Wlog – Without Loss of Generality*

Dominique Unruh
RWTH Aachen, University of Tartu

February 6, 2026

Abstract

We introduce a new command `wlog` in Isabelle/HOL that allows us to (soundly) assume facts without loss of generality inside a proof.

Contents

1 Introduction **1**

2 *Wlog* – Setting up the command **3**

3 *Wlog-Examples* – Examples how to use `wlog` **4**

1 Introduction

We introduce a command `wlog` for assuming facts without loss of generality inside a proof in Isabelle/HOL. The `wlog` command makes sure this is sound by requiring us to prove that the assumption is indeed made without loss of generality.

A simple example is the following:

```
lemma card_nth_roots_strengthened:
  assumes "c ≠ 0"
  shows   "card {z::complex. z ^ n = c} = n"
proof -
  wlog n_pos: "n > 0"
    using negation by (simp add: infinite_UNIV_char_0)
  have "card {z. z ^ n = c} = card {z::complex. z ^ n = 1}"
    by (rule sym, rule bij_betw_same_card, rule bij_betw_nth_root_unity) fact+
```

*Supported by the ERC consolidator grant CerQuS (819317), the PRG team grant Secure Quantum Technology (PRG946) from the Estonian Research Council, and the Estonian Cluster of Excellence "Foundations of the Universe" (TK202).

```
  also have "... = n" by (rule card_roots_unity_eq) fact+
  finally show ?thesis .
qed
```

This proof is exactly like the proof of `Complex.card_nth_roots` in the Isabelle/HOL library, except that the latter uses the additional assumption `n > 0` in the theorem statement. We omit this assumption and instead state that it can be assumed without loss of generality. (`wlog n_pos: "n > 0"`) The next line then shows that this can be assumed without loss of generality.[1]

Of course, we could have shown this theorem also, e.g., by doing a case distinction on whether $n = 0$. But this would additionally clutter the proof; the case $n = 0$ is almost trivial, yet in the proof it will be a separate case on the same level as the main proof. So doing a `wlog` improves readability here by allowing us to focus on the important parts of the proof and reducing boilerplate.

In other cases, a `wlog` argument cannot easily be done as a case distinction. E.g., if we say that we can assume w.l.o.g. that $a \geq b$ because the case $a < b$ can be easily reduced to the $a \geq b$ case. (This is common in symmetric situations.) We give an example of this in the proof of lemma `schur_ineq` below.

The full syntax of the `wlog` command is roughly as follows:

```
wlog wlogassmname: wlogassm1 wlogassm2
     goal G  generalizing x y z  keeping fact1 fact2
  [... your proof ...]
```

(The defaults being: The goal is `?thesis`. And empty lists of variables and facts for `generalizing` and `keeping`.)

This means that we assume w.l.o.g. that the facts `wlogassm1` and `wlogassm2` hold when proving the goal `G`. We say that the assumptions `fact1` and `fact2` (made prior to the `wlog` command) should still be available afterwards. (If we include less assumptions here, the justification for the `wlog` command becomes easier.) And we wish to generalize the variables `x, y, z`; that is, inside the justification of the `wlog`, we want to be allowed to use the theorem that we are proving for other values of `x, y, z` (needed, e.g., in symmetry arguments). And `[... your proof ...]` is a proof of the fact that we can make the w.l.o.g.-assumption, either as an apply-script or as an Isar subproof.

The `wlog` command is realized by translation to existing Isar commands. The above translates roughly to:

```
  presume hypothesis:
    ⋀x y z. wlogassm ⟹ fact1 ⟹ fact2 ⟹ G
```

[1] The argument is basically: If $\neg(n > 0)$, then $n = 0$ (since n is a natural number). Then $\{z.\ z^n = c\}$ is infinite, and for infinite sets, the cardinality `card` is defined to be 0 in Isabelle/HOL. Thus that cardinality is 0. This reasoning is done almost automatically by Isabelle.

```
  have G if negation: ň (wlogassm1 ∧ wlogassm2)
    [... your proof ...]
  then show G
    [... autogenerated proof ...]
next
  fix x y z
  assume fact1: fact1 and fact2: fact2
  assume wlogassmname: wlogassm1 wlogassm2
```

(There are more steps and additional convenience definitions, but this is the main part.)
More examples of how to use `wlog` are given in the theory `Wlog_Examples` below.

2 *Wlog* – Setting up the command

theory *Wlog*
imports *Main*
keywords *wlog* :: *prf-goal* % *proof*
  and *generalizing* **and** *keeping* **and** *goal*
begin

ML-file *wlog.ML*

For symmetric predicates involving 3–5 variables on a linearly ordered type, the following lemmas are very useful for wlog-proofs.

For two variables, we already have *linorder-wlog*.

lemma *linorder-wlog-3*:
  fixes $x\ y\ z :: \langle 'a :: linorder \rangle$
  assumes $\langle \bigwedge x\ y\ z.\ P\ x\ y\ z \Longrightarrow P\ y\ x\ z \wedge P\ x\ z\ y \rangle$
  assumes $\langle \bigwedge x\ y\ z.\ x \leq y \wedge y \leq z \Longrightarrow P\ x\ y\ z \rangle$
  shows $\langle P\ x\ y\ z \rangle$
  using *assms*
  by (*metis linorder-le-cases*)

lemma *linorder-wlog-4*:
  fixes $x\ y\ z\ w :: \langle 'a :: linorder \rangle$
  assumes $\langle \bigwedge x\ y\ z\ w.\ P\ x\ y\ z\ w \Longrightarrow P\ y\ x\ z\ w \wedge P\ x\ z\ y\ w \wedge P\ x\ y\ w\ z \rangle$
  assumes $\langle \bigwedge x\ y\ z\ w.\ x \leq y \wedge y \leq z \wedge z \leq w \Longrightarrow P\ x\ y\ z\ w \rangle$
  shows $\langle P\ x\ y\ z\ w \rangle$
  using *assms*
  by (*metis linorder-le-cases*)

lemma *linorder-wlog-5*:
  fixes $x\ y\ z\ w\ v :: \langle 'a :: linorder \rangle$
  assumes $\langle \bigwedge x\ y\ z\ w\ v.\ P\ x\ y\ z\ w\ v \Longrightarrow P\ y\ x\ z\ w\ v \wedge P\ x\ z\ y\ w\ v \wedge P\ x\ y\ w\ z\ v \wedge P\ x\ y\ z\ v\ w \rangle$
  assumes $\langle \bigwedge x\ y\ z\ w\ v.\ x \leq y \wedge y \leq z \wedge z \leq w \wedge w \leq v \Longrightarrow P\ x\ y\ z\ w\ v \rangle$
  shows $\langle P\ x\ y\ z\ w\ v \rangle$
  using *assms*

by (*smt* (*verit*) *linorder-le-cases*)

end

3 *Wlog-Examples* – Examples how to use `wlog`

theory *Wlog-Examples*
imports *Wlog Complex-Main*
begin

The theorem *Complex.card-nth-roots* has the additional assumption $0 < n$. We use exactly the same proof except for stating that w.l.o.g., $0 < n$.

lemma *card-nth-roots-strengthened*:
assumes $c \neq 0$
shows *card* $\{z::complex.\ z \hat{} n = c\} = n$
proof −
wlog *n-pos*: $n > 0$
using *negation* **by** (*simp add*: *infinite-UNIV-char-0*)
have *card* $\{z.\ z \hat{} n = c\}$ = *card* $\{z::complex.\ z \hat{} n = 1\}$
by (*rule sym*, *rule bij-betw-same-card*, *rule bij-betw-nth-root-unity*) *fact+*
also have ... = n **by** (*rule card-roots-unity-eq*) *fact+*
finally show *?thesis* **.**
qed

This example very roughly follows Harrison [1]:

lemma *schur-ineq*:
fixes $a\ b\ c$ :: ‹'a :: *linordered-idom*› **and** k :: *nat*
assumes *a0*: ‹$a \geq 0$› **and** *b0*: ‹$b \geq 0$› **and** *c0*: ‹$c \geq 0$›
shows ‹$a \hat{} k * (a - b) * (a - c) + b \hat{} k * (b - a) * (b - c) + c \hat{} k * (c - a) * (c - b) \geq 0$›
(**is** ‹*?lhs* ≥ 0›)
proof −
wlog *ordered*[*simp*]: ‹$a \leq b$› ‹$b \leq c$› **generalizing** $a\ b\ c$ **keeping** *a0 b0 c0*
apply (*rule rev-mp*[*OF c0*]; *rule rev-mp*[*OF b0*]; *rule rev-mp*[*OF a0*])
apply (*rule linorder-wlog-3*[*of* - $a\ b\ c$])
apply (*simp add*: *algebra-simps*)
by (*simp add*: *hypothesis*)

from *ordered* **have** [*simp*]: ‹$a \leq c$›
by *linarith*

have ‹*?lhs* = $(c - b) * (c \hat{} k * (c - a) - b \hat{} k * (b - a)) + a \hat{} k * (c - a) * (b - a)$›
by (*simp add*: *algebra-simps*)
also have ‹... ≥ 0›
by (*auto intro*!: *add-nonneg-nonneg mult-nonneg-nonneg mult-mono power-mono zero-le-power simp*: *a0 b0 c0*)
finally show ‹*?lhs* ≥ 0›
by −
qed

The following illustrates how facts already proven before a **wlog** can be still be used after the wlog. The example does not do anything useful.

lemma ‹$A \Longrightarrow B \Longrightarrow A \wedge B$›
proof −
have *test1*: ‹*1=1*› **by** *simp*
assume *a*: ‹*A*›
then have *test2*: ‹$A \vee 1 \neq 2$› **by** *simp*
— Isabelle marks this as being potentially based on assumption *a*. (Note: this is not done by actual dependency tracking. Anything that is proven after the **assume** command can depend on the assumption.)
assume *b*: ‹*B*›
with *a* **have** *test3*: ‹$A \wedge B$› **by** *simp*
— Isabelle marks this as being potentially based on assumption *a*, *b*
wlog *true*: ‹*True*› **generalizing** *A B* **keeping** *b*
— A pointless wlog: we can wlog assume *True*. Notice: we only keep the assumption *b* around.
using *negation* **by** *blast*

The already proven theorems cannot be accessed directly anymore (wlog starts a new proof block). Recovered versions are available, however:

thm *wlog-keep.test1*
— The fact is fully recovered since it did not depend on any assumptions.
thm *wlog-keep.test2*
— This fact depended on assumption *a* which we did not keep. So the original fact might not hold anymore. Therefore, *wlog-keep.test2* becomes $A \Longrightarrow A \vee 1 \neq (2::'a)$. (Note the added *A* premise.)
thm *wlog-keep.test3*
— This fact depended on assumptions *a* and *b*. But we kept *b*. Therefore, *wlog-keep.test2* becomes $A \Longrightarrow A \wedge B$. (Note that only *A* is added as a premise.)
oops
— Aborting the proof here because we cannot prove $A \wedge B$ anymore since we dropped assumption *a* for demonstration purposes.

end

References

[1] J. Harrison. Without loss of generality. In S. Berghofer, T. Nipkow, C. Urban, and M. Wenzel, editors, *Theorem Proving in Higher Order Logics*, pages 43–59, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg. Eprint available at https://www.cl.cam.ac.uk/~jrh13/papers/wlog.pdf.