Procrastination
A proof engineering technique

Armaël Guéneau
Proving existential quantifications

Goal $\exists x, \ldots$ complicated expression ....

Proof.

(* ??? *)
Guessing the witness

Goal $\exists x, \ldots$ complicated expression $\ldots$

Proof.
exists 0.
Guessing the witness

\[
\textbf{Goal} \exists x, \ldots \ \text{complicated expression} \ldots \\
\textbf{Proof.} \\
\quad \text{exists } 0. \\
\quad \ldots \\
\quad \text{Qed.}
\]
Guessing the witness

Goal $\exists x, \ldots$ complicated expression $\ldots$

Proof.

exists 53367.
Guessing the witness

Goal \( \exists x, \ldots \) complicated expression \( \ldots \).

Proof.

exists 53367.

- Hard to guess
Guessing the witness

Goal $\exists x, \ldots$ complicated expression $\ldots$.

Proof.

exists 53367.

- Hard to guess
- Found using a trial-and-error process?
Guessing the witness

\[ \exists x, \ldots \text{complicated expression} \ldots \]

Proof.

exists 53367.

• Hard to guess
• Found using a trial-and-error process?
• “Magic witness” may be large or duplicate information
Guessing the witness

Goal: $\exists x, \ldots$ complicated expression $\ldots$

Proof:
exists 387.

- Hard to guess
- Found using a trial-and-error process?
- “Magic witness” may be large or duplicate information
Guessing the witness

\[ \text{Goal} \exists x, \ldots \text{complicated expression} \ldots \]

Proof.

\[
\exists x, \ldots \text{complicated expression} \ldots
\]

- Hard to guess
- Found using a trial-and-error process?
- "Magic witness" may be large or duplicate information
- Not maintainable wrt. refactoring or changes in the definitions
**Motivation: asymptotic complexity proofs**

In the context of *A Fistful of Dollars: Formalizing Asymptotic Complexity Claims via Deductive Program Verification*, Guéneau, Charguéraud, Pottier, ESOP’18.

**Goal**

\[ \exists (f: \text{nat} \rightarrow \text{nat}), \]

\[ (\forall n, \]

\[ 1 + \text{if} \zerop n \text{ then } 0 \text{ else } 1 + \max (f(n/2))(f(n - (n/2) - 1))) \]

\[ \leq f(n) \]

\[ \wedge \]

\[ f \in O(\log2). \]
Motivation: asymptotic complexity proofs

In the context of *A Fistful of Dollars: Formalizing Asymptotic Complexity Claims via Deductive Program Verification*, Guéneau, Charguéraud, Pottier, ESOP’18.

**Goal**

\[ \exists (f: \text{nat} \rightarrow \text{nat}), \]

\[ (\forall n, \]

\[ 1 + (\text{if zerop } n \text{ then } 0 \text{ else } 1 + \max (f (n / 2)) (f (n - (n / 2) - 1))) \leq f n) \]

\[ \wedge \]

\[ f \in O(\log 2). \]

**Proof**. \text{exists}(\text{fun } n \Rightarrow \text{if zerop } n \text{ then } 1 \text{ else } 2 \times \log 2 \ n + 3).
The problem: how to delay providing a witness?
The solution: “use evars!”
The solution: “use evars!”

but maybe not naively...
Basic evar workflow

Goal: \( \exists x, P \ x. \)

Proof:

\( \text{eexists.} \)

\( (* \vdash P \ ?x *) \)
Basic evar workflow

Goal $\exists x, P \; x$.
Proof.
  eexists.
  ($^* \vdash P \; ?x^*$)
  ...
  ($^* \vdash ?x = 5^*$)
Basic evar workflow

Goal $\exists x, P x$.

Proof.

$\text{eexists}$. 

(* ⊢ $P \ ?x$ *)

...

(* ⊢ $?x = 5$ *)

reflexivity.

Qed.

Useful if *equalities* are discovered about the evar.
Goal $\exists x, P x$.

Proof.

eeexists.

(* $\vdash P \ ?x$ *)
Basic evar workflow

Goal $\exists x, P x$.

Proof.

  eexists.

  (* $\vdash P ?x *$)

  ...

  (* $\vdash ?x \geq 5 *$)
Basic evar workflow

Goal $\exists x, P \, x$.

Proof.

\[ \text{eexists.} \]
\[ (* \vdash P \, ?x \, *) \]
\[ ... \]
\[ (* \vdash \, ?x \geq 5 \, *) \]
\[ (* \, x \,:=\, 5 \, ? \, *) \]
Basic evar workflow

Goal $\exists x, P x$.

Proof.

$\texttt{eexists}.$

$(* \vdash P \ ?x *)$

...

$(* \vdash \ ?x \geq 5 *)$

$(* x := 5 ? *)$

...

$(* \vdash \ ?x \geq 8 *)$
Cyril Cohen’s *bigenough* library enables new tricks with evars:

**Goal** \( \exists x, P x \).

**Proof.**

\[ \text{exists}_\text{big} \ x \ \text{nat.} \ (* \ x := \ldots \ \vdash P x *) \]
Cyril Cohen’s *bigenough* library enables new tricks with evars:

Goal $\exists x, P x$.  
Proof.

exists_big $x$ nat. (* $x := \ldots \vdash P x$ *)

...  
(* $x := \ldots \vdash x \geq 5$ *)
Cyril Cohen’s *bigenough* library enables new tricks with evars:

**Goal** $\exists x, P \ x$.

**Proof.**

exists _big_ $x$ nat. (* $x := \ldots \vdash P \ x$ *)

...  

(* $x := \ldots \vdash x \geq 5$ *)

big.
Cyril Cohen’s bigenough library enables new tricks with evars:

Goal $\exists x, P x$.

Proof.

exists_big $x$ nat. (* $x := \ldots \vdash P x$ *)

... 

(* $x := \ldots \vdash x \geq 5$ *)

big.

...

(* $x := \ldots \vdash x \geq 8$ *)

big.

Useful if lower bounds are discovered about $x$. Builds an iterated maximum.
Cyril Cohen’s `bigenough` library enables new tricks with evars:

Goal $\exists x, P x$.

Proof.

exists_big x nat. (* $x := \ldots \vdash P x$ *)

... (*) $x := \ldots \vdash x \geq 5$ *)

big.

... (*) $x := \ldots \vdash x \geq 8$ *)

big.

... close.

Qed.

Useful if lower bounds are discovered about $x$. Builds an iterated maximum.
“Big-enough”

- What if arbitrary side-conditions can be encountered? E.g. "\(x \geq 5\)" and "\(x \leq 8\)"

- We might also be inferring not an integer but a function \(f\) satisfying "\(\forall n, f(n) \geq 1 + f(n-1)\)"

Hard to guess the shape of the solution beforehand (e.g. as an iterated maximum).
Procrastination, as a small Coq library

• Allows deferring arbitrary side-conditions about zero or several variables;
• Does not try to solve these, simply gathers them;
• **Enforces** a separation of the proof in two phases: (1) side conditions are collected; (2) side conditions are solved and variables are instantiated.
Demo
Procrastination: key properties

- When collecting side-conditions about variables, these variables are “rigid”
- Side-conditions are collected and gathered under a common context, in a single subgoal
- `begin defer .. end defer` blocks can be arbitrarily nested
Lemma defer_1: \( \forall A \, (P: \text{Prop}) \, (Q: A \rightarrow \text{Prop}) \),

\[ (\forall a, Q \, a \rightarrow P) \rightarrow (\exists a, Q \, a) \rightarrow P. \]

\[
\begin{align*}
\text{begin} & \, \text{defer assuming } a := \\
& \quad \text{eapply defer_1;} [\text{intros } a?].
\end{align*}
\]
Start procrastinating now!

```
opam install coq-procrastination

Require Import Procrastination.Procrastination.

https://github.com/Armael/coq-procrastination
```