Refining Conjectures via Proof-Based Generalization

Anshula Gandhi

Timothy Gowers

Anand Rao Tadipatri

The University of Cambridge

Refining Conjectures

via

Proof-Based Generalization

What is "Proof-Based Generalization"?

As mathematicians, we typically look back over what we have proven, and see if it lends itself to **some straightforward generalization — one that doesn't really require modification of the proof.**

Example: When we look at the standard proof that

$$\sqrt{2} \textbf{ is irrational,}$$

we can quickly notice the "same proof" would work if 2 was replaced by any prime. That is, we run a *proof-based generalization* on it to yield

$$\forall \textbf{ primes } p, \sqrt{p} \textbf{ is irrational.}$$

What is “Proof-Based Generalization”?

As mathematicians, we typically look back over what we have proven, and see if it lends itself to **some straightforward generalization — one that doesn’t really require modification of the proof.**

Example: When we look at the standard proof that

$$\sqrt{2} \textbf{ is irrational,}$$

we can quickly notice the “same proof” would work if 2 was replaced by any prime. That is, we run a *proof-based generalization* on it to yield

$$\forall \textbf{ primes } p, \sqrt{p} \textbf{ is irrational.}$$

Proof-Based Generalization := a generalization of a proof in which the hypotheses are weakened as much as the proof will allow.

What is "Proof-Based Generalization"?

But from the standard proof that $\sqrt{2}$ is irrational, it is more difficult to see that:

$$\forall p, p \text{ is not a perfect square} \implies \sqrt{p} \text{ is irrational.}$$

So, we would *not* consider the above a *proof-based generalization.*

Refining Conjectures

via

Proof-Based Generalization

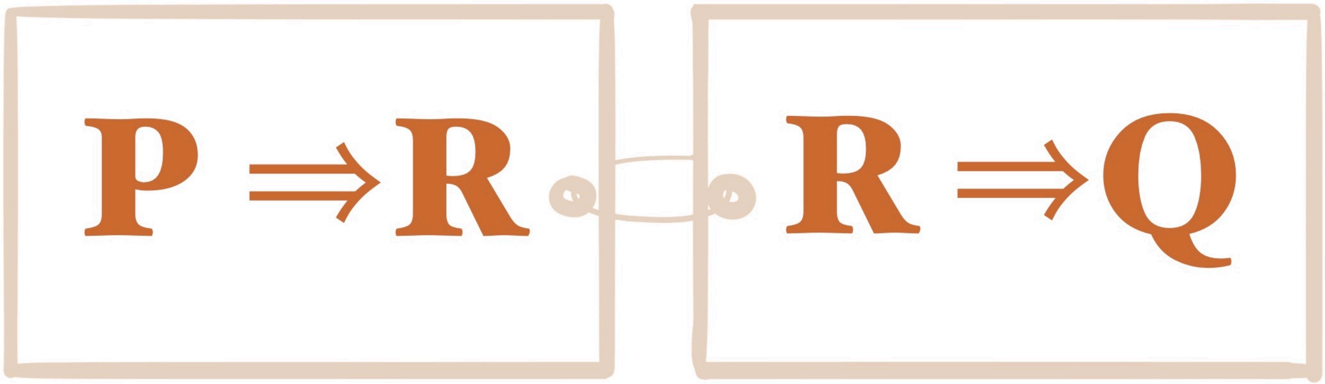
How Do We Refine Conjectures?

When people think of conjectures, they tend to think of big open problems (e.g. P = NP). But conjecturing also happens in research on a day-to-day basis — especially when **conjecturing an intermediate statement**.

$$\mathbf{P} \Rightarrow \mathbf{Q}$$

How Do We Refine Conjectures?

When people think of conjectures, they tend to think of big open problems (e.g. P = NP). But conjecturing also happens in research on a day-to-day basis — especially when **conjecturing an intermediate statement**.



When we do this, we are implicitly conjecturing both $P \implies R$ and $R \implies Q$. And we often must *refine* this R until it is "just right" (that is, proving $P \implies R$ and $R \implies Q$ is easier than proving $P \implies Q$).

In this talk, we will discuss a method for refining R toward this goal.

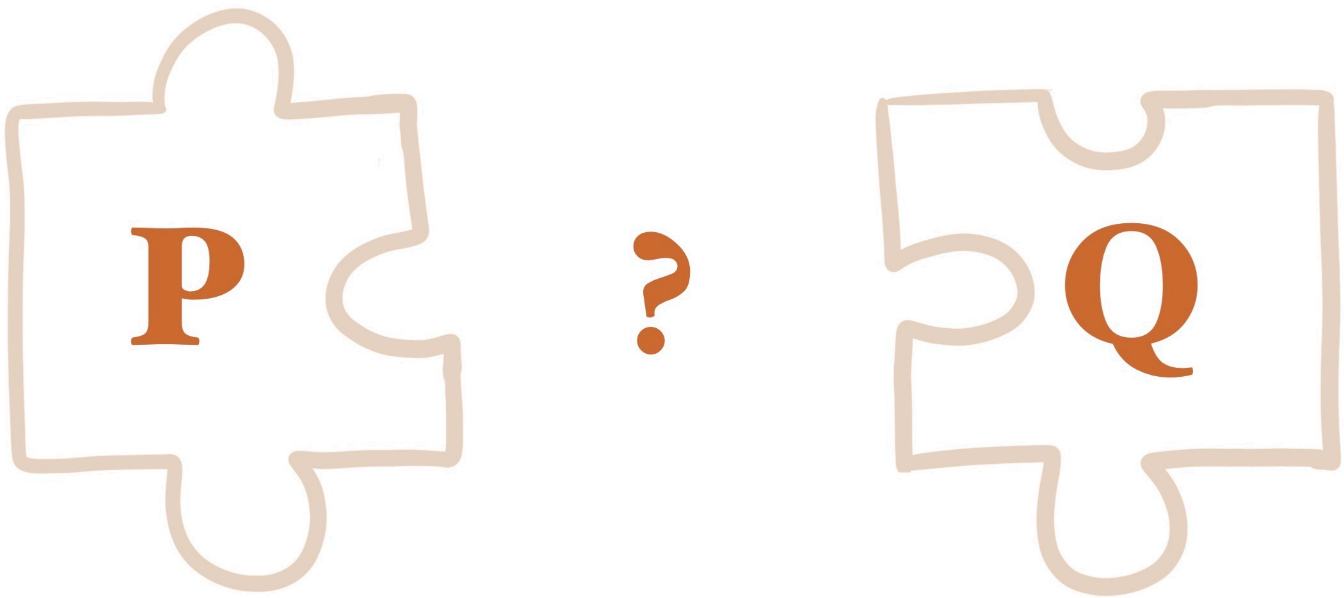
What do they have to do with each other?

Refining Conjectures

Proof-Based Generalization

Finding Intermediate Statements

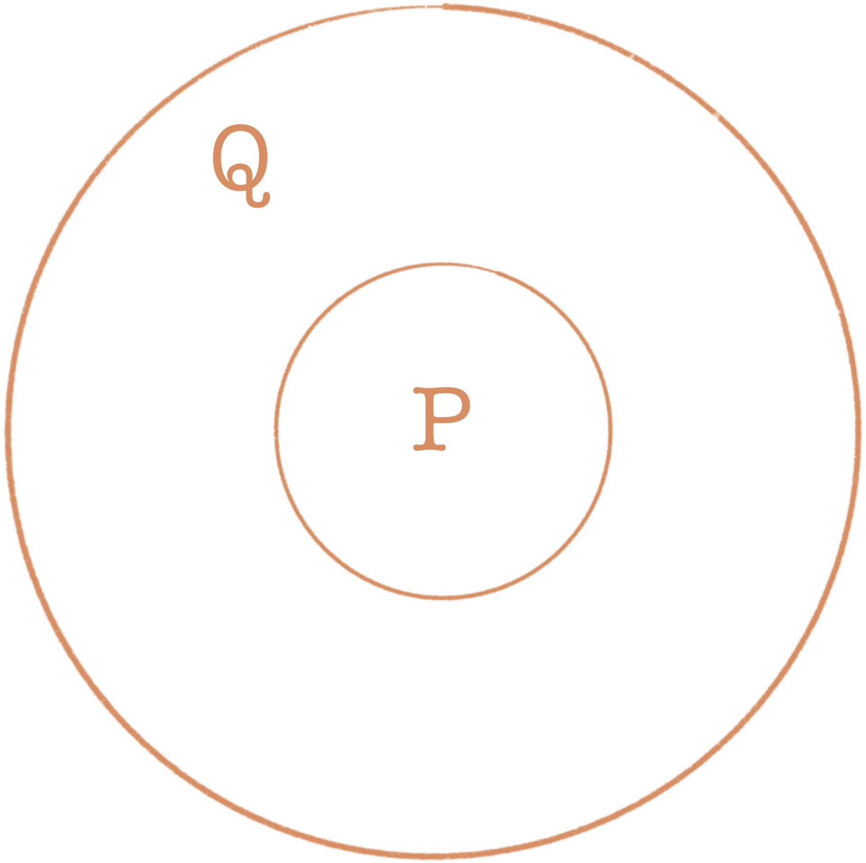
Well, **coming up with a suitable intermediate statement is hard**.



It turns out proof-based generalization can help. Here's how...

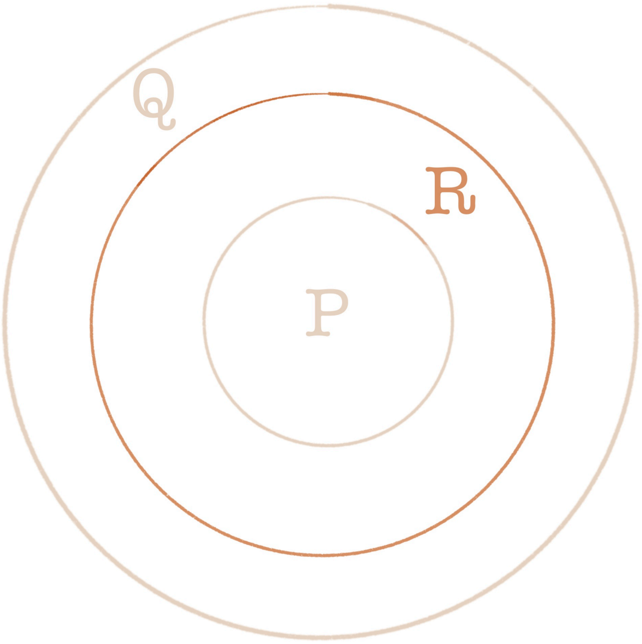
How To Find Intermediate Statements

Suppose we want to prove some statement $\forall x, P(x) \implies Q(x)$.



How To Find Intermediate Statements

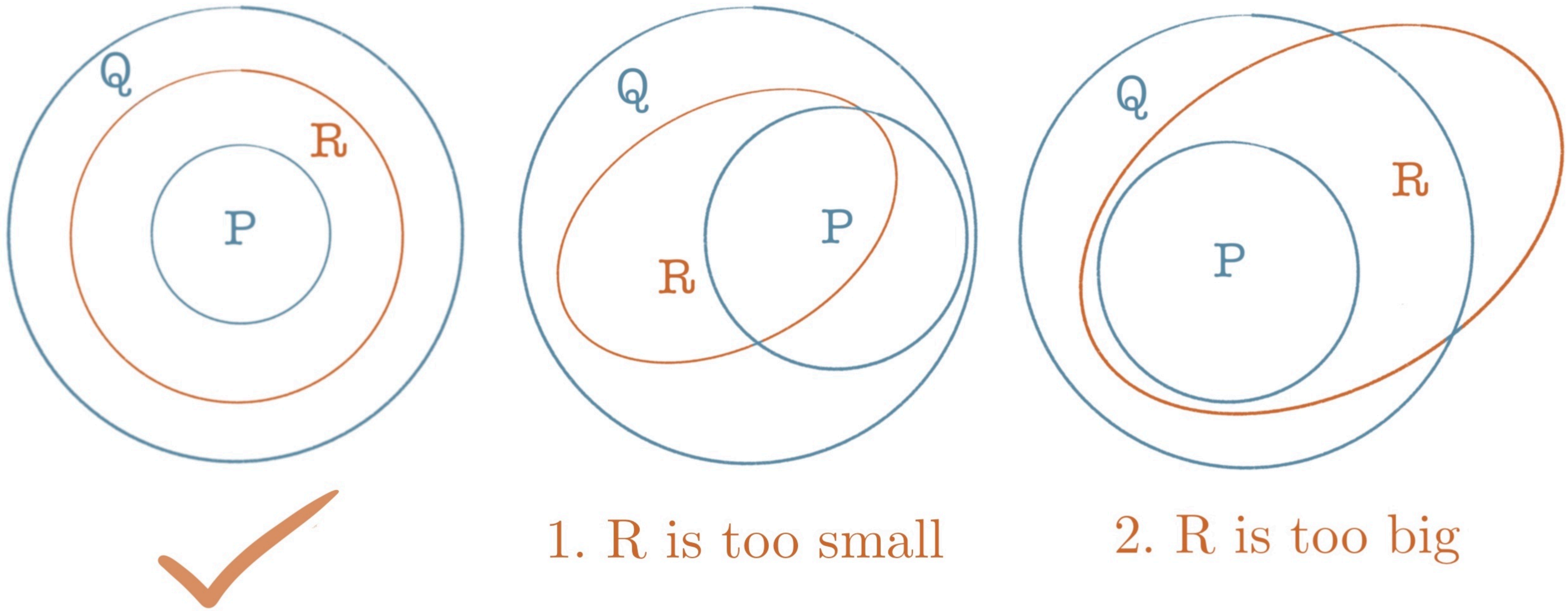
Our work focuses on the following two ways of generating an intermediate statement R: by **weakening** the hypothesis P, or by **strengthening** the conclusion Q.



And while we might luck out and immediately find some R such that $\forall x, P(x) \implies R(x) \implies Q(x)$...

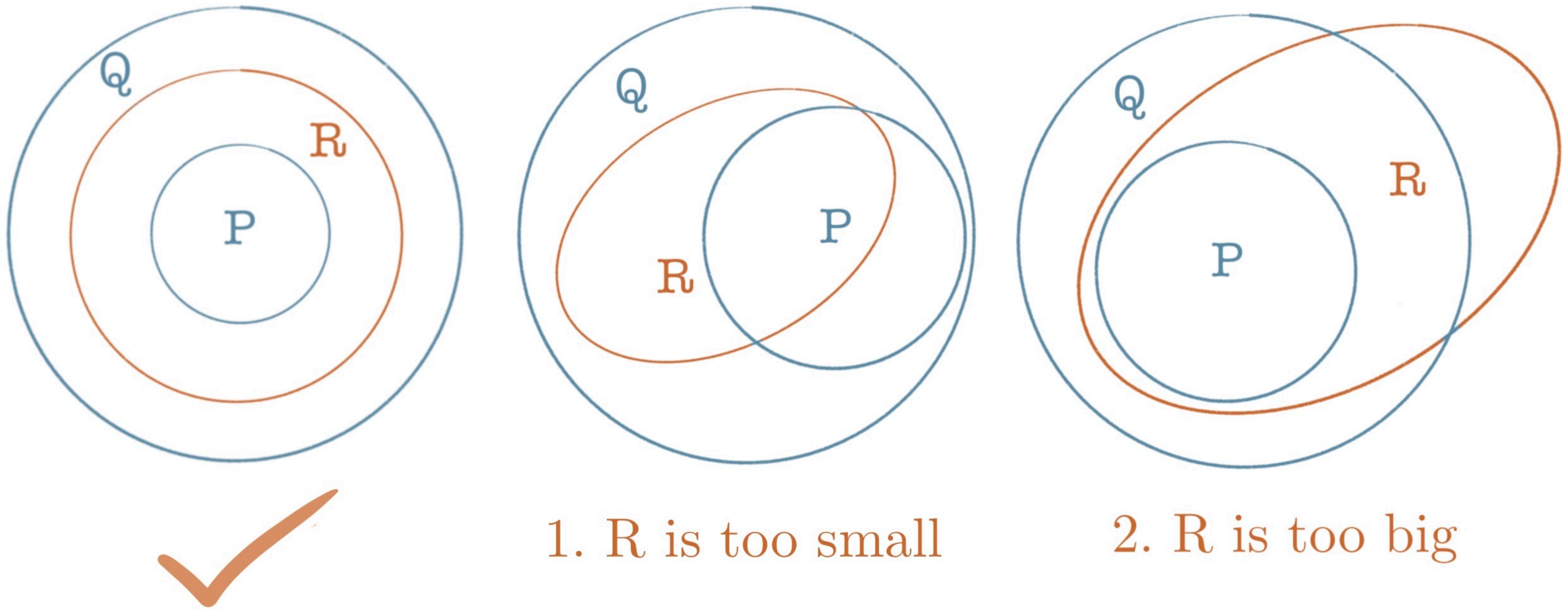
How To Find Intermediate Statements

...there are two ways in which we can fail:



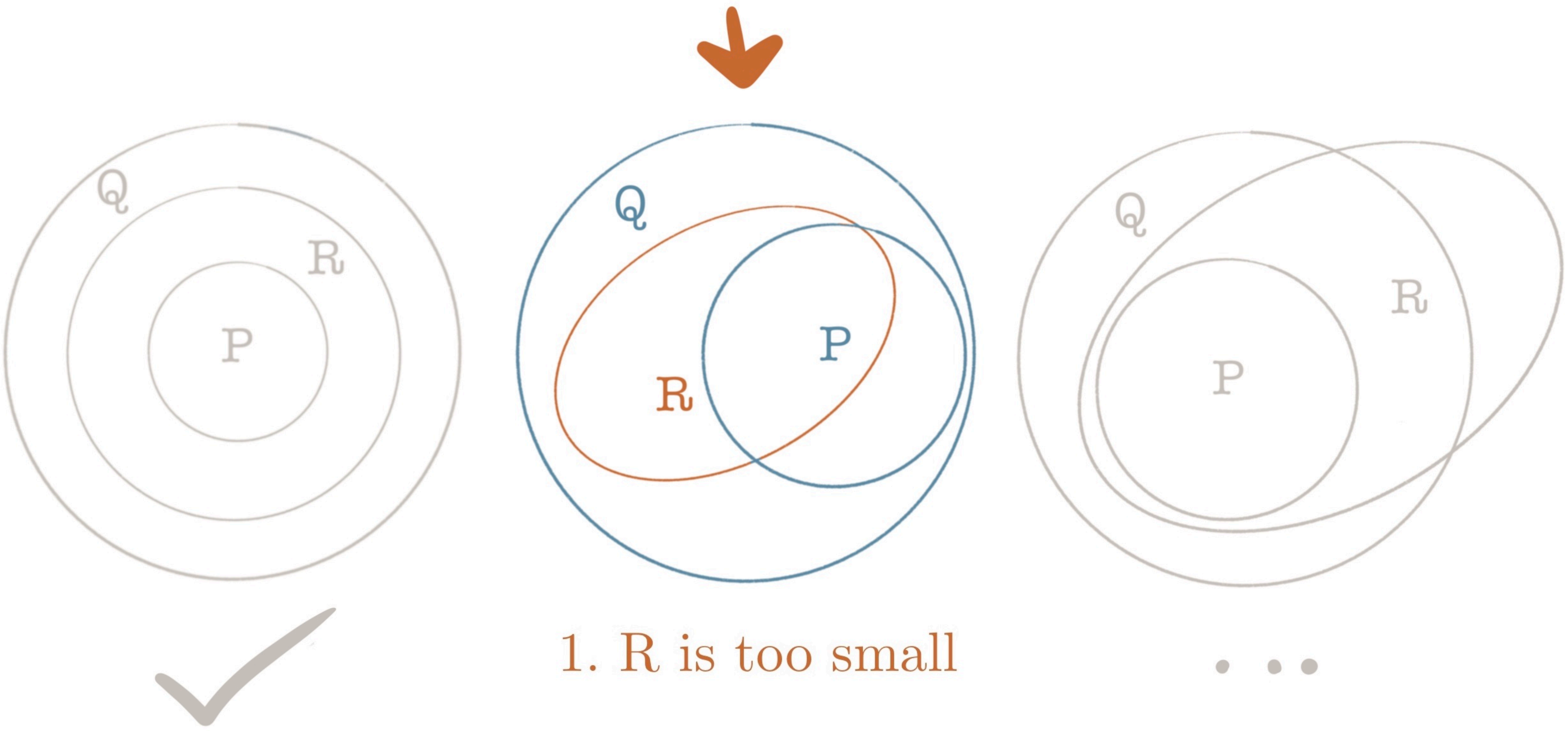
How To Find Intermediate Statements

But there are two ways in which we can fail:



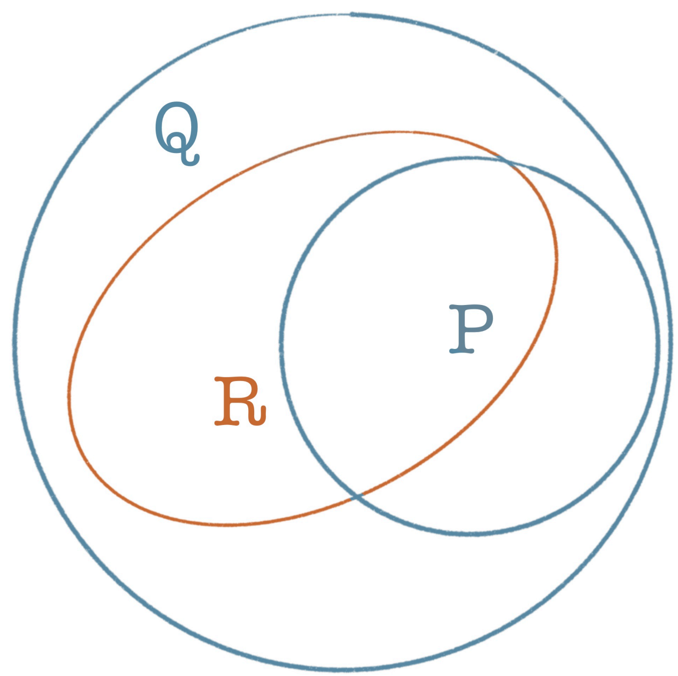
Our work focuses on what to do in these two cases.

If R is "too small"...



If R is "too small"...

Suppose we create an initial intermediate statement R by **strengthening** the conclusion Q.

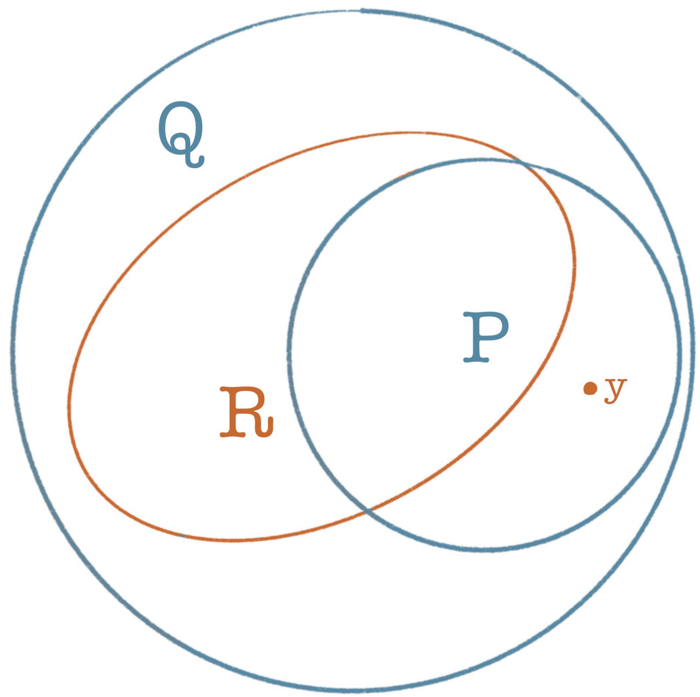


Then, we have that $R \implies Q$, but it is not obvious whether $P \implies R$:

$$P \overset{?}{\implies} R \implies Q$$

If R is "too small"...

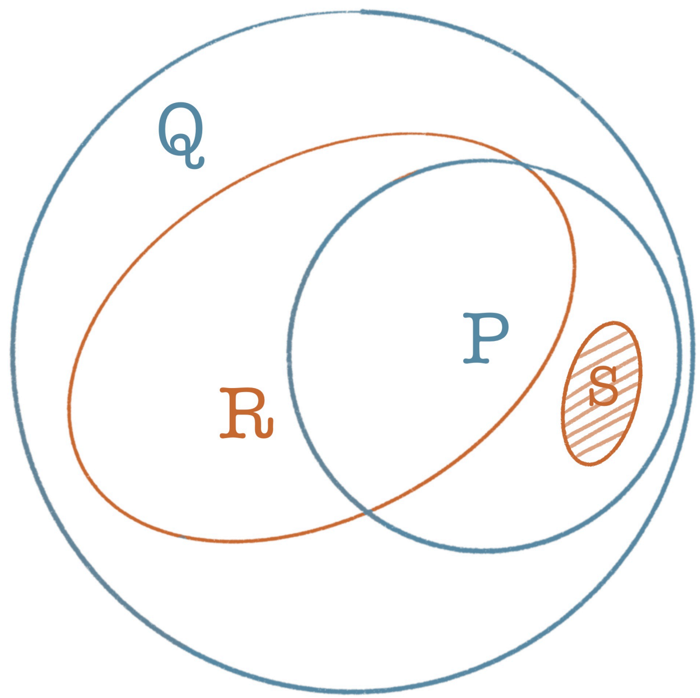
But now suppose we discover that $P \not\Longrightarrow R$ by constructing a counterexample.



$$\exists y, P(y) \land \neg R(y)$$

If R is "too small"...

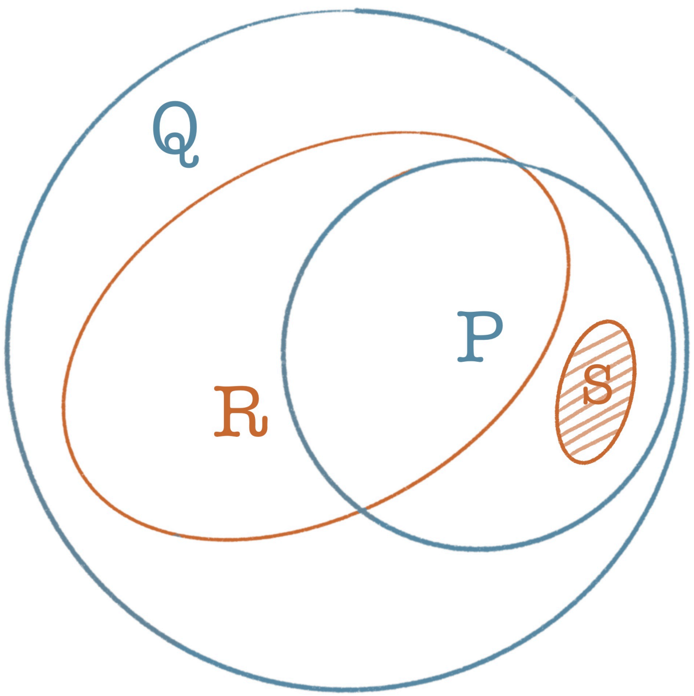
It often helps us to generalize the counterexample y to a class of counter-examples S. That is:



$$\forall y, S(y) \implies P(y) \land \neg R(y)$$

If R is "too small"...

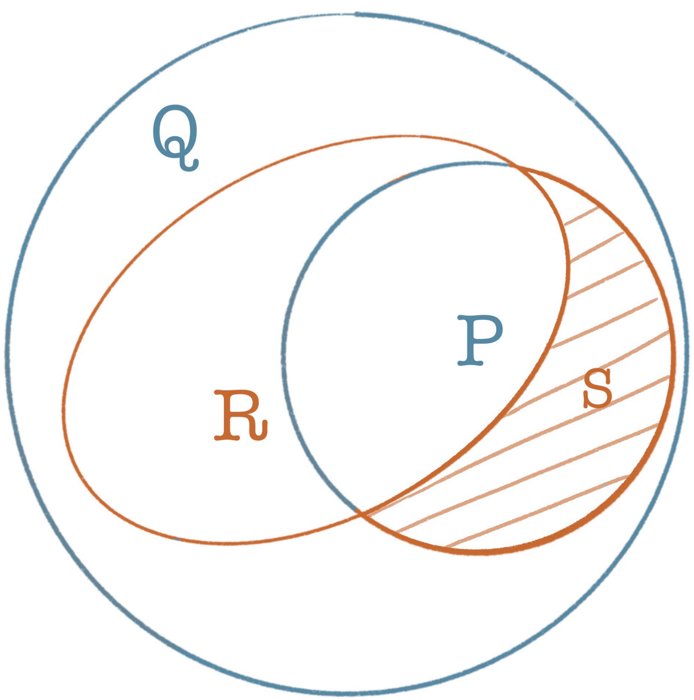
So, we use **proof-based generalization** on the statement that y is a counterexample, together with its proof, to obtain a class S.



$$\forall y, S(y) \implies P(y) \land \neg R(y)$$

If R is "too small"...

We then hope the converse of $S \implies P \wedge \neg R$ is true as well (which means we have found the most general class of counterexamples), we have:

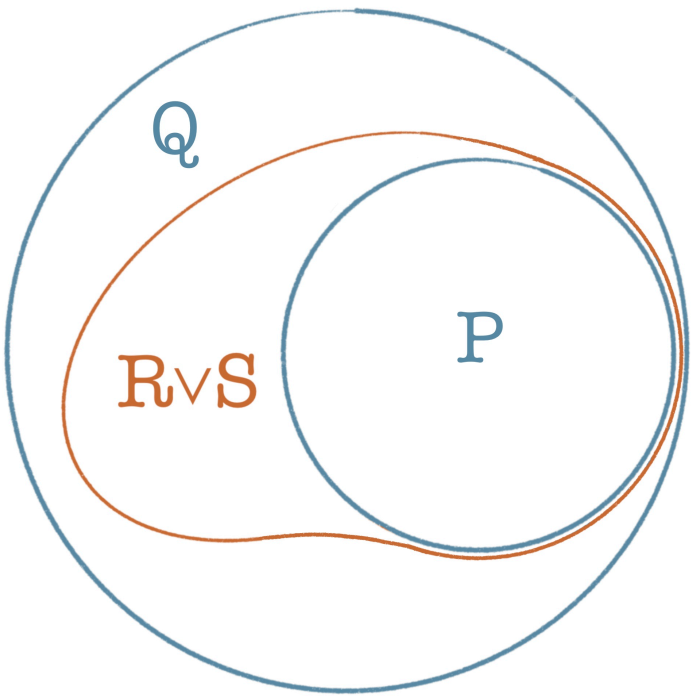


$$\forall y, S(y) \iff P(y) \wedge \neg R(y)$$

That is, we have determined, in some sense, the "entire reason" why $P \not\Rightarrow R$...

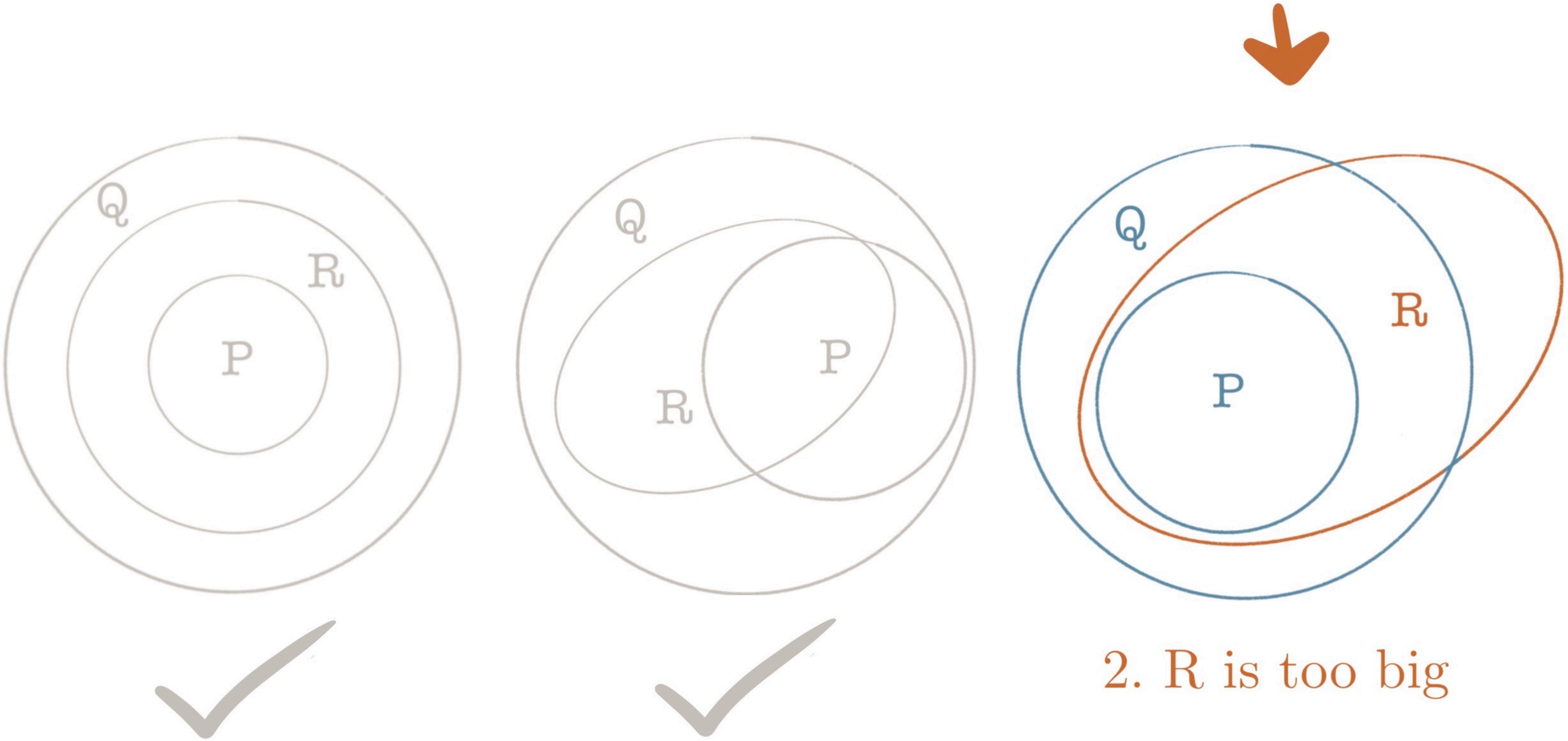
If R is "too small"...

...which means a new candidate for an intermediate statement is $R \vee S$, since:



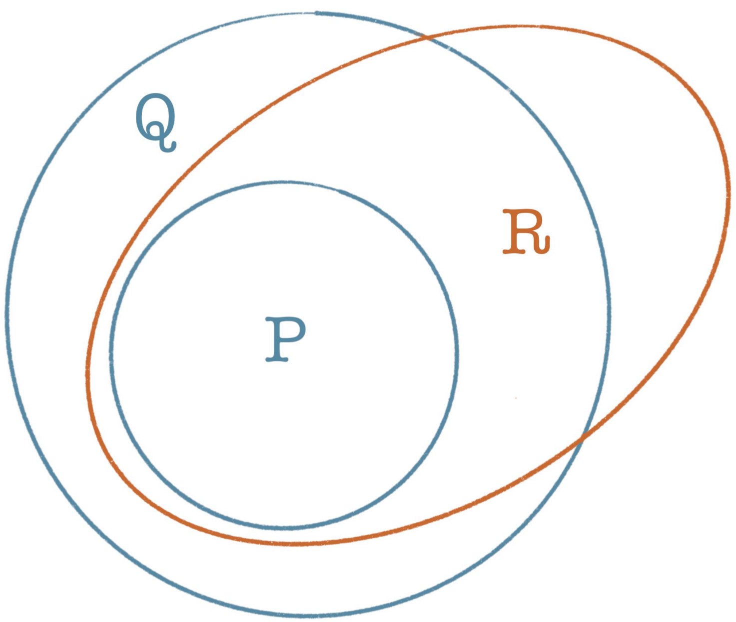
$$P \implies R \vee S \implies Q$$

If R is "too big"...



If R is "too big"...

Suppose we make the initial intermediate statement by **weakening** the hypothesis P.

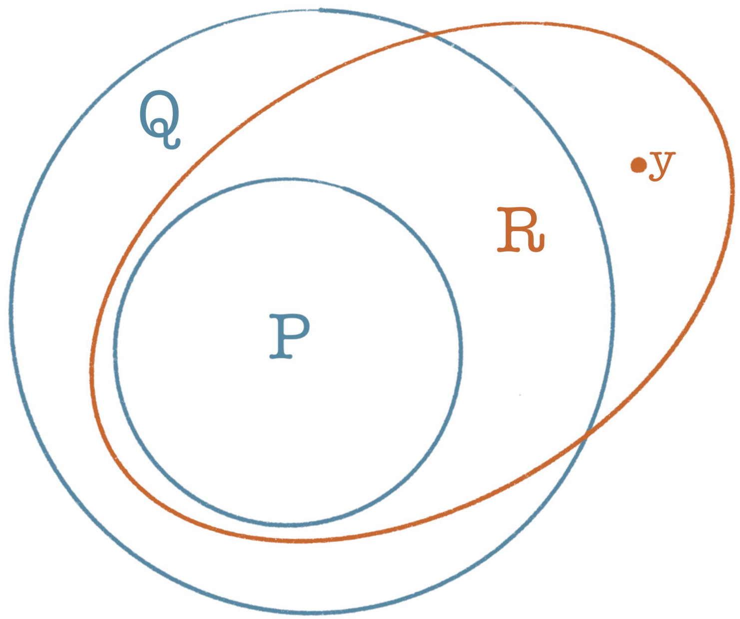


Then, we have that $P \implies R$, but it is not obvious whether $R \implies Q$:

$$P \implies R \overset{?}{\implies} Q$$

If R is "too big"...

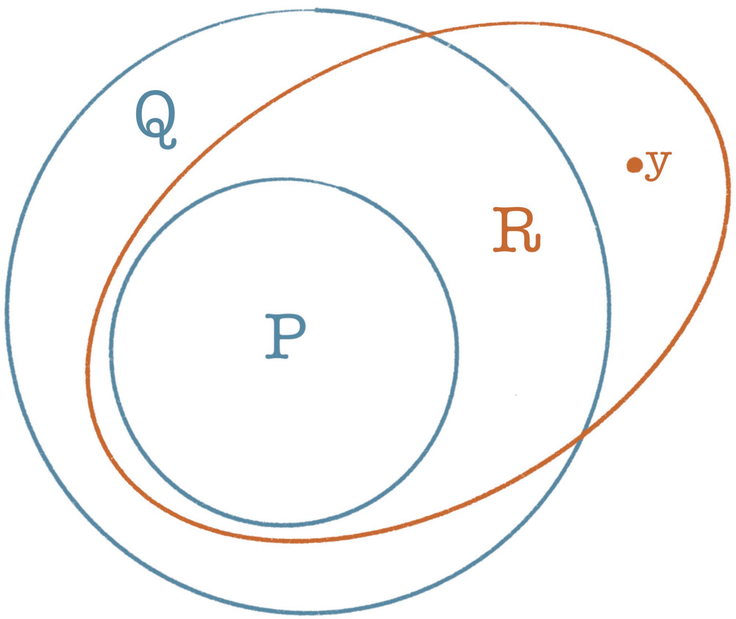
Suppose we end up proving that $R \not\Longrightarrow Q$ by constructing a counterexample.



$$\exists y, R(y) \wedge \neg Q(y)$$

If R is "too big"...

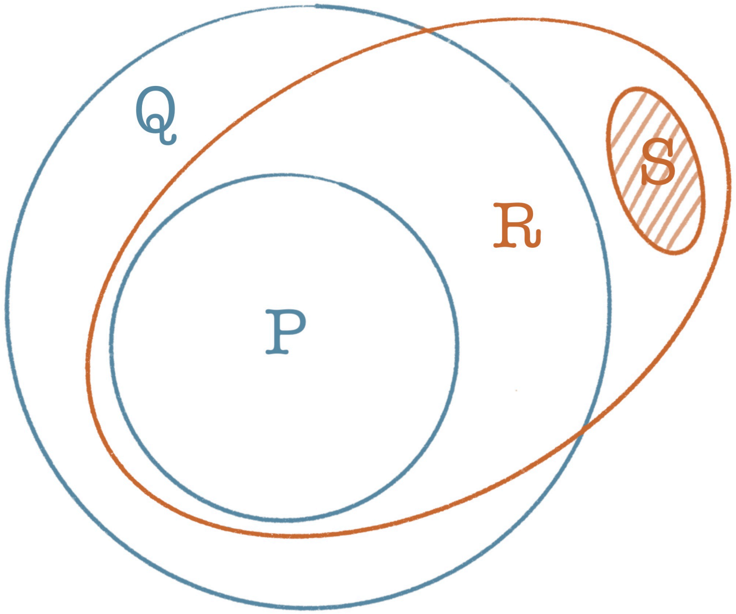
Again, we would like to eliminate the reason R doesn't imply Q.



$$\exists y, R(y) \land \neg Q(y)$$

If R is "too big"...

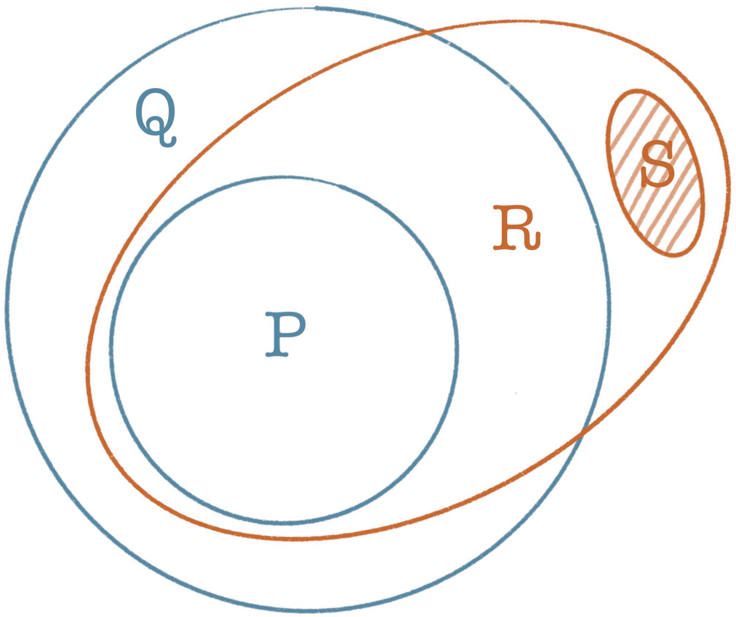
So, we use **proof-based generalization** on the statement that y is a counterexample, together with its proof, to obtain a class S.



$$\forall y, S(y) \implies R(y) \land \neg Q(y)$$

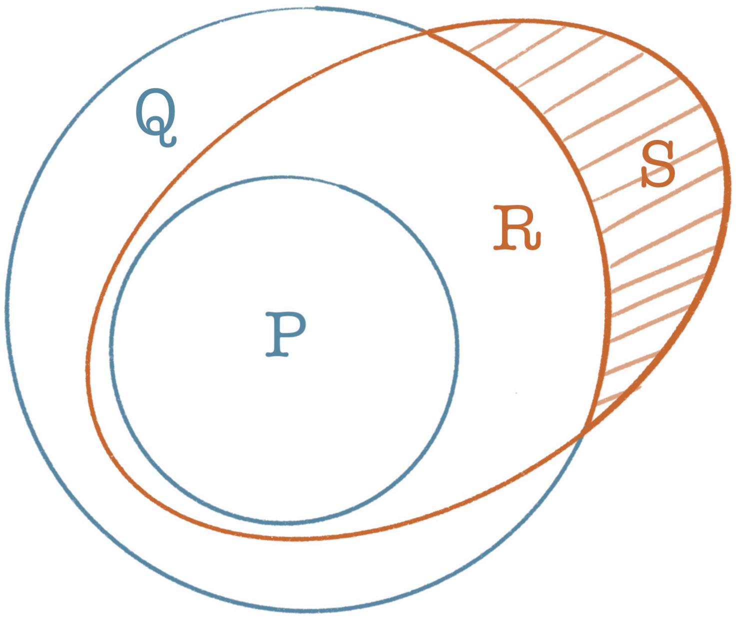
If R is "too big"...

We then hope that we have actually found the most general class of counterexamples to $R \not\Longrightarrow Q$...



If R is "too big"...

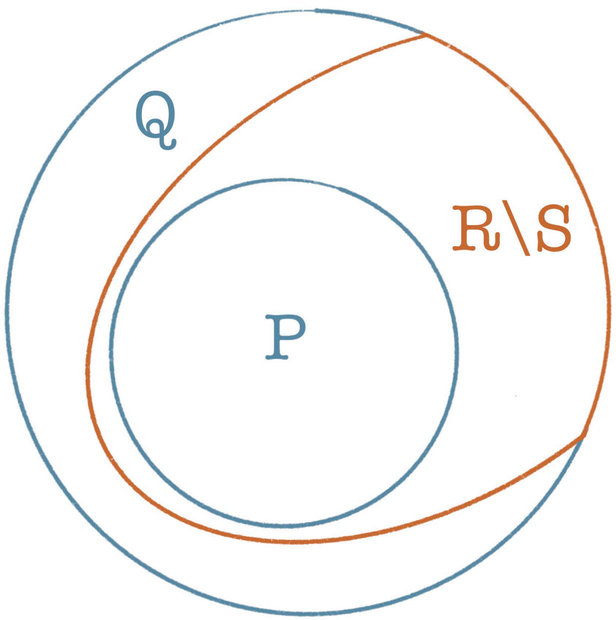
...so, in particular, we hope that the converse is also true. This would mean we have found the “entire reason” that $R \not\Longrightarrow Q$.



$$\forall y, S(y) \iff R(y) \land \neg Q(y)$$

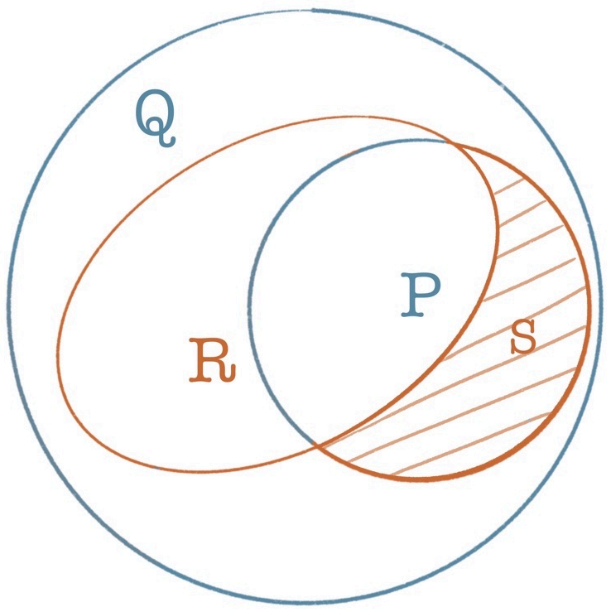
If R is "too big"...

Consequently, a new candidate for an intermediate statement is $R \wedge \neg S$ or equivalently $R \setminus S$.



Iterative Conjecture Refinement

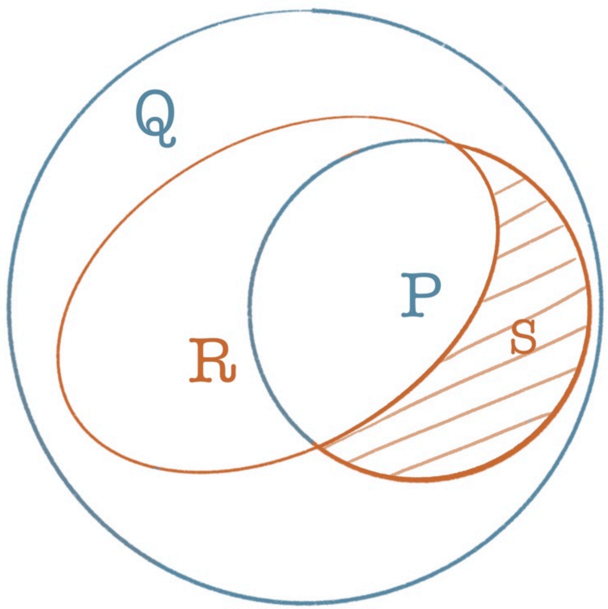
Problem: But...what if we don't immediately find the "entire reason" the implication doesn't hold?



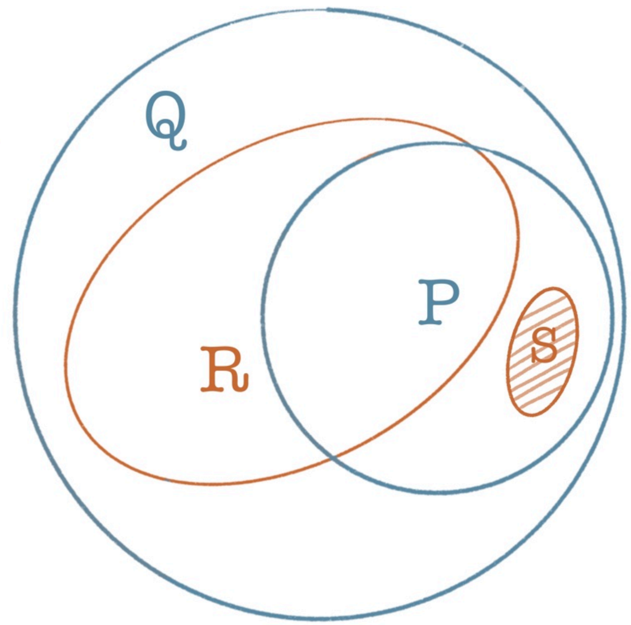
What if instead of this...

Iterative Conjecture Refinement

Problem: But...what if we don't immediately find the "entire reason" the implication doesn't hold?



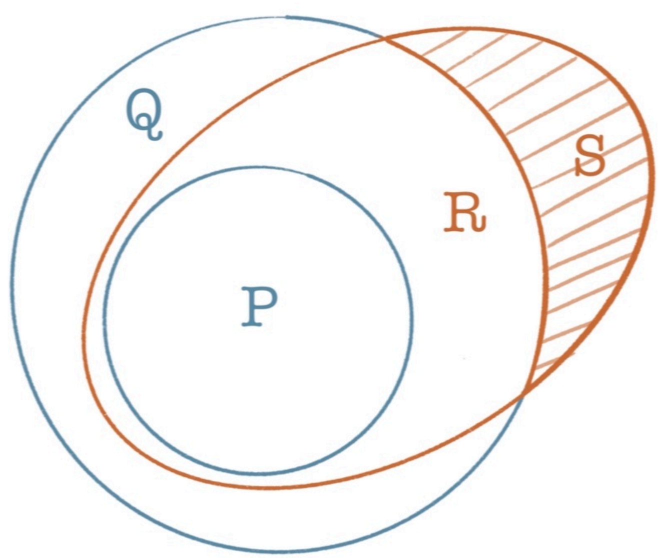
What if instead of this...



we have this?

Iterative Conjecture Refinement

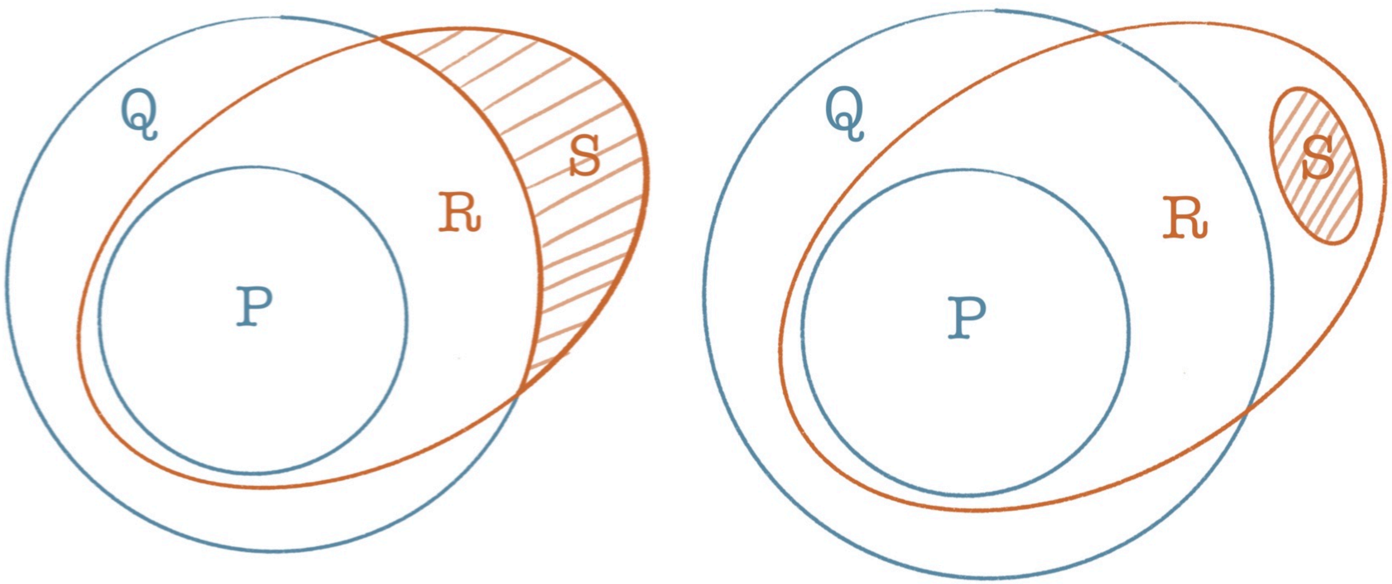
Problem: But...what if we don't immediately find the "entire reason" the implication doesn't hold?



What if instead of this...

Iterative Conjecture Refinement

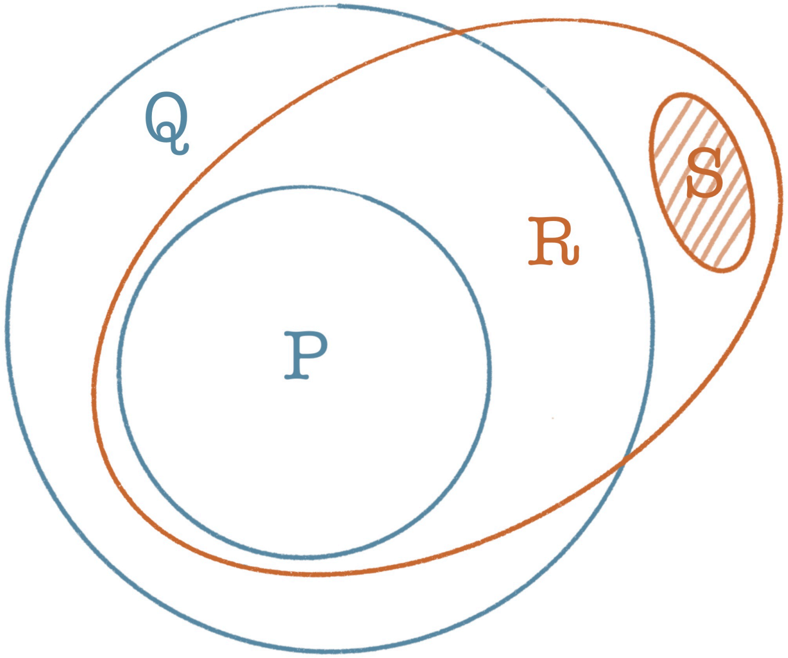
Problem: But...what if we don't immediately find the "entire reason" the implication doesn't hold?



What if instead of this... we have this?

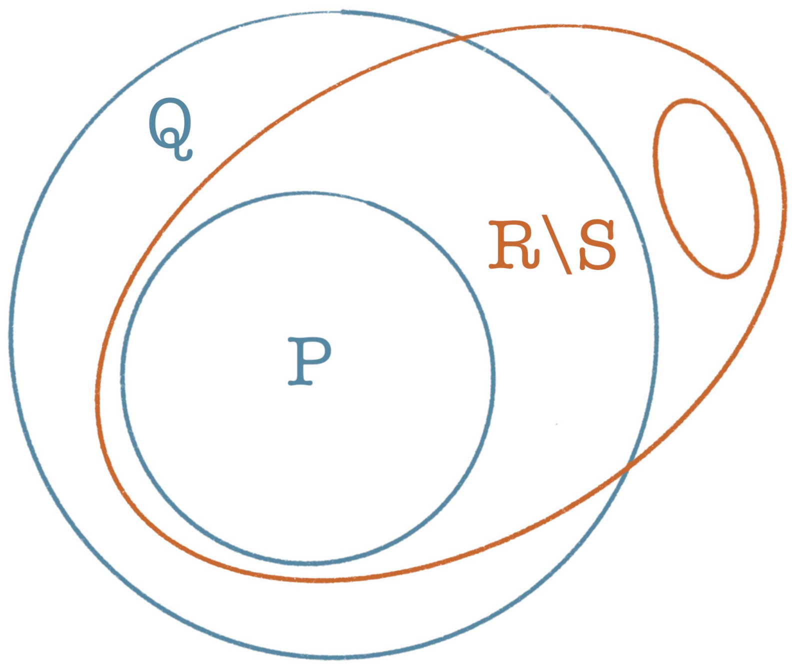
Iterative Conjecture Refinement

Solution: If the class of counterexamples S is not big enough...



Iterative Conjecture Refinement

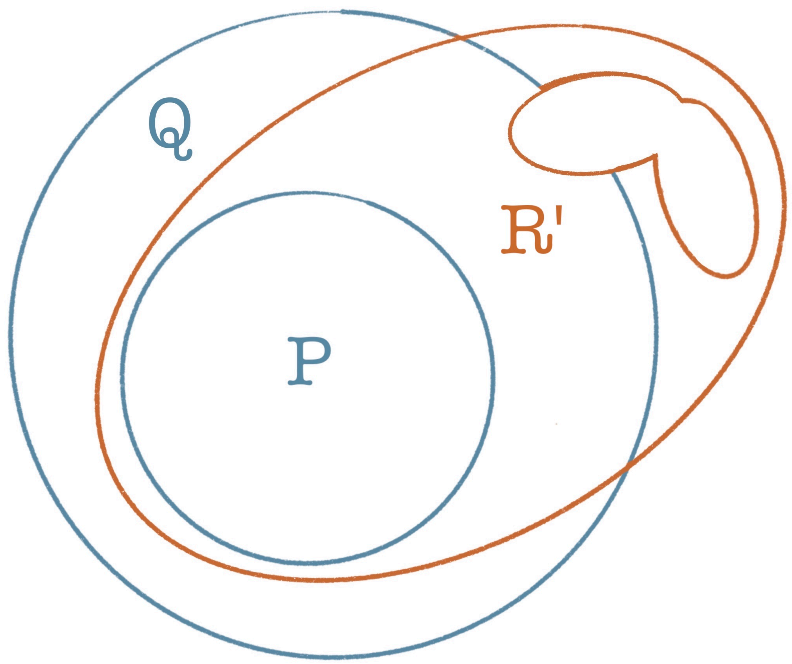
Solution: If the class of counterexamples S is not big enough, we can repeat the refinement process on the new intermediate statement.



(We couldn't eliminate the *entire* reason $R \not\Longrightarrow Q$, but we could eliminate part of it).

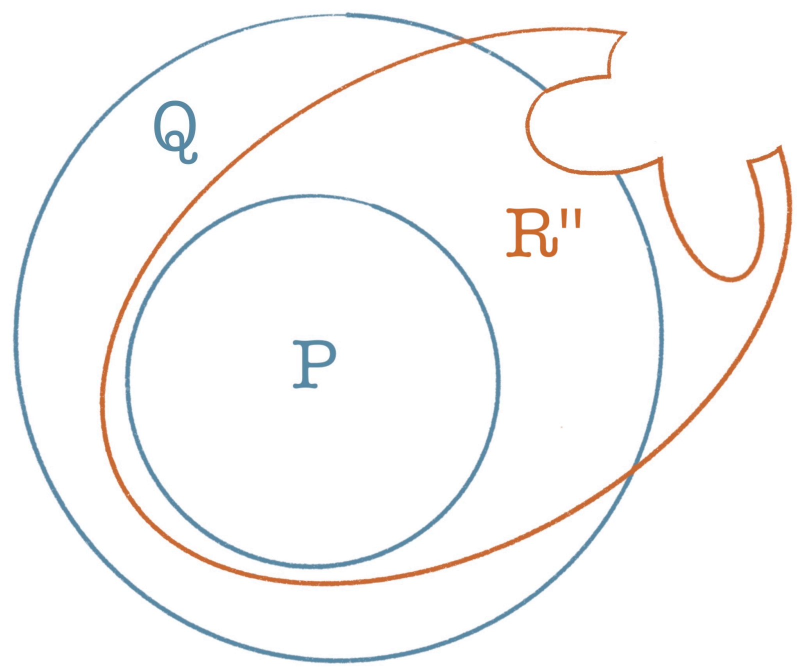
Iterative Conjecture Refinement

Solution: If the class of counterexamples S is not big enough, we can repeat the refinement process on the new intermediate statement.



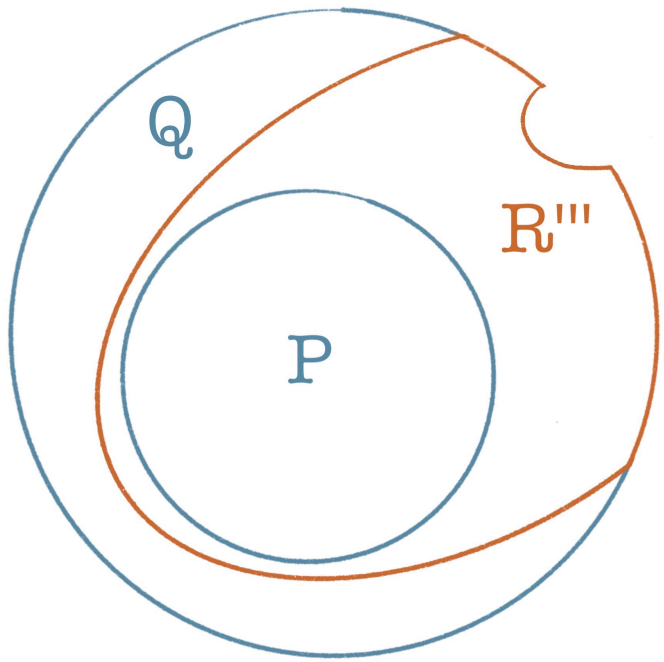
Iterative Conjecture Refinement

Solution: If the class of counterexamples S is not big enough, we can repeat the refinement process on the new intermediate statement.



Iterative Conjecture Refinement

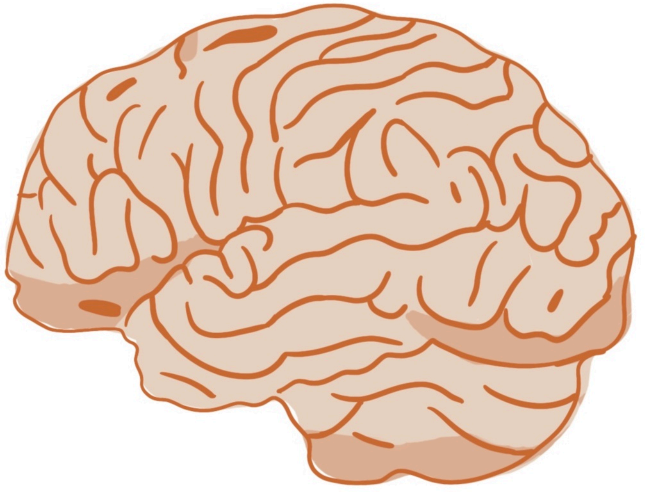
Solution: If the class of counterexamples S is not big enough, we can repeat the refinement process on the new intermediate statement.



Eventually, we have: $P \implies R''' \implies Q$.

Conjecture Refinement, Diagrammatically

These diagrams provide an explanation for **why** we have the **intuitions** we do as mathematicians about how to conjecture and how to adapt our conjecturing approaches.



Is there a concrete example of this approach in action?

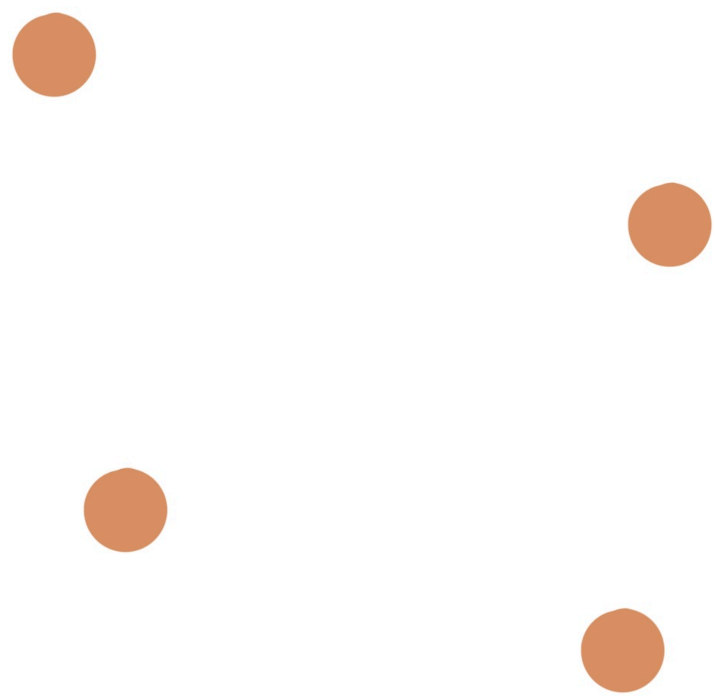
An Example of Conjecture Refinement

I have asked professors, graduate students, undergraduate students, and non-mathematicians the following question.

Almost everyone who discovered the proof used more or less the same process of conjecture generation and refinement.

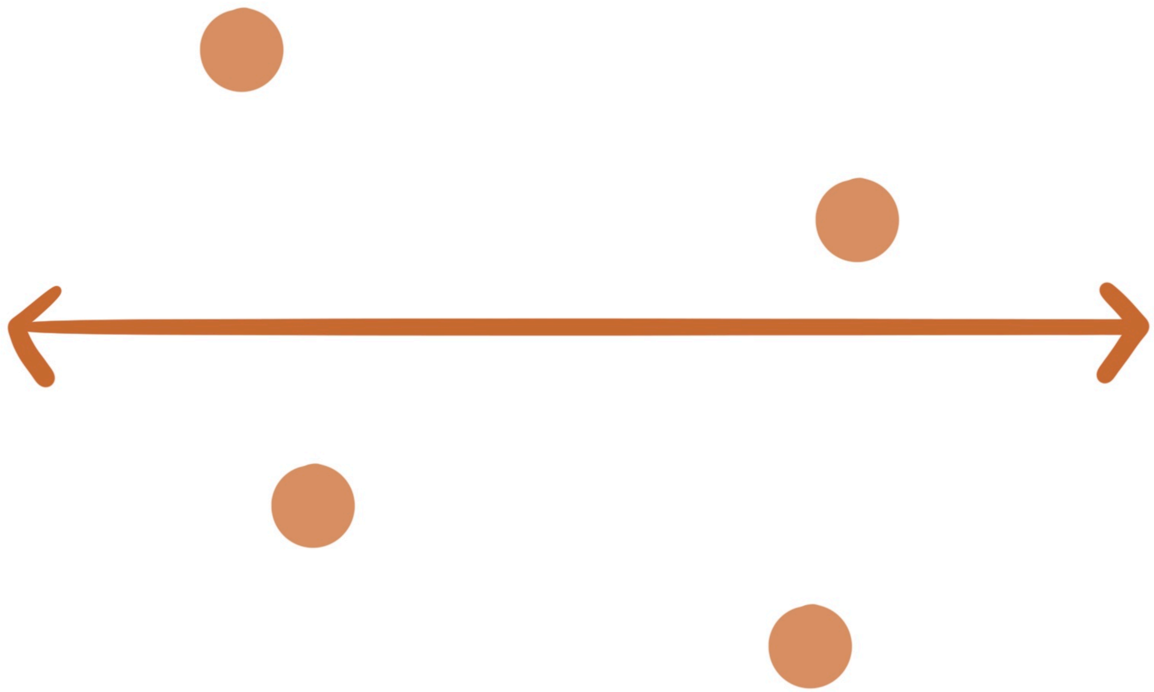
An Example of Conjecture Refinement

Given $2n$ points on a plane, does there always exist a line such that n points are strictly on one side of the line, and n strictly on the other?



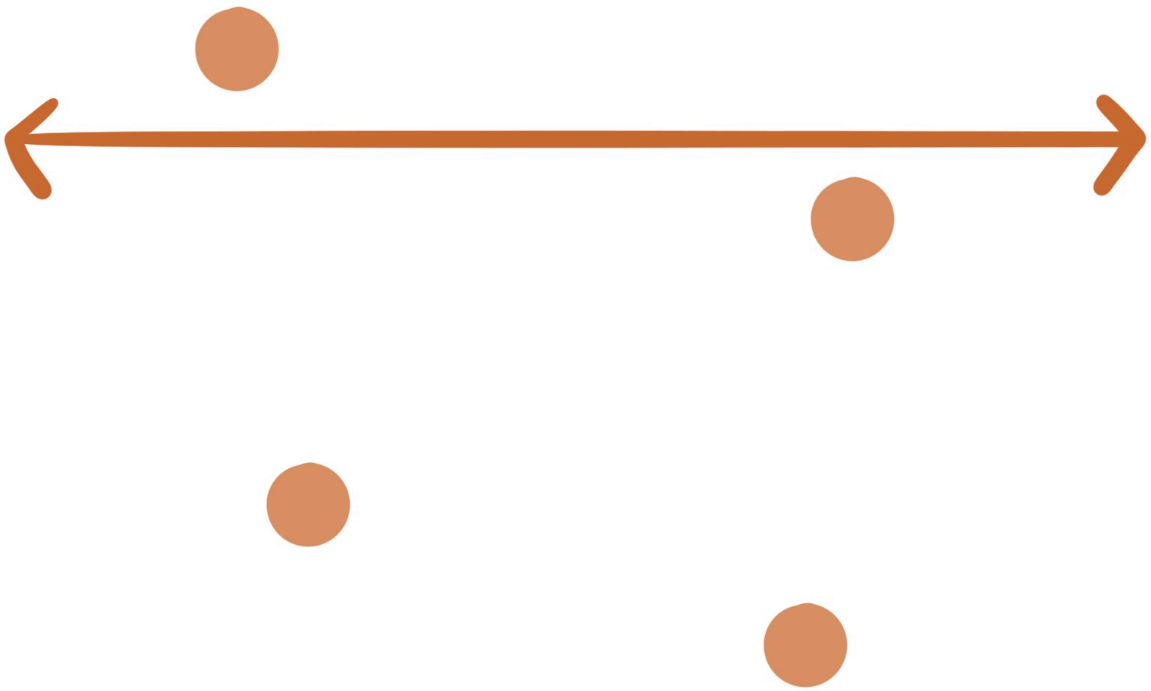
An Example of Conjecture Refinement

Given $2n$ *points on a plane, does there always exist a line such that* n *points are strictly on one side of the line, and* n *strictly on the other?*



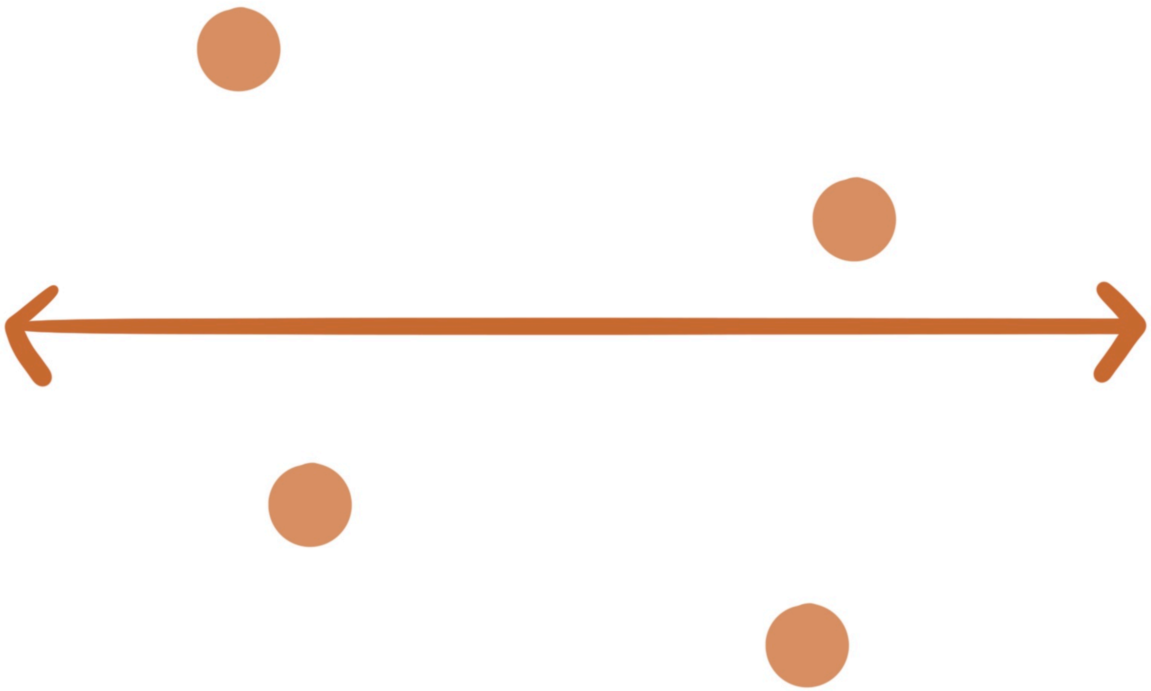
An Example of Conjecture Refinement

A reasonable first conjecture is: Any line, translated appropriately, should do the trick.



An Example of Conjecture Refinement

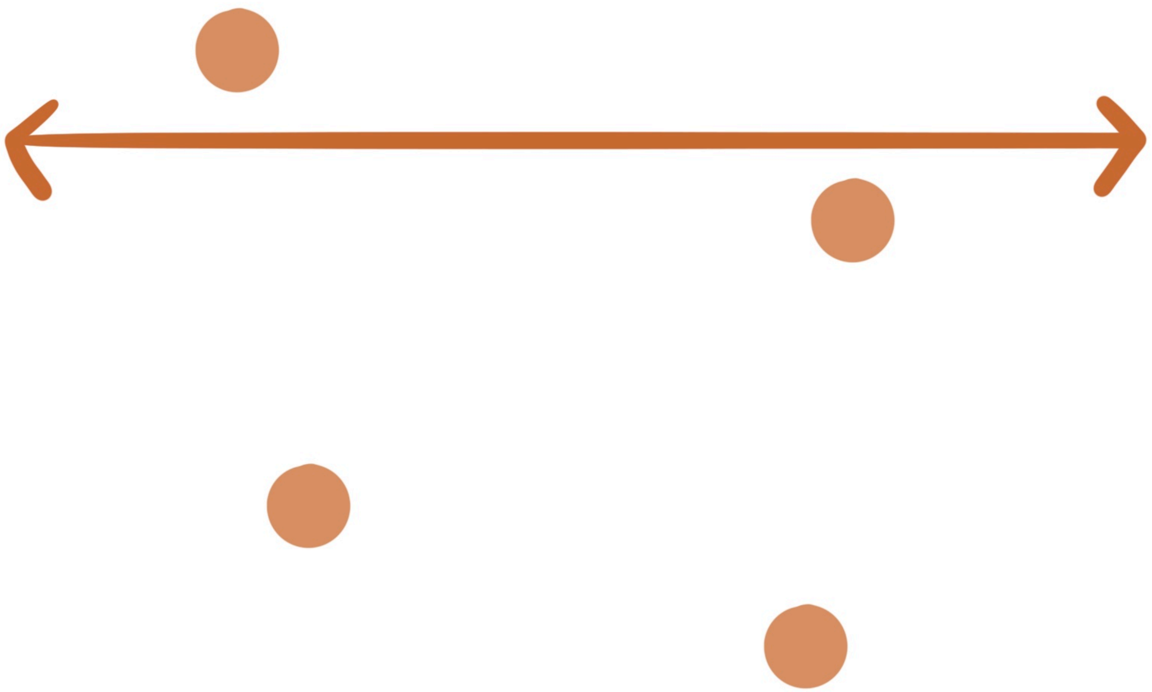
A reasonable first conjecture is: Any line, translated appropriately, should do the trick.



In particular, a horizontal line (appropriately translated) should always work. (This isn't a particularly "clever" conjecture...it is a straightforward strengthening of the conclusion).

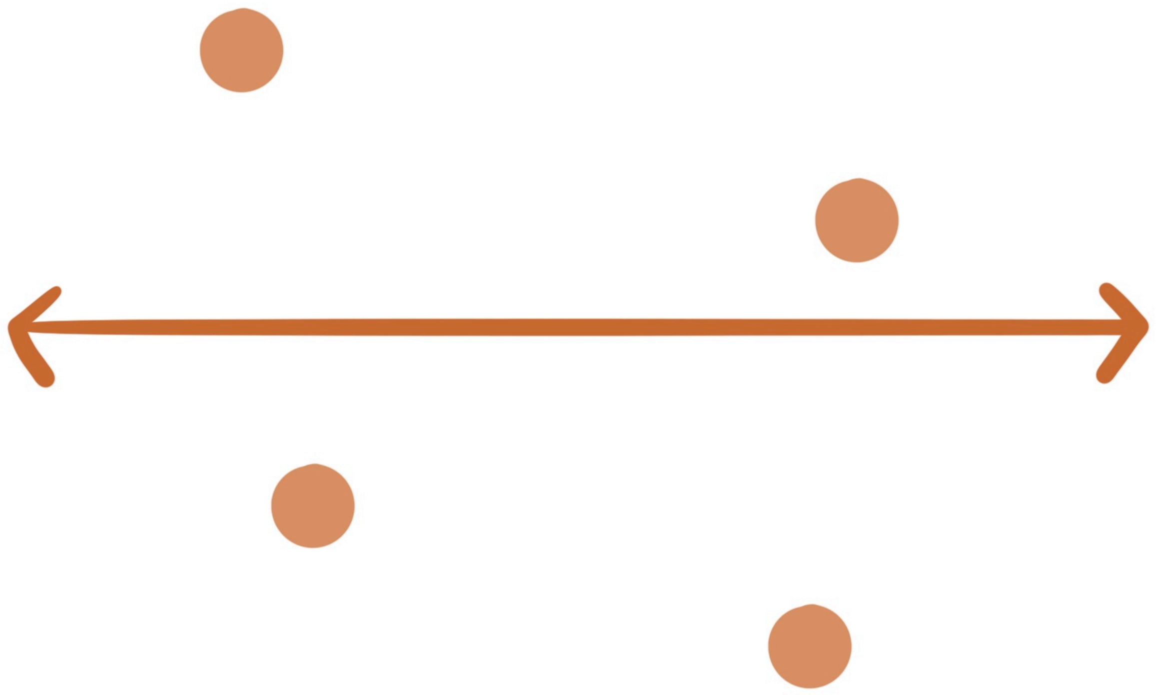
An Example of Conjecture Refinement

Implicitly, we are **conjecturing** the following: A moving horizontal line will pass through one point at a time. So, appropriately translated, it will eventually bisect the set. We can refer to this as the “discrete intermediate value theorem” or “discrete IVT.”



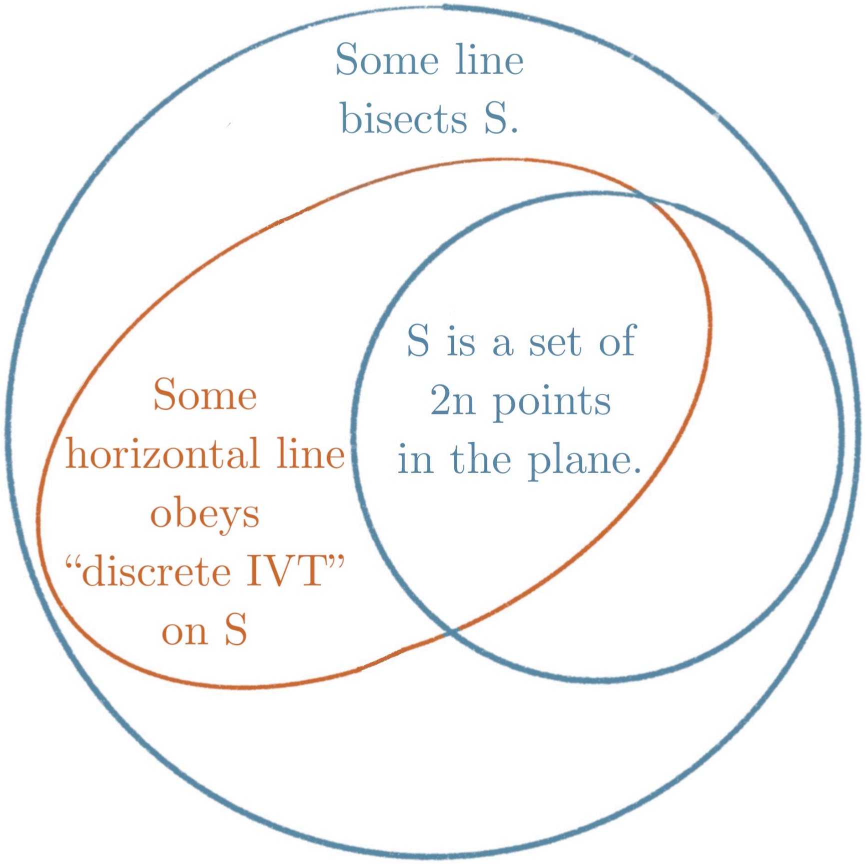
An Example of Conjecture Refinement

Implicitly, we are **conjecturing** the following: A moving horizontal line will pass through one point at a time. So, appropriately translated, it will eventually bisect the set. We can refer to this as the "discrete intermediate value theorem" or "discrete IVT."



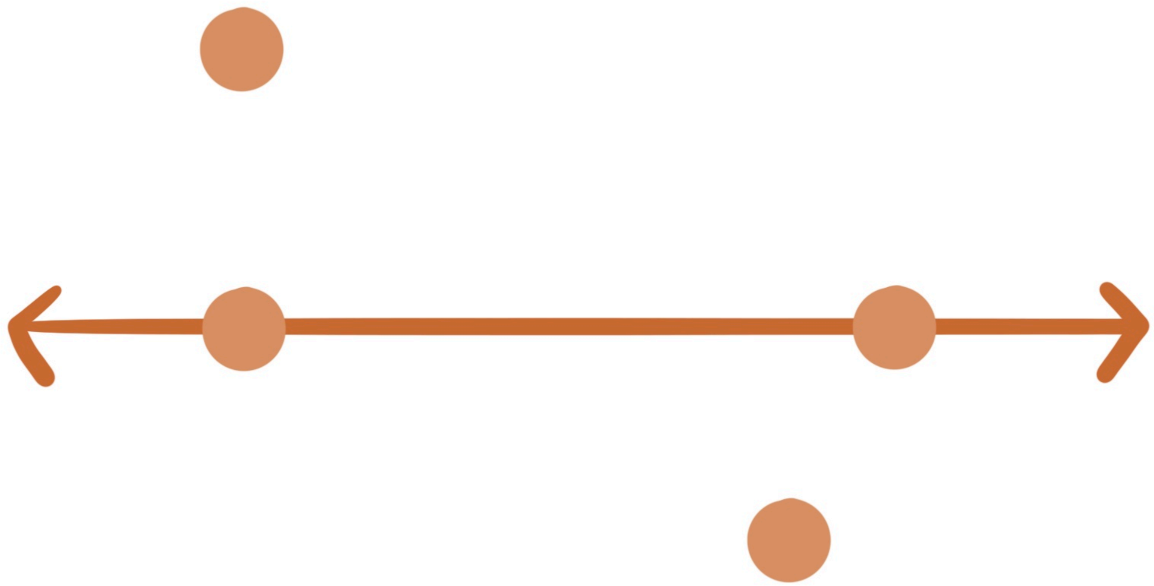
An Example of Conjecture Refinement

Implicitly, we are **conjecturing** the following: A moving horizontal line will pass through one point at a time.



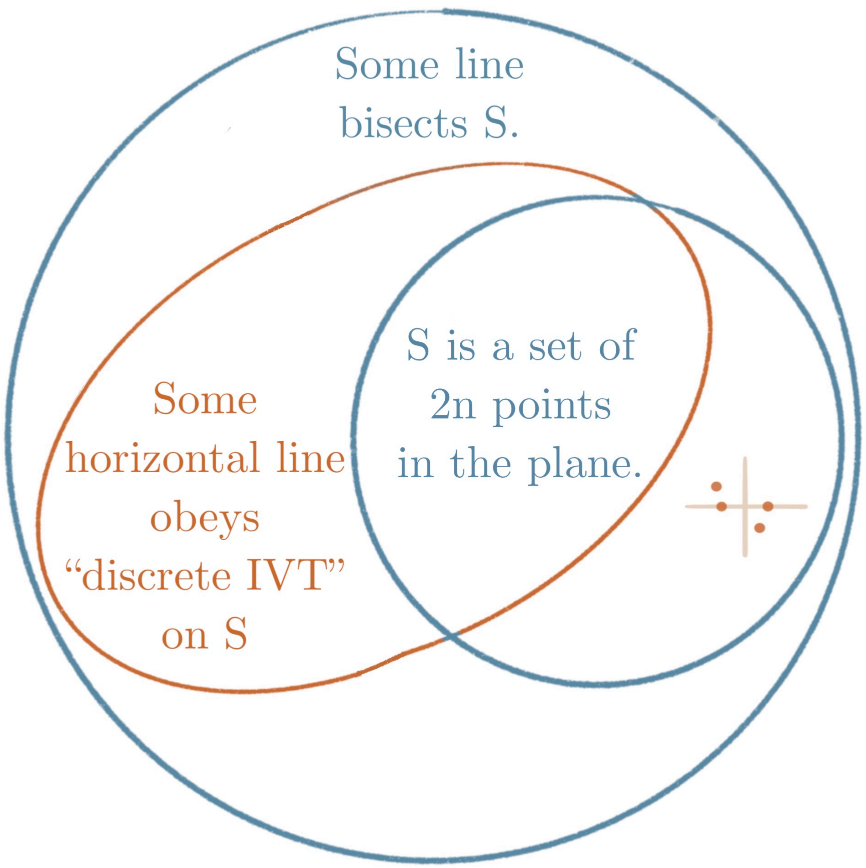
An Example of Conjecture Refinement

We then **disprove** the conjecture: we find a set of points such that a horizontal line does not pass through exactly one point at a time as it is translated.



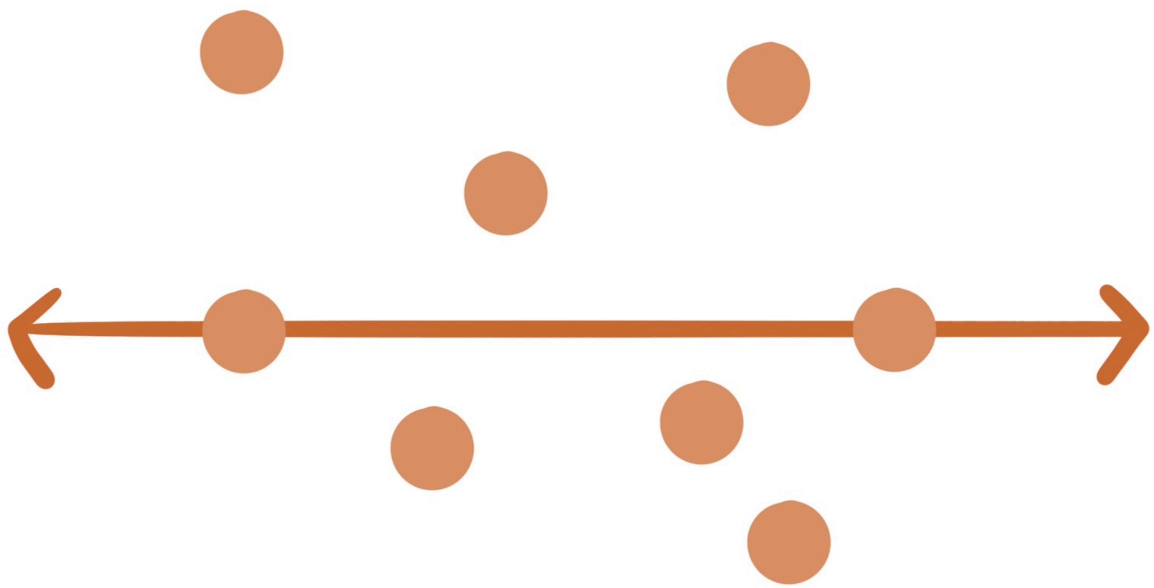
An Example of Conjecture Refinement

We then **disprove** the conjecture: we find a set of points such that a horizontal line does not pass through exactly one point at a time as it is translated.



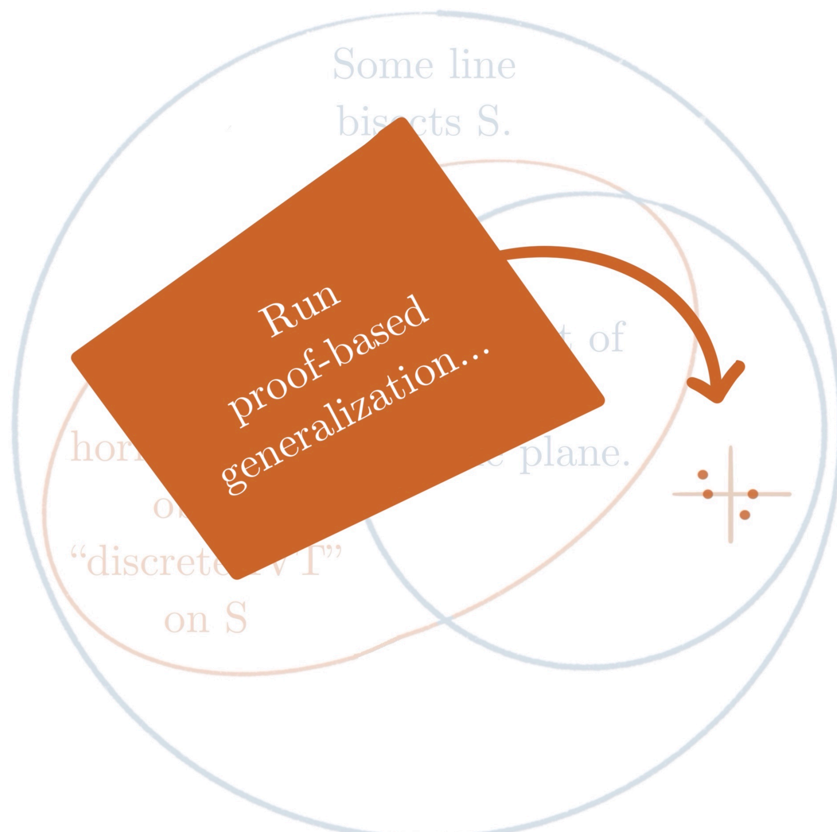
An Example of Conjecture Refinement

We **learn from the disproof** by **generalizing the failure.** If *any* point set contains two points in a horizontal line, discrete IVT doesn't hold (and thus, there might not exist a horizontal line which bisects the set).



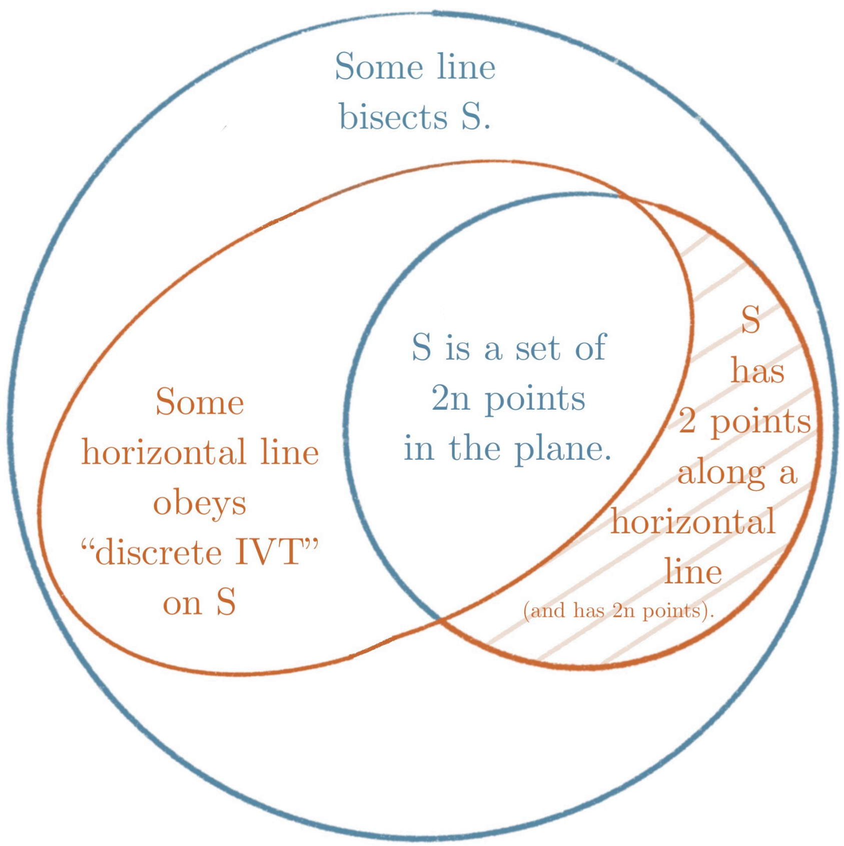
An Example of Conjecture Refinement

We **learn from the disproof** by **generalizing the failure.** If *any* point set contains two points in a horizontal line, discrete IVT doesn't hold (and thus, there might not exist a horizontal line which bisects the set).



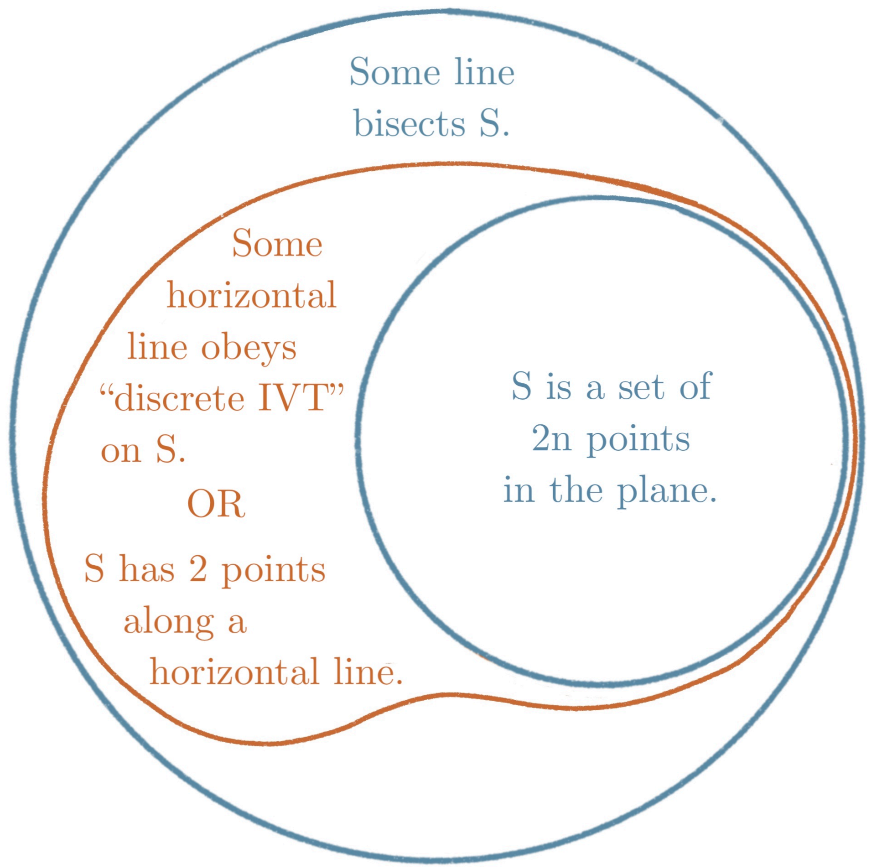
An Example of Conjecture Refinement

We **learn from the disproof** by **generalizing the failure.** If *any* point set contains two points in a horizontal line, discrete IVT doesn't hold (and thus, there might not exist a horizontal line which bisects the set).



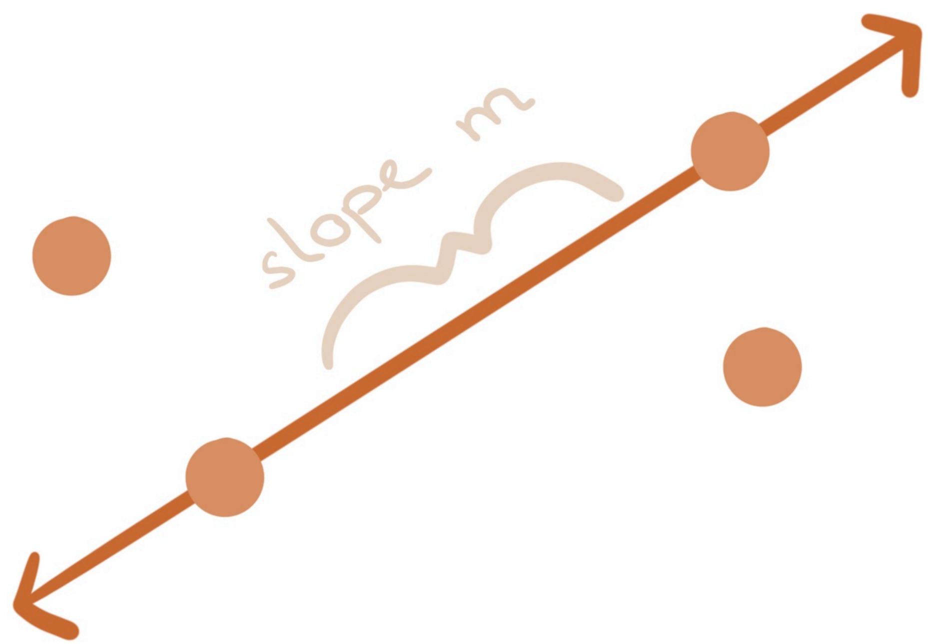
An Example of Conjecture Refinement

We recognize that because we've found the entire reason that the implication is false, we can formulate a better intermediate statement.



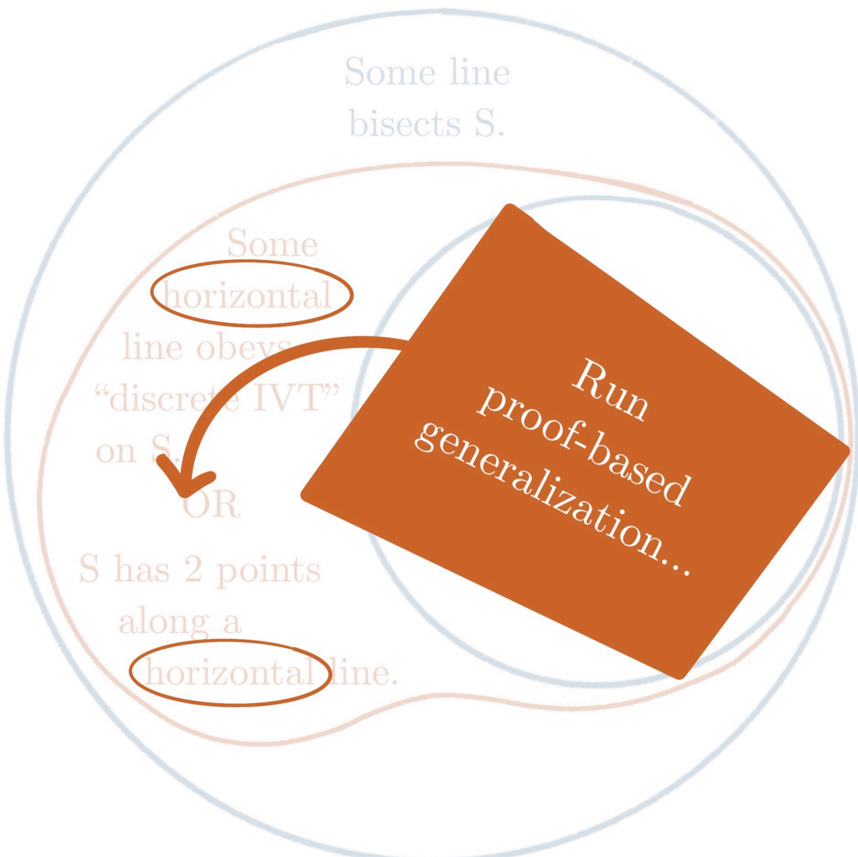
An Example of Conjecture Refinement

Then, we run **proof-based generalization** again — **generalizing "horizontal" to an arbitrary slope**.



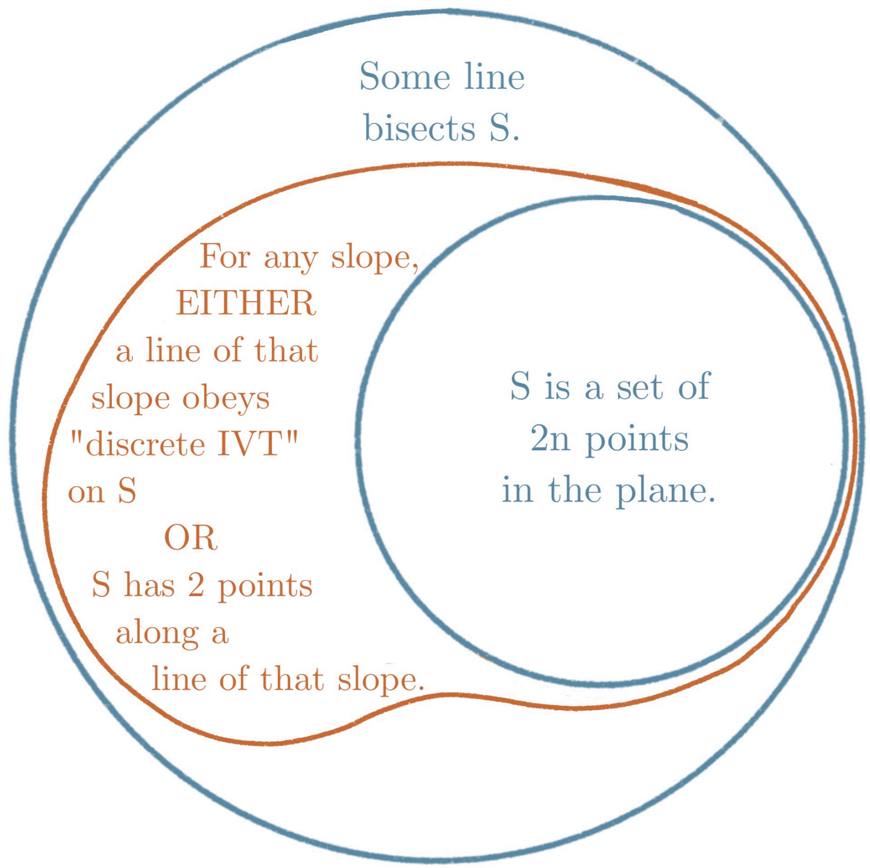
An Example of Conjecture Refinement

We run **proof-based generalization** on the new implication — **generalizing "horizontal" to an arbitrary slope**.



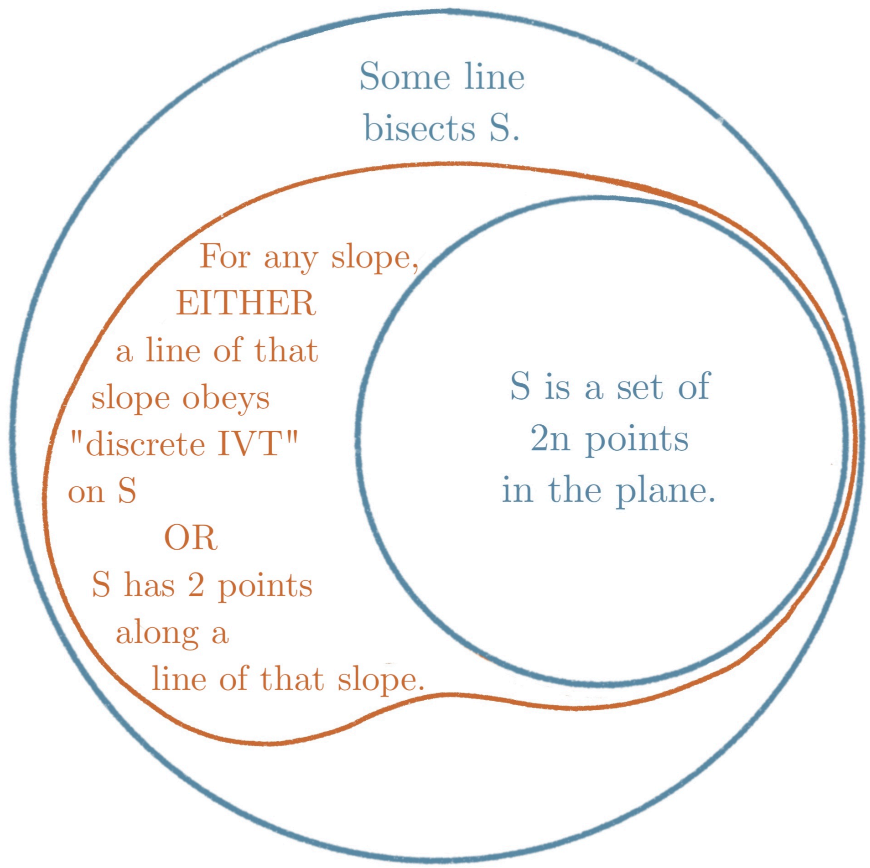
An Example of Conjecture Refinement

We run **proof-based generalization** on the new implication — **generalizing "horizontal" to an arbitrary slope**.



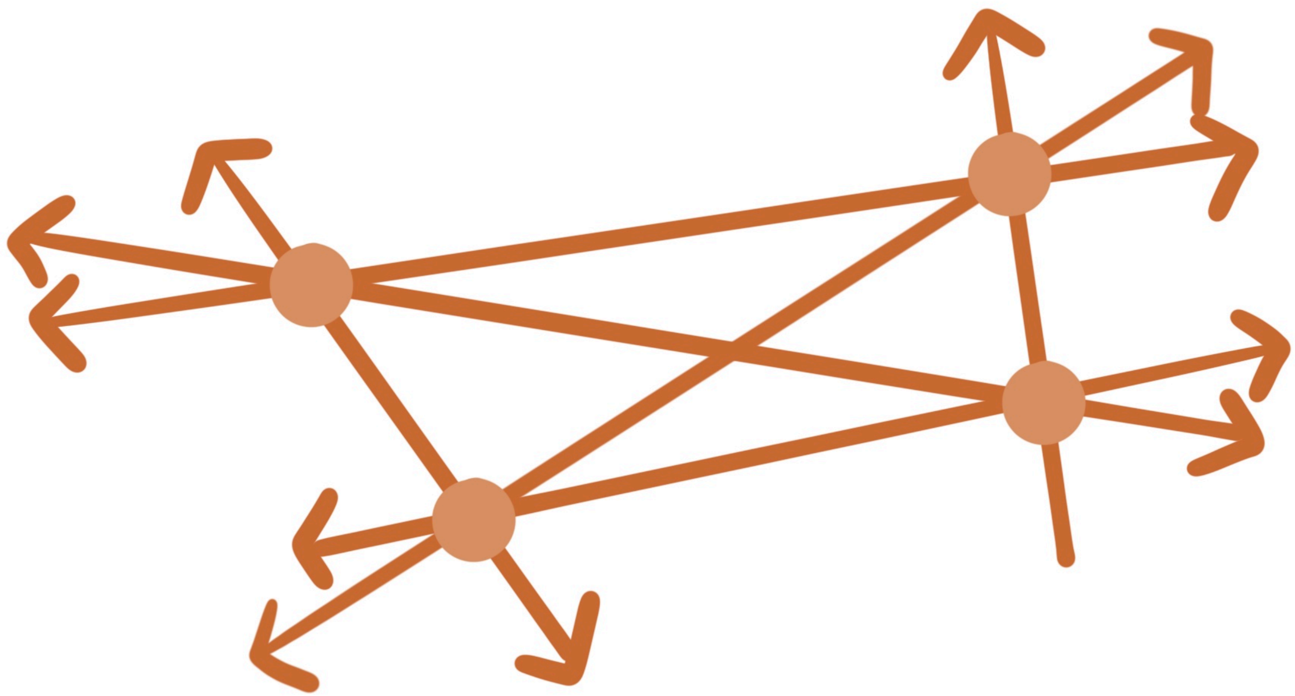
An Example of Conjecture Refinement

Now we're on our way to finishing the proof. We just need to show that the second possibility doesn't occur for all slopes...



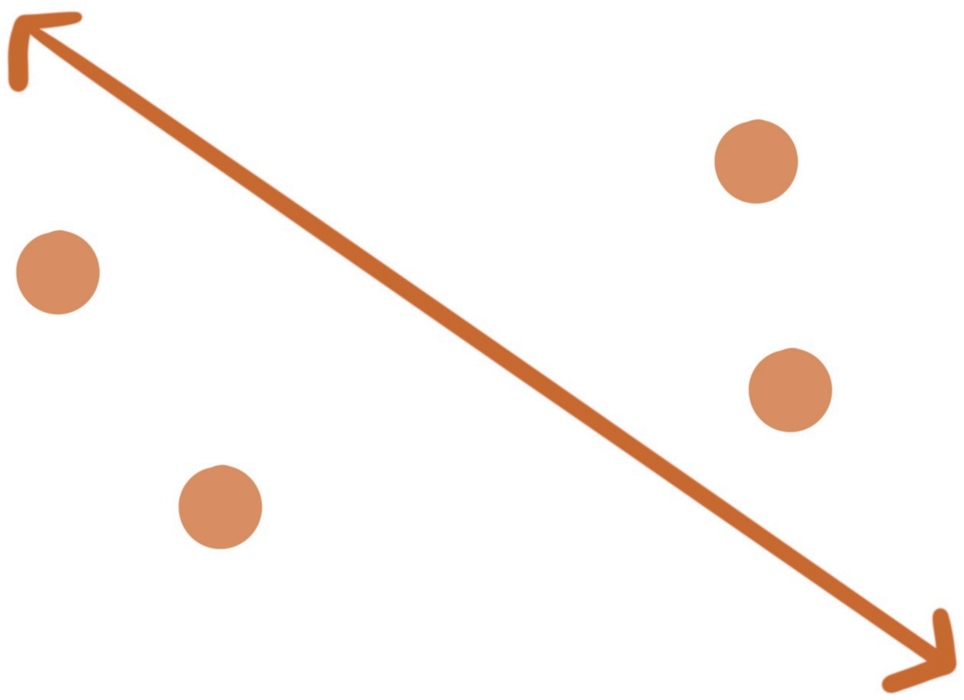
An Example of Conjecture Refinement

Only $\binom{2n}{2}$ "forbidden" slopes (i.e. a line with that slope intersects 2 points) exist...



An Example of Conjecture Refinement

Only $\binom{2n}{2}$ "forbidden" slopes (i.e. a line with that slope intersects 2 points) exist...so any other slope must obey "discrete IVT" on S, and therefore bisect the set.



Note...

We don't have to come up with particularly "clever" initial conjectures!

As long as we can **learn from the failures** of our disproved conjectures, we can often be guided towards more sophisticated, clever conjectures by building on top of more straightforward ones.

An Algorithm to Generalize Proofs

We applied (in our heads) a *proof-based generalization* algorithm (by generalizing “as far as the proof allows”) several times in the lines-bisecting-points example...

This method of proof-based generalization lends itself to mechanization...

An Algorithm to Generalize Proofs

We've implemented a **proof-based generalization algorithm** in Lean. That is, we've developed an algorithm that can take in a mathematical proof, and outputs a more general statement that the "same" proof works for.

This algorithm builds on the work of Olivier Pons ("Generalization in type theory based proof assistants"), who implemented a precursor to this algorithm in Rocq.

An Algorithm to Generalize Proofs

Suppose we prove:

$$\sqrt{2} \text{ is irrational.}$$

```
example := by
  let irrat_sqrt : Irrational (sqrt 2) := by {apply irrat_d
```

▼Tactic state

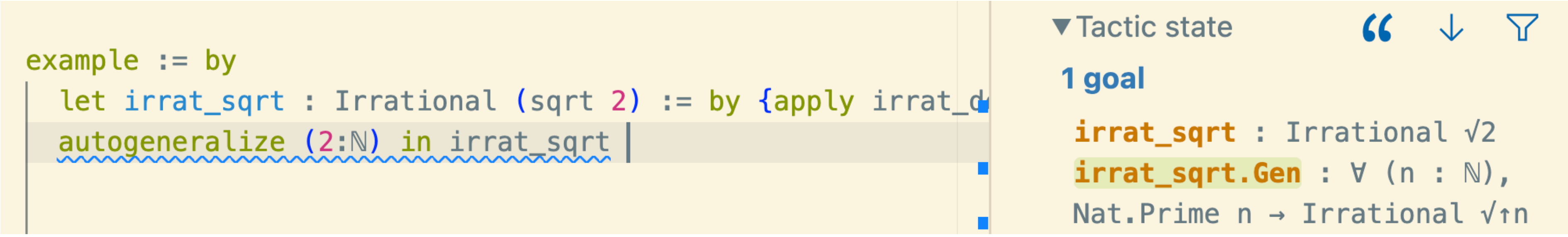
1 goal

```
irrat_sqrt : Irrational √2
```

An Algorithm to Generalize Proofs

Suppose we prove:

$$\sqrt{2} \text{ is irrational.}$$



This algorithm examines the statement and its proof, and by checking which lemmas in the proof are used, **generalizes** to the theorem:

$$\forall \text{ primes } p, \sqrt{p} \text{ is irrational.}$$

An Algorithm to Generalize Proofs

Suppose we prove:

> *The union of two sets of size* **2** *has size at most* 4.

```
example := by
  let union_of_sets (A B : Finset α)
    (hA : A.card = 2) (hB : B.card = 2) : (A ∪ B).card ≤ 4 := by app
```

```
▼Tactic state
1 goal
α β : Type
inst : Fintype α
inst_1 : Fintype β
inst_2 : DecidableEq α
union_of_sets : ∀ (A B : Finset
α), A.card = 2 → B.card = 2 →
(A ∪ B).card ≤ 4
```

An Algorithm to Generalize Proofs

Suppose we prove:

> *The union of two sets of size* 2 *has size at most* 4.

```
example := by
  let union_of_sets (A B : Finset α)
    (hA : A.card = 2) (hB : B.card = 2) : (A ∪ B).card ≤ 4 := by

  autogeneralize (2:ℕ) in union_of_sets
```

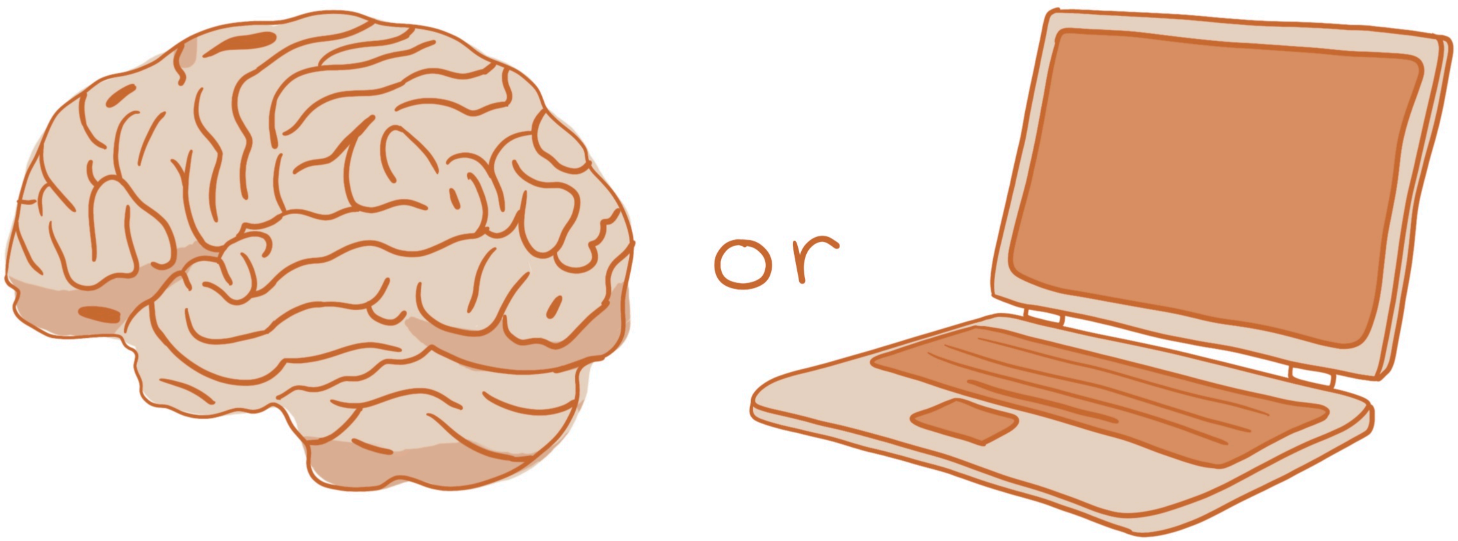
```
▼Tactic state
1 goal
α β : Type
inst : Fintype α
inst_1 : Fintype β
inst_2 : DecidableEq α
union_of_sets : ∀ (A B : Finset α), A.card = 2 → B.card = 2 → (A ∪ B).card ≤ 4
union_of_sets.Gen : ∀ (n m : ℕ) (A B : Finset α), A.card = n → B.card = m → (A ∪ B).card ≤ n + m
```

The algorithm recognizes that the 4 is actually a $2 + 2$, and that the 2s need not be generalized to the same variable (abilities we've added to the algorithm which weren't present in the precursor). So it **generalizes** to the theorem:

> *The union of sets of size* n *and* m *has size at most* $n + m$.

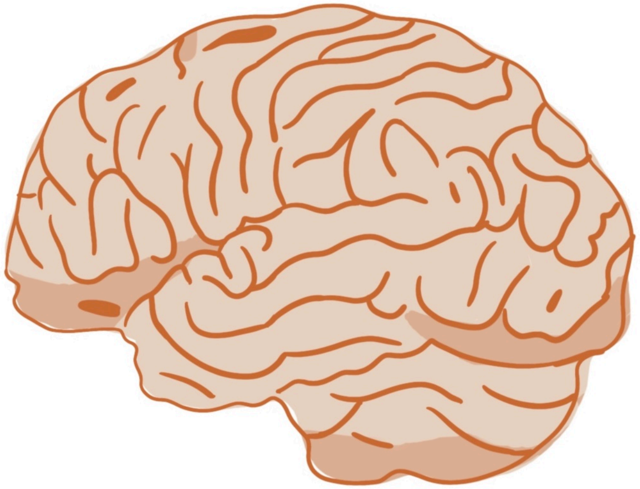
Applications

We want to elucidate the process of mathematical proof finding — both to **aid mathematicians**, and to **aid computers** (which then aid mathematicians).



How Does This Aid Mathematics?

A lot of people find it hard to get started with mathematical research.



The advice to students to just "do a lot of proofs" isn't always helpful. If we can **better understand how research mathematics is done — including how we conjecture and how we generalize**, we can more **effectively teach this skill.**

Thank You

Questions?