toulouse has the best French accent

You   Obscure Coauthor

Abstract

This article demonstrates that the Toulouse accent is the supremum of the set of French accents $\mathcal{A}$ by exhaustively enumerating its competitors, ultimately revealing that none can match its warmth, charm, and delightful character.

In this document, $\mathcal{A}$ denotes the set of all French accents.

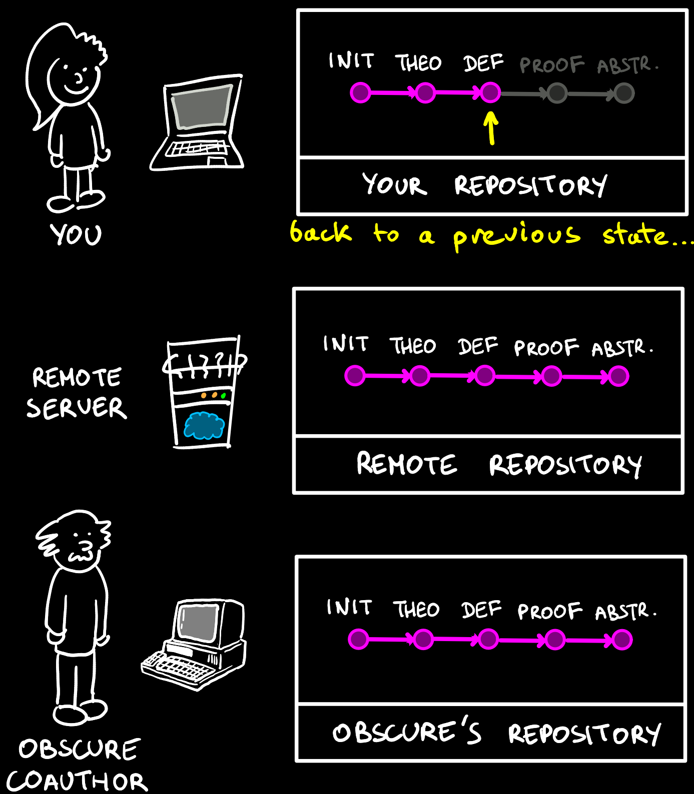
Definition 1. *Define a relation order $>$ on $\mathcal{A}$ such that for any two accents $a_1, a_2 \in \mathcal{A}$, we have $a_1 > a_2$ if and only if accent a_1 is considered* "greater than" *accent a_2 based on an objective set of criteria (including musicality, clarity, cultural significance).*

Theorem 2. *The Toulouse accent is the supremum of $\mathcal{A}$.*

Proof. To prove this, we will enumerate the various French accents in $\mathcal{A}$ and demonstrate that none can surpass the Toulouse accent.

1. **Paris Accent.** Known for its elegance and sophistication, it's a bit too snobbish. Verdict: **Not as warm as Toulouse.**
2. **Marseille Accent.** Full of passion and energy, this accent is like a nursery rhyme in heavy metal. Verdict: **Too noisy.**

WORKING TOGETHER ON AN ARTICLE WITH GIT



your article.pdf

Toulouse has the best French accent

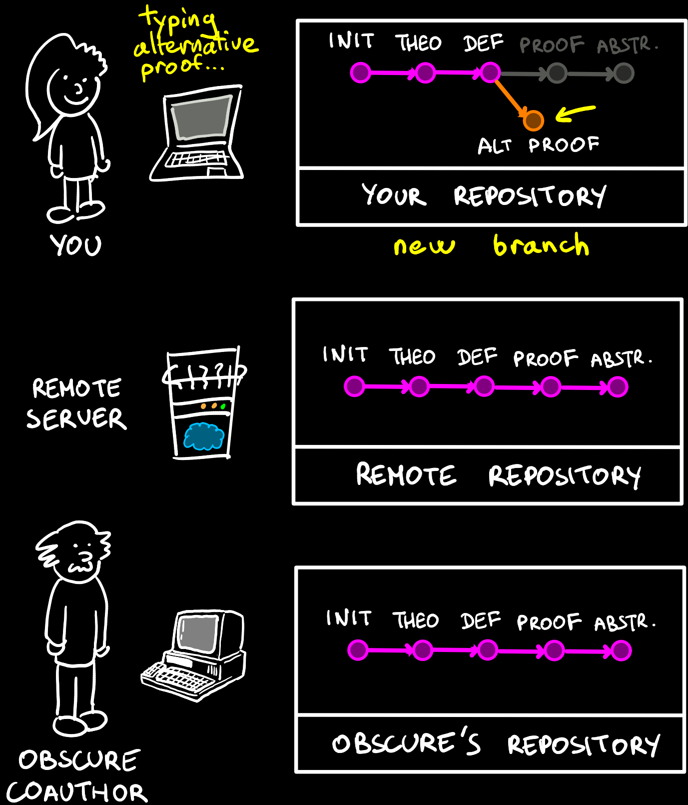
You   Obscure Coauthor

In this document, $\mathcal{A}$ denotes the set of all French accents.

Definition 1. *Define a relation order $>$ on $\mathcal{A}$ such that for any two accents $a_1, a_2 \in \mathcal{A}$, we have $a_1 > a_2$ if and only if accent a_1 is considered* "greater than" *accent a_2 based on an objective set of criteria (including musicality, clarity, cultural significance).*

Theorem 2. *The Toulouse accent is the supremum of $\mathcal{A}$.*

WORKING TOGETHER ON AN ARTICLE WITH GIT



your article.pdf

Toulouse has the best French accent

You Obscure Coauthor

In this document, $\mathcal{A}$ denotes the set of all French accents.

Definition 1. *Define a relation order $>$ on $\mathcal{A}$ such that for any two accents $a_1, a_2 \in \mathcal{A}$, we have $a_1 > a_2$ if and only if accent a_1 is considered* "greater than" *accent a_2 based on an objective set of criteria (including musicality, clarity, cultural significance).*

Theorem 2. *The Toulouse accent is the supremum of $\mathcal{A}$.*

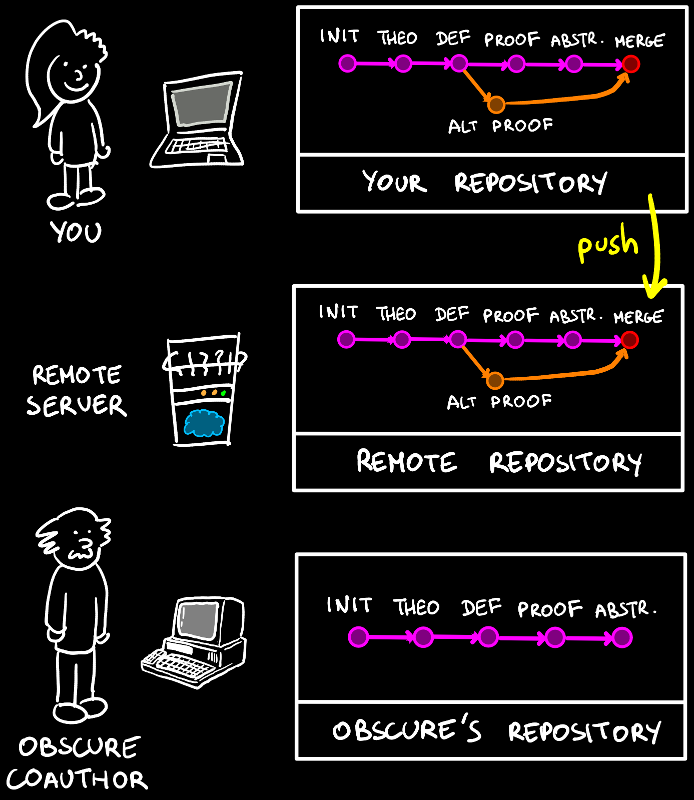
Proof. (shorter proof ??)

Assume, for the sake of contradiction, that the Toulouse accent is **not** the supremum of $\mathcal{A}$. Then, there exists an accent $a \in \mathcal{A}$ such that a is greater than the Toulouse accent.

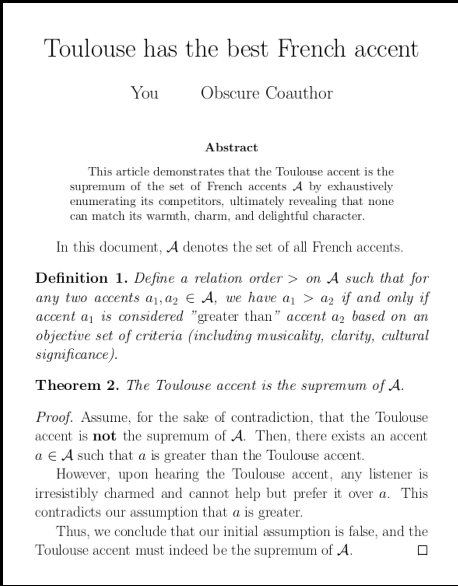
However, upon hearing the Toulouse accent, any listener is irresistibly charmed and cannot help but prefer it over a. This contradicts our assumption that a is greater.

Thus, we conclude that our initial assumption is false, and the Toulouse accent must indeed be the supremum of $\mathcal{A}$. □

WORKING TOGETHER ON AN ARTICLE WITH GIT



your article.pdf



Toulouse has the best French accent

You Obscure Coauthor

Abstract

This article demonstrates that the Toulouse accent is the supremum of the set of French accents $\mathcal{A}$ by exhaustively enumerating its competitors, ultimately revealing that none can match its warmth, charm, and delightful character.

In this document, $\mathcal{A}$ denotes the set of all French accents.

Definition 1. *Define a relation order $>$ on $\mathcal{A}$ such that for any two accents $a_1, a_2 \in \mathcal{A}$, we have $a_1 > a_2$ if and only if accent a_1 is considered* "greater than" *accent a_2 based on an objective set of criteria (including musicality, clarity, cultural significance).*

Theorem 2. *The Toulouse accent is the supremum of $\mathcal{A}$.*

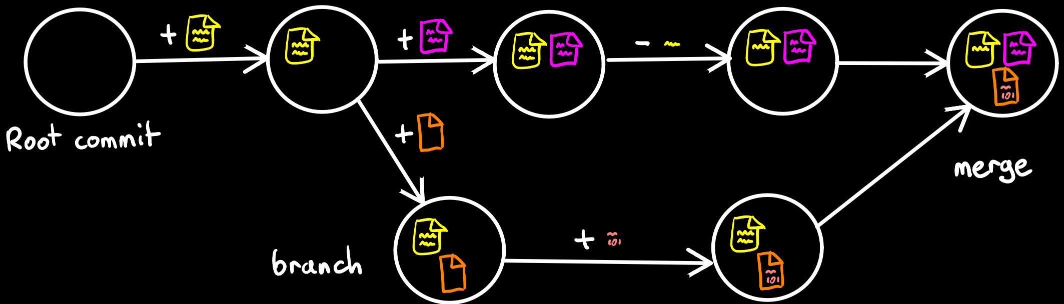
Proof. Assume, for the sake of contradiction, that the Toulouse accent is **not** the supremum of $\mathcal{A}$. Then, there exists an accent $a \in \mathcal{A}$ such that a is greater than the Toulouse accent.

However, upon hearing the Toulouse accent, any listener is irresistibly charmed and cannot help but prefer it over a. This contradicts our assumption that a is greater.

Thus, we conclude that our initial assumption is false, and the Toulouse accent must indeed be the supremum of $\mathcal{A}$. □

IN SUMMARY...

git is a Version Control System (VCS) :
it stores all the project states over time.

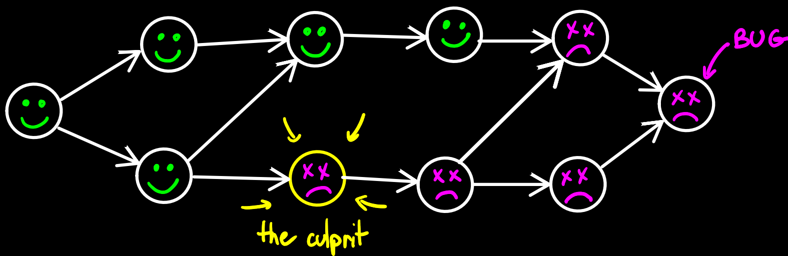


Git lets you create parallel development branches
that can be integrated later.

This forms a Directed Acyclic Graph (DAG),
where the vertices are the project states, also named commits.

PART I
GIT BISECT

PROBLEM: FINDING A REGRESSION



Input A commit graph in which a commit is known to be bugged, the other commits are bugged or clean (= bug-free)

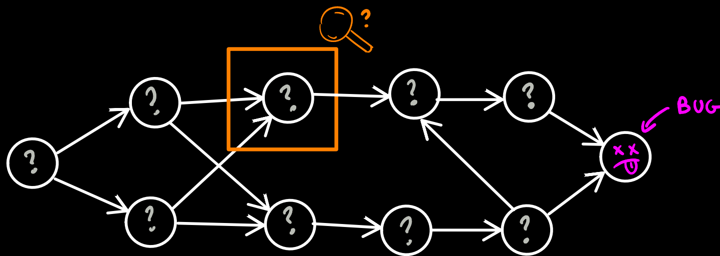
Question Which commit has originally introduced the bug?

Assumptions

- If a parent of a commit is bugged, then the commit is bugged.
- Only one commit has introduced the bug, namely the faulty commit (or regression)

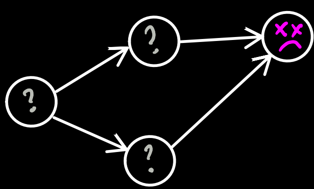
HOW TO INVESTIGATE

Unique operation: QUERY of a commit with unknown status



If bugged,

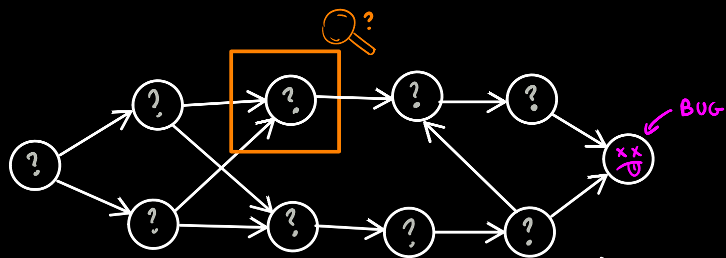
then the faulty commit is an ancestor of this commit



ancestor of a vertex v =
v or
an ancestor of a parent of v

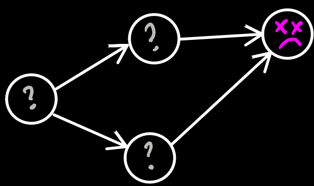
HOW TO INVESTIGATE

Unique operation: QUERY of a commit with unknown status



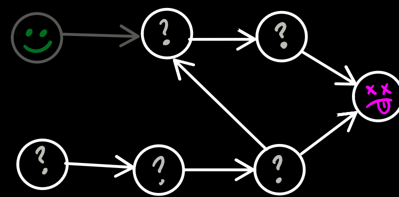
If bugged,

then the faulty commit is an ancestor of this commit



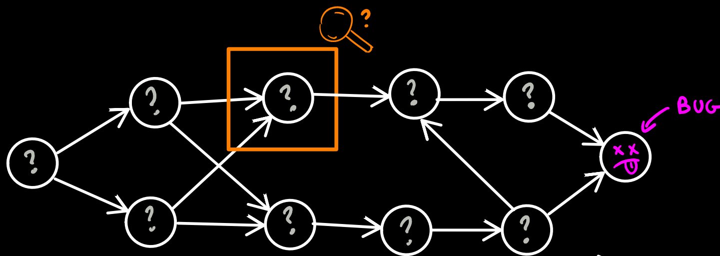
If clean,

then the faulty commit is not an ancestor of this commit

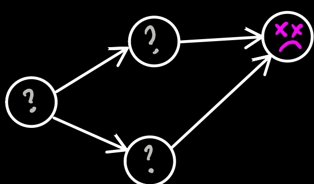


HOW TO INVESTIGATE

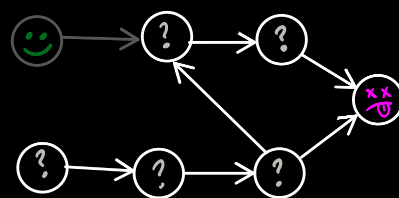
Unique operation: QUERY of a commit with unknown status



If bugged, then the faulty commit is an ancestor of this commit



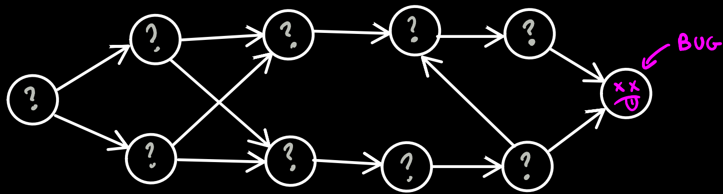
If clean, then the faulty commit is not an ancestor of this commit



The faulty commit is found whenever there remains only 1 suspect.

REGRESSION SEARCH PROBLEM

Input: a DAG where each vertex has an unknown status, except one, which is bugged.

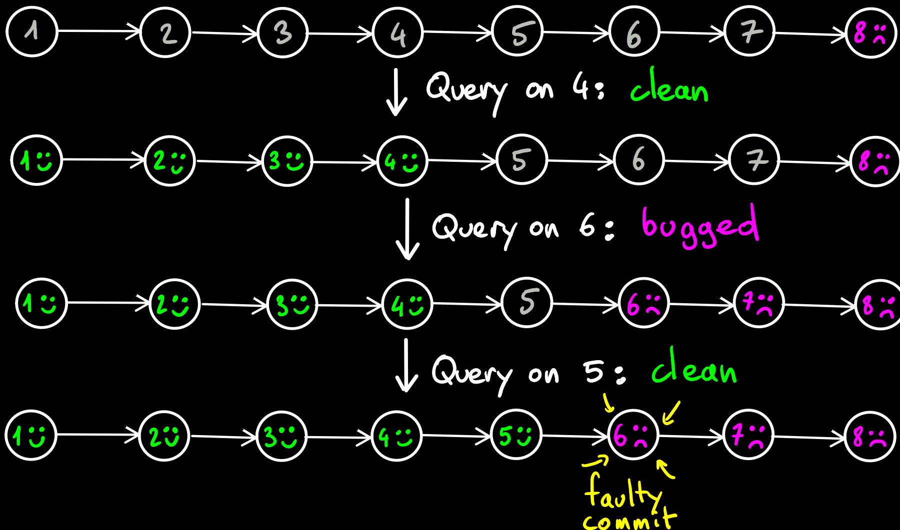


Output: A strategy that finds the faulty commit with a minimal number of queries in the worst-case scenario
= optimal strategy

(the faulty commit can be any ancestor of the bugged vertex)

In real life, queries are costly.

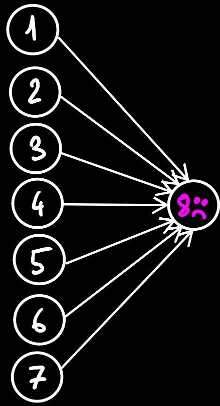
FIRST EXAMPLE : CHAINS

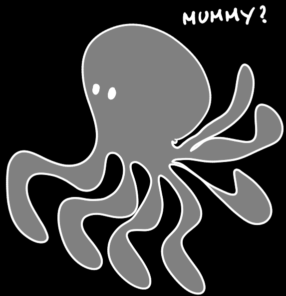


optimal strategy = binary search

More generally, number of queries in an optimal strategy in a chain of length n = $\lceil \log_2(n) \rceil$

SECOND EXAMPLE : OCTOPUSES





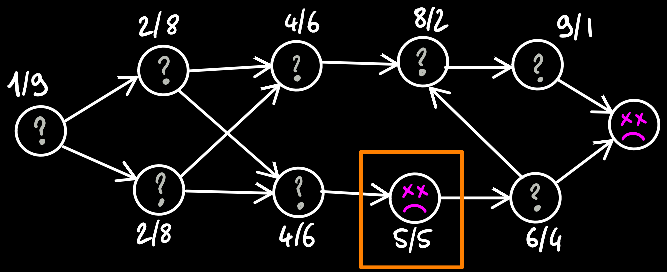
optimal strategy = whatever

More generally, number of queries in an optimal strategy in an octopus of size n = $n-1$

THE GIT BISECT ALGORITHM

git uses a heuristic to find the faulty commit: git bisect

STEP 1: Compute the number of ancestors/non-ancestors for each commit



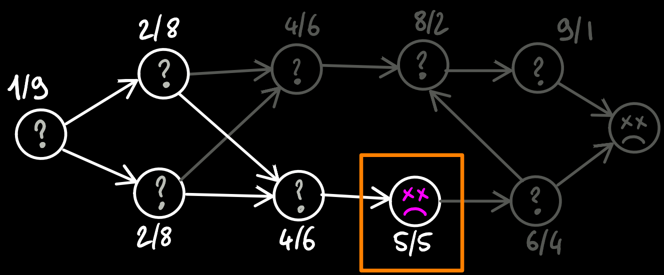
STEP 2: Query the vertex with the most balanced ratio

STEP 3:

THE GIT BISECT ALGORITHM

git uses a heuristic to find the faulty commit: git bisect

STEP 1: Compute the number of ancestors/non-ancestors for each commit



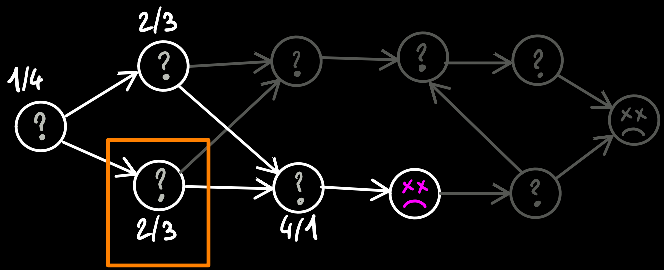
STEP 2: Query the vertex with the most balanced ratio

STEP 3: Delete the innocent commits and recurse.

THE GIT BISECT ALGORITHM

git uses a heuristic to find the faulty commit: git bisect

STEP 1: Compute the number of ancestors/non-ancestors for each commit



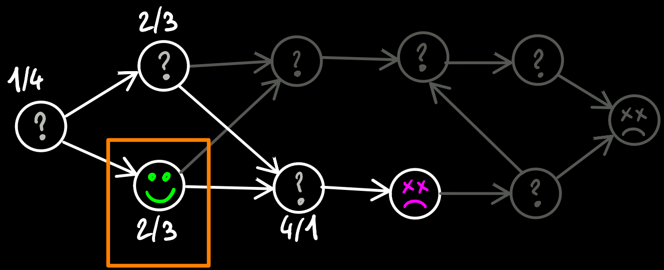
STEP 2: Query ~~the~~ a vertex with the most balanced ratio

STEP 3: Delete the innocent commits and recurse.

THE GIT BISECT ALGORITHM

git uses a heuristic to find the faulty commit: git bisect

STEP 1: Compute the number of ancestors/non-ancestors for each commit



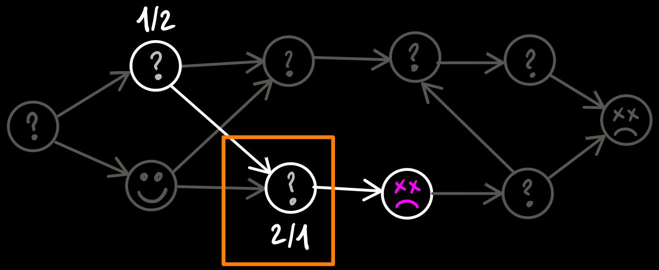
STEP 2: Query ~~the~~ a vertex with the most balanced ratio

STEP 3: Delete the innocent commits and recurse.

THE GIT BISECT ALGORITHM

git uses a heuristic to find the faulty commit: git bisect

STEP 1: Compute the number of ancestors/non-ancestors for each commit



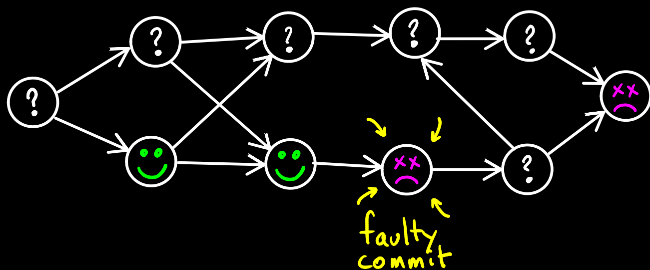
STEP 2: Query ~~the~~ a vertex with the most balanced ratio

STEP 3: Delete the innocent commits and recurse.

THE GIT BISECT ALGORITHM

git uses a heuristic to find the faulty commit: git bisect

STEP 1: Compute the number of ancestors/non-ancestors for each commit



STEP 2: Query ~~the~~ a vertex with the most balanced ratio

STEP 3: Delete the innocent commits and recurse.

QUESTION: Does git bisect always give an optimal strategy?

HOW GOOD IS GIT BISECT?

QUESTION: Does git bisect always give an optimal strategy?

NO The Regression Search Problem is NP-complete.

[Carmo Donadelli Kohayakawa Laber 2004] [We've also proved it!]

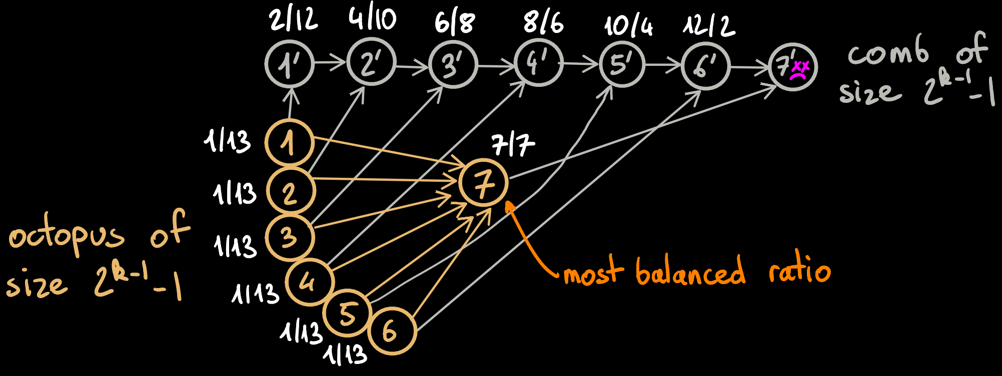
But is it really that bad? **YEAH!**

Proposition For any k, there exists a DAG such that an optimal strategy uses k queries and git bisect always uses $2^{k-1}-1$ queries.

HOW GOOD IS GIT BISECT?

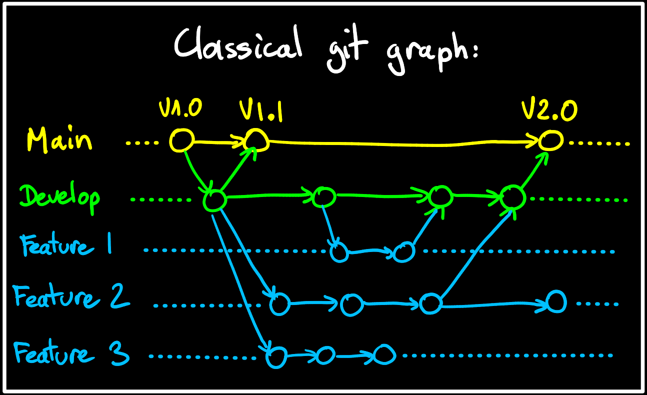
Proposition For any k, there exists a DAG such that an optimal strategy uses k queries and git bisect always uses $2^{k-1}-1$ queries.

Proof for $k=4$:



BACK TO REALITY?

Octopus substructures are unrealistic



Usually, we never merge more than 2 branches.

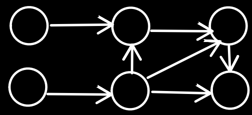
(Otherwise it is called an octopus merge )

Binary DAGs

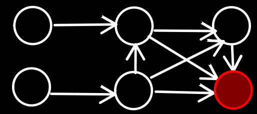
Definition

binary DAG = DAG where the vertices have indegree ≤ 2

Ex:



Good



Bad

THEOREM

For any binary DAG with n vertices, git bisect uses at most $\leq \log_{\frac{3}{2}}(n)$ queries

---

git bisect is a $\dfrac{1}{\log_2\left(\frac{3}{2}\right)}$-approximation algorithm when it is used on binary DAGs.

$\dfrac{1}{\log_2\left(\frac{3}{2}\right)} \simeq 1{,}71$ is the optimal constant.

BINARY DAGS

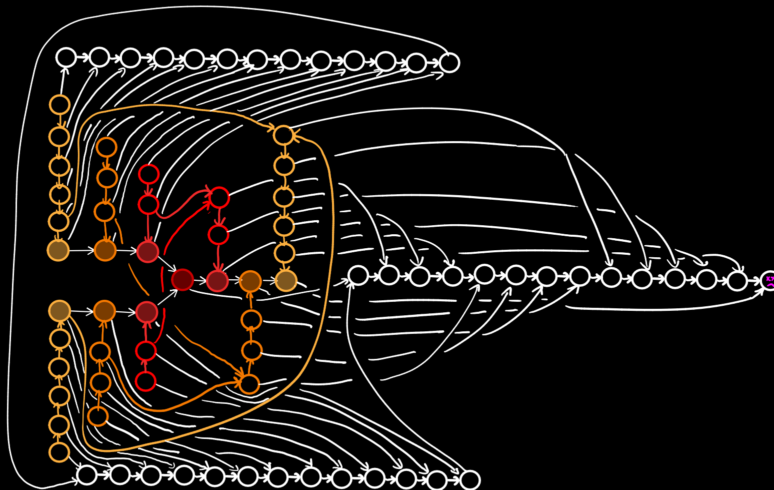
THEOREM

For any binary DAG with n vertices, git bisect uses at most $\leq \log_{\frac{3}{2}}(n)$ queries

git bisect is a $\frac{1}{\log_2(\frac{3}{2})}$-approximation algorithm when it is used on binary DAGs.

$\frac{1}{\log_2(\frac{3}{2})} \simeq 1{,}71$ is the optimal constant.

Example of a mean 😈 binary DAG



GOLDEN BISECT

We've improved git bisect in the worst-case scenario

→ new algorithm : ⭐ golden bisect ⭐

Idea behind golden bisect	• If git bisect queries a vertex with score $\geq \frac{\text{nb vertices}}{\phi^2}$, query the same • Otherwise, query some special vertex even if it has not best score.

ϕ = golden ratio = $(1+\sqrt{5})/2$

THEOREM

For any binary DAG with n vertices, golden bisect uses at most $\leq \log_\phi(n)+1$ queries

golden bisect is a $\frac{1}{\log_2(\phi)}$-approximation algorithm when it is used on binary DAGs.

$\frac{1}{\log_2(\phi)} \simeq 1{,}44$ is the optimal constant.

WHERE IS MY ALEA ?

Number of queries in the average-case scenario?

- DAG : binary, fixed, n vertices (not random)
- Position of the faulty commit: Uniformly at random amongst vertices of the DAG

NEW RESULT

THEOREM

- The average number of queries for git bisect is $\leq \log_{\frac{3}{2}2^{1/3}}(n)$
- The average number of queries for golden bisect is $\leq \log_{\phi^{1+\phi^{-2}}}(n)$

[C., Dorbec, Jugé]

QUESTION

Have we really proved that git bisect is bad?

→ The examples of DAGs where git bisect performs poorly shouldn't occur in real life

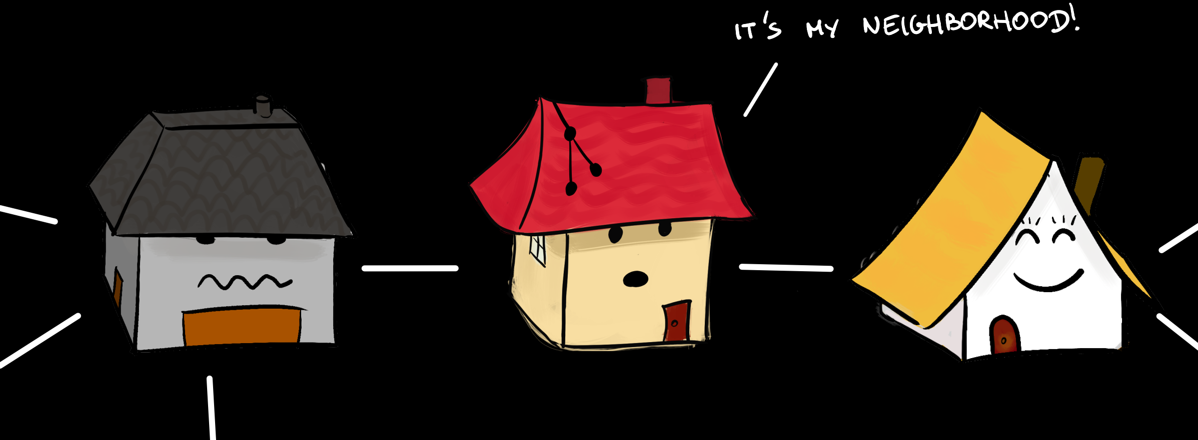
→ Sometimes git bisect is better than golden bisect

More relevant(?) question	What is the average-case complexity of git bisect?

This question calls for many more, notably what is a random Git graph?

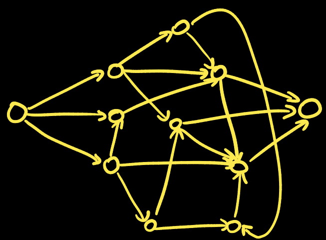
PART II
RANDOM GIT GRAPHS

ongoing work

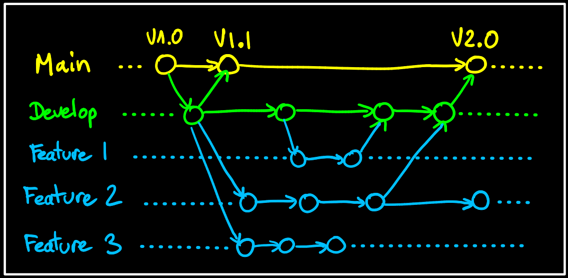


WHICH GRAPHS TO CONSIDER ?

In git, every DAG without restriction can be generated...



... but many projects follow a workflow



In the following, we consider a simple workflow but widely used in industry: the <u>feature branch workflow</u>

GIT GRAPH

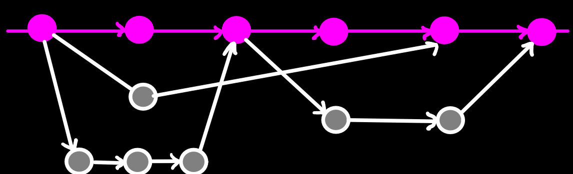
DEFINITION

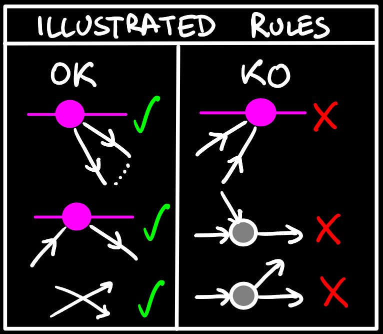
(feature branch) Git graph = DAG with

- a main branch (path of magenta vertices)
- + 0, 1 or several feature branches, paths of ≥ 1 white vertices starting and ending on magenta vertices
- indegree ≤ 2 for all vertices

previously defined in [Lecoq 2024]

e.g.





GIT GRAPH

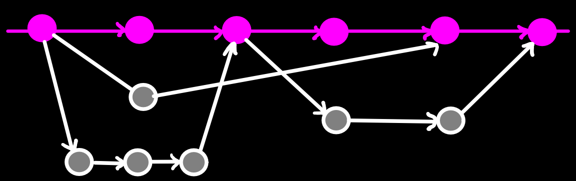
DEFINITION

(feature branch) Git graph = DAG with

- a main branch (path of magenta vertices)
  +
- 0, 1 or several feature branches, paths of ≥ 1 white vertices starting and ending on magenta vertices
- indegree ≤ 2 for all vertices

previously defined in [Lecoq 2024]

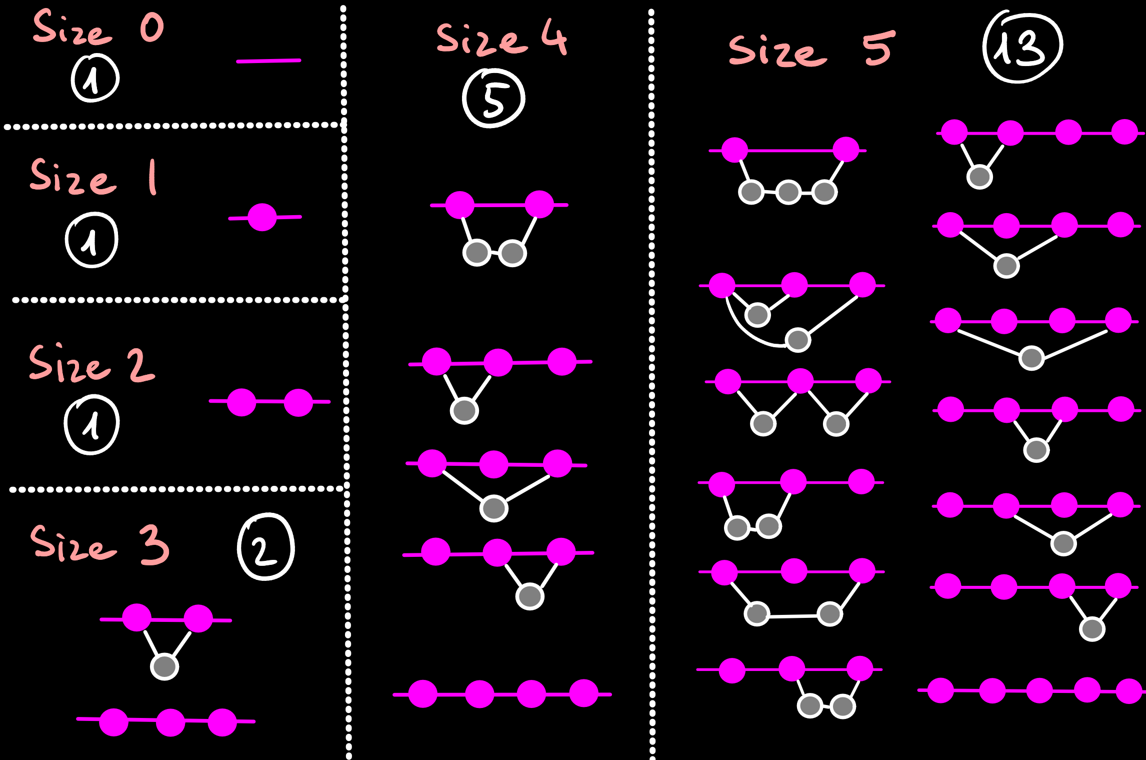
e.g.



GOALS

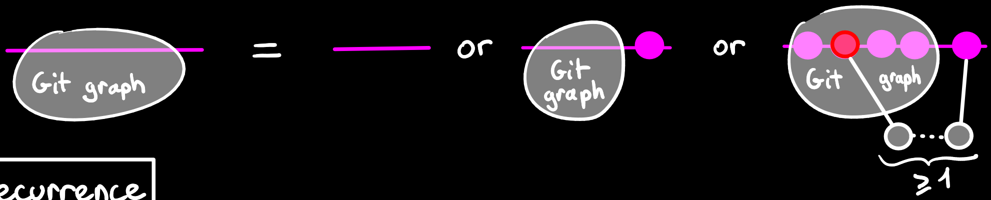
- Counting Git graphs (exactly or asymptotically)
- Random generation of Git graphs

ALL SMALL GIT GRAPHS



RECURSIVE DECOMPOSITION

Decomposition



Recurrence

$$g_{n,k} = g_{n-1,k-1} + \sum_{\ell \geq 1} (k-1)\, g_{n-1-\ell,\,k-1} \quad \text{for } n \geq 1$$

where $g_{n,k}$:= number of Git graphs with n vertices, k of them being magenta

Differential Equation for the Generating Function

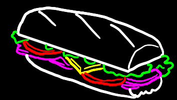
$$G(z,u) = 1 + zu\,G(z,u) + \frac{z^2u^2}{1-z}\frac{\partial G}{\partial u}(z,u)$$

where $G(z,u) = \sum_{n \geq 0} \sum_{k \geq 0} g_{n,k} z^n u^k$

⚠ $G(z,u)$ is not analytic.

THREE ANGLES OF ATTACK

Sandwich method



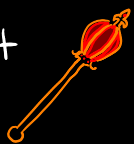
Finding subsets and supersets of Git graphs easier to count

↓

Counting Git Graphs

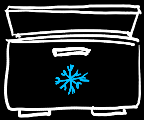
↗ ↖

Ennoblement method



making the generating function less ordinary

Freezer method



Counting by (temporarily) fixing the number of magenta vertices

ENNOBLEMENT METHOD

Recurrence	$g_{n,k} = g_{n-1,k-1} + \sum_{\ell \geq 1} (k-1)\, g_{n-1-\ell,k-1}$
Differential Equation	$G(z,u) = 1 + zu\,G(z,u) + \frac{z^2u^2}{1-z}\frac{\partial G}{\partial u}(z,u)$

Usual trick:

Borel transform

Ordinary Generating Function $\underbrace{\sum_{n,k\geq 0} g_{n,k} z^n u^k}_{G(z,u)}$, not analytic ✗

Exponential Generating Function $\underbrace{\sum_{n,k\geq 0} \frac{g_{n,k}}{n!} z^n u^k}$ analytic, but no pretty equation ✗

Borel transform on u

$$\tilde{G}(z,u) = \sum_{n,k\geq 0} \frac{g_{n,k}}{k!} z^n u^k$$

analytic ✓

and

Differential Equation for $\tilde{G}$
$\frac{\partial \tilde{G}}{\partial u} = z\,\tilde{G} + \frac{z^2u}{1-z}\frac{\partial \tilde{G}}{\partial u}$

✓

ENNOBLEMENT METHOD

Recurrence	$g_{n,k} = g_{n-1,k-1} + \sum_{\ell \geq 1} (k-1)\, g_{n-1-\ell,k-1}$
Differential Equation	$G(z,u) = 1 + zu\,G(z,u) + \frac{z^2 u^2}{1-z} \frac{\partial G}{\partial u}(z,u)$

$$\tilde{G}(z,u) = \sum_{n,k \geq 0} \frac{g_{n,k}}{k!} z^n u^k$$

analytic ✓

and

Differential Equation for $\tilde{G}$

$$\frac{\partial \tilde{G}}{\partial u} = z\tilde{G} + \frac{z^2 u}{1-z} \frac{\partial \tilde{G}}{\partial u}$$

✓

this can be solved!

Theorem

$$\tilde{G}(z,u) = \left(1 - \frac{z^2 u}{1-z}\right)^{-\frac{1-z}{z}}$$

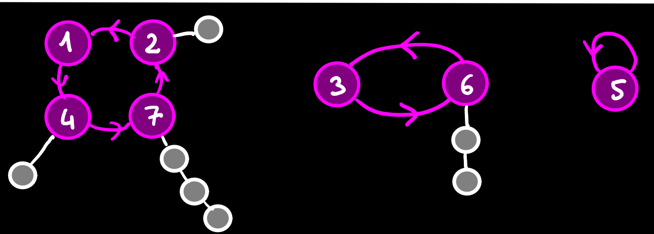
How can it be exploited?

DIFFERENT PERSPECTIVE

Definition

cyclarium = set of cycles of magenta vertices labeled from 1 to k where a chain of white unlabeled vertices is attached to each magenta vertex, except to the ones having the smallest label in their cycles.

e.g.:



Is there a combinatorial explanation for the formula

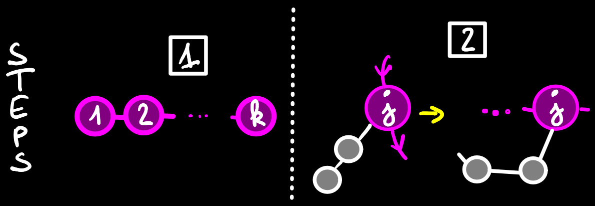
$$\tilde{G}(z,u) = \left(1 - \frac{z^2 u}{1-z}\right)^{-\frac{1-z}{z}} = \exp\left(\frac{1}{\frac{z}{1-z}} \ln\left(\frac{1}{1 - uz\frac{z}{1-z}}\right)\right) ?$$

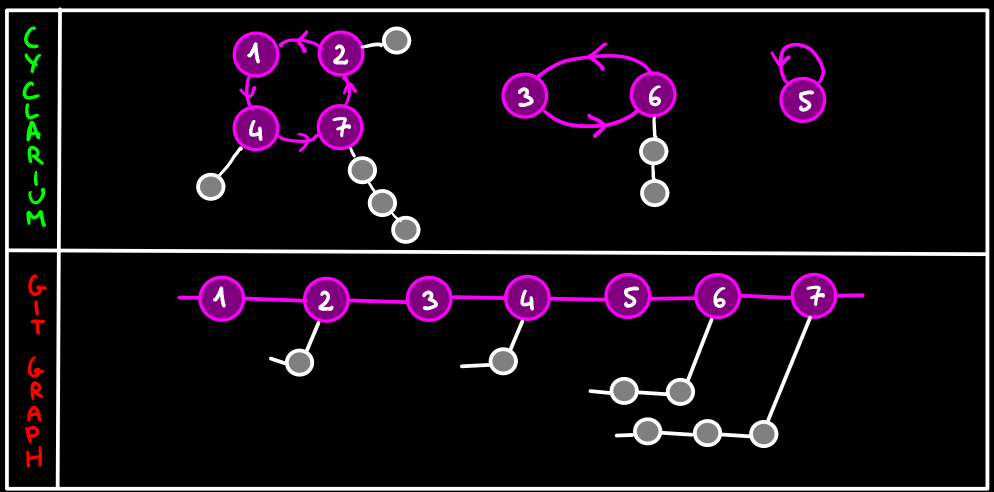
It's the generating function of cyclariums!

BIJECTION

Proposition

There is a bijection from cyclariums to Git graphs:

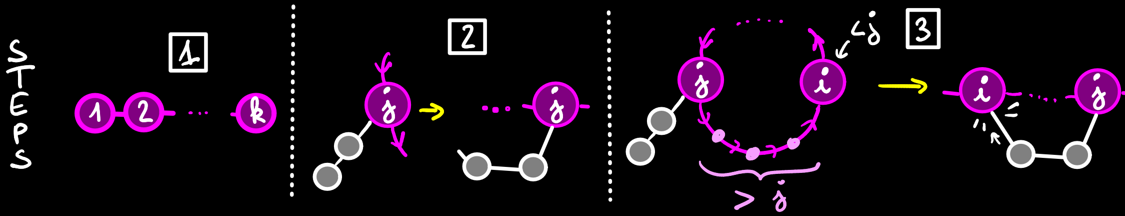


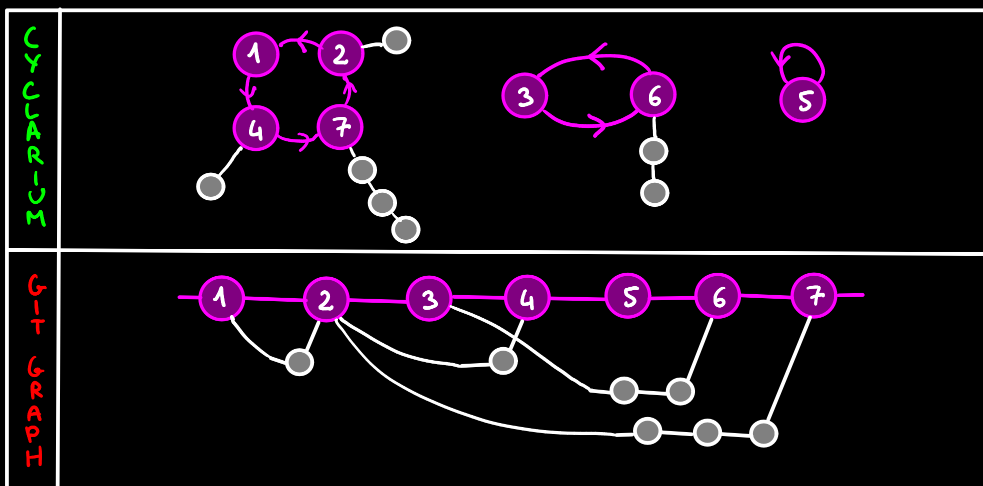


BIJECTION

Proposition

There is a bijection from cyclariums to Git graphs:

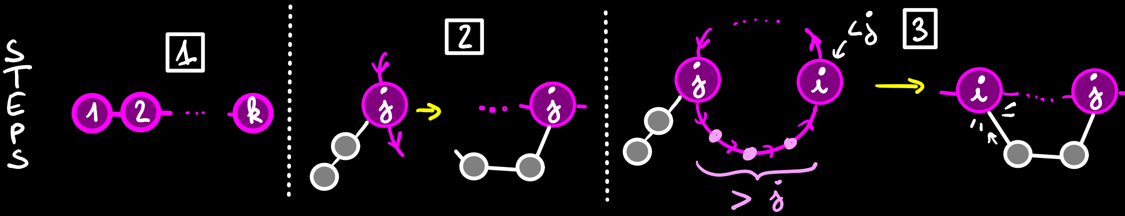


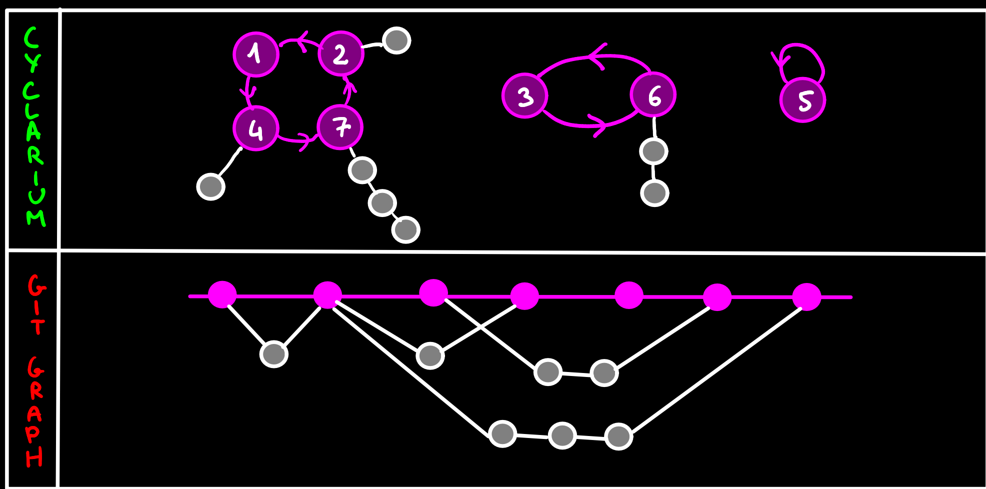


BIJECTION

Proposition

There is a bijection from cyclariums to Git graphs:





BIJECTION

Proposition

There is a bijection from cyclariums to Git graphs sending

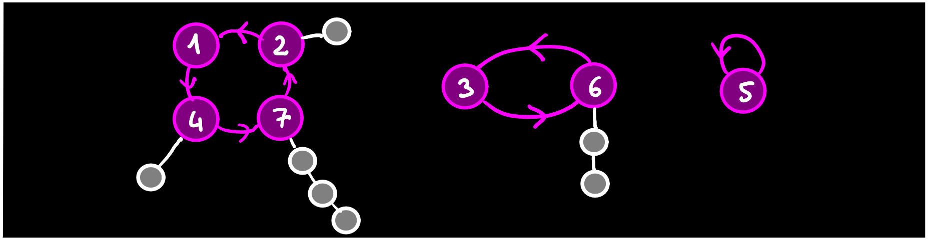
vertices ⟶ vertices

magenta vertices ⟶ magenta vertices

cycles ⟶ free vertices
i.e magenta vertices of indegree ≤ 1

cycle lengths ⟶ magenta vertices in the connected components when the magenta edges are erased.

CYCLARIUM



Corollary

$$g_{n,k} = \sum_{f=1}^{k-1} \begin{bmatrix} k \\ f \end{bmatrix} \binom{n-k-1}{k-f-1} \qquad (k < n)$$

where $g_{n,k}$ = number of Git graphs counted by vertices & magenta vertices

and $\begin{bmatrix} \cdot \\ \cdot \end{bmatrix}$ = (unsigned) Stirling number of 1st kind

RANDOM MODEL

"Boltzmann model" (exponential in u, ordinary in z)

Fix $z > 0$* and $u > 0$.*

We wish to draw a Git graph γ with a weight ~~proportional~~ equal to

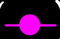
$$\frac{z^{\#\text{vertices in } \gamma}}{\tilde{G}(z,u)} \frac{u^{\#\text{magenta vertices in } \gamma}}{(\#\text{magenta vertices in } \gamma)!}$$

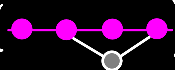
(Size is not fixed)

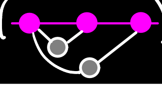
where $\tilde{G}(z,u) = \sum_{n,k \geq 0} \frac{g_{n,k}}{k!} z^n u^k = \left(1 - \frac{z^2 u}{1-z}\right)^{-\frac{1-z}{z}}.$

Examples

$\mathbb{P}($  $) = \frac{1}{\tilde{G}(z,u)}$

$\mathbb{P}($  $) = \frac{z\,u}{\tilde{G}(z,u)}$

$\mathbb{P}($  $) = \frac{z^5}{\tilde{G}(z,u)} \frac{u^4}{24}$

$\mathbb{P}($  $) = \frac{z^5}{\tilde{G}(z,u)} \frac{u^3}{6}$

*: in the disk of convergence of $\tilde{G}$

RANDOM MODEL

Proposition

Let γ be a random Git Graph sampled with respect to the previous Boltzmann model, conditioned to have size n

$$\mathbb{E}(\#\text{ magenta vertices}(\gamma)) \sim \frac{1-\rho_u}{2-\rho_u}\, n$$

$$\mathbb{V}(\#\text{ magenta vertices}(\gamma)) \sim \frac{\rho_u(1-\rho_u)}{(2-\rho_u)^3}\, n$$

where $\rho_u = \dfrac{\sqrt{1+4u}-1}{2u}$

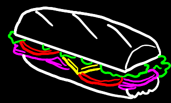
Proof: Transfer Theorem from $\tilde{G}(z,u) = \left(1 - \dfrac{z^2 u}{1-z}\right)^{-\frac{1-z}{z}}$

Consequence: A random generator for Git graphs with $\approx n$ vertices and $\simeq k$ magenta vertices $(k \leq \frac{n}{2})$

1. Tune u so that $\dfrac{1-\rho_u}{2-\rho_u} = \dfrac{k}{n}$
2. Tune z so that $z = \rho_u - \dfrac{1-\rho_u}{n}$
3. Make a Boltzmann sampler with parameters z and u for cycloriums.
4. Bijection to Git graphs

THREE ANGLES OF ATTACK

Sandwich method

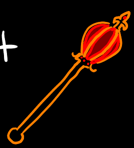


Finding subsets and supersets of Git graphs easier to count

↓

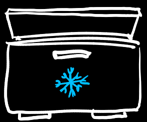
Counting Git Graphs

Ennoblement method



making the generating function less ordinary

Freezer method



Counting by (temporarily) fixing the number of magenta vertices

IT'S EASIER WHEN YOU FIX k

$$G_k(z) := \sum_{n \geq 0} g_{n,k}\, z^n$$

= Generating Function of Git graphs where the number k of magenta vertices is fixed

Claim:

$$G_k(z) = z^k \prod_{j=0}^{k-1} \left(1 + \frac{j z}{1-z}\right)$$

$$= \frac{z^{2k}}{(1-z)^k} \frac{\Gamma(k + z/(1-z))}{\Gamma(z/(1-z))}$$

Cauchy integral formula

$$g_{n,k} = \frac{1}{2\pi i} \oint \frac{z^{2k-n-1}}{(1-z)^k} \frac{\Gamma(k + z/(1-z))}{\Gamma(z/(1-z))}\, dz$$

→ invitation to saddle-point ...

ASYMPTOTIC ESTIMATE

$$g_{n,k} = \frac{1}{2\pi i} \oint \frac{z^{2k-n-1}}{(1-z)^k} \frac{\Gamma(k + z/(1-z))}{\Gamma(z/(1-z))} dz$$

NEW RESULT

Theorem The number of Git graphs with n vertices is asymptotically equivalent to

$$g_n \sim \frac{e^{1/8}}{2} \left(\frac{n}{2e}\right)^{\frac{n}{2}} \exp\left(\frac{1}{2} \ln\left(\frac{n}{2}\right)\sqrt{\frac{n}{2}} + \sqrt{2n} + \frac{\left(\ln \frac{n}{2}\right)^2}{32}\right) \left(\frac{n}{2}\right)^{-\frac{3}{8}}$$

$g_{n,k}$ is maximal for $k \approx \frac{n}{2} + \frac{1}{4} \ln\left(\frac{n}{2}\right)\sqrt{\frac{n}{2}}$

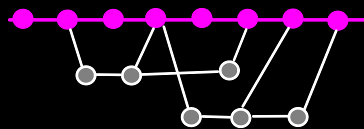
and we should have a Gaussian local limit law ...

PERSPECTIVES ABOUT RANDOM GIT GRAPHS

→ Other random models

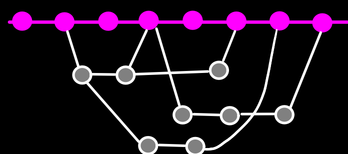
- Other graph models ▷ Collaboration with Clement + Maréchal

Phoenix graphs



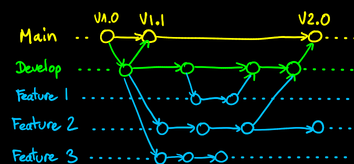
merged branches can be reborn

Fork Anywhere graphs



branches can be born anywhere but must be merged into main

More involved workflows



- more probabilistic models

LINKS BETWEEN GIT GRAPHS & GIT BISECT

<u>Still to do</u> : Average-case complexity of git bisect where the input is taken w.r.t the Boltzmann distribution.

<u>But also</u> : - Is there a polynomial algorithm for the Regression Problem when the input is a Git graph?

(We proved that git bisect fails to be optimal for some Git graphs)

- Is the Regression Problem NP-complete when the input is binary?

NEW RESULT [Bulteau +] : No (if ETH is true)

- Other algorithms from Version Control Systems to be analyzed?

THANK YOU!