Well-orderings and Well-foundedness

R.D. Arthan and R.B. Jones

Date: 2014/08/17 16:07:53

Abstract

This document consists of two parts. The first is a theory of well-orderings prepared by Rob Arthan for possible inclusion in the ProofPower theory of ordered sets. The second is material on well-foundedness, mainly consisting in the proof of the recursion theorem which is needed for consistency proofs of definitions by transfinite recursion respecting (if that's the right term) some well founded relationship.

Created 2004/10/03

Last Change Date: 2014/08/17 16:07:53

http://www.rbjones.com/rbjpub/pp/doc/t009.pdf

Id: t009.doc,v 1.20 2014/08/17 16:07:53 rbj Exp

Contents

References

[1] Roger Bishop Jones. Introduction to Work in Progress. *RBJones.com*, 2010. `http://www.rbjones.com/rbjpub/pp/doc/t000.pdf`.

0.1 RBJ's Preface

For context and motivation see [1].

This document has been produced by Roger Jones by hacking a document written by Rob Arthan called *ordered_sets*.

This preface contains my notes on what I have done to it:

1. Change the name of the document.
2. Make sure theory listing uses aliases.
3. Change so that the theory of orders is not incorporated by duplication.
4. Change to suppress the listing of proofs and to include instead displays of key theorems and lemmas in the narrative.
5. Messed about with the section structure.
6. Put material on relations over a whole type and the independence proofs into a separate theory.
7. Added the material on transitive closure, the well-founded part of a relationship, and the recursion theorem.

1 INTRODUCTION

This note contains various definitions and theorems, many of which are ultimately destined for the ProofPower theory *orders*, which, it is intended, will supply a generally useful collection of definitions and results relating to ordered sets of various kinds (including ordered sets accompanied by some finer structure, e.g., an ordered set given as the transitive closure of a, not necessarily transitive, well-founded, relation, representing some pattern for defining functions by recursion).

The significant mathematical facts that this note adds to the theory of ordered sets are: Zorn's lemma for relations viewed as propositional functions, the well-ordering principle and some useful equivalences to do with well-foundedness (e.g., we show that a set is well-founded iff. it has no infinite descending chains). These results and their proofs serve to validate the definitions.

In part, the purpose of this note is to investigate relationships between various primitive notions that are commonly used in talking about ordered sets. In particular, it is common in the literature for concepts such as well-foundedness or antisymmetry to be defined so that they entail other properties such as irreflexivity. The goal here is to fill out the list of primitive notions offered in the theory *orders* to give an adequate vocabulary to state the various definitions commonly given in the literature in terms of primitive notions that are independent. The result is a list of six primitive notions: reflexivity, irreflexivity, antisymmetry, transitivity, trichotomy and a notion we call the "weak minimal condition" which is an analogue of well-foundedness broadened so as not to entail reflexivity or antisymmetry.

Where useful notions, like the customary definition of well-foundedness, entail more than one primitive notion, in this note, the notion is expressed in terms of the chosen primitive ones and the customary definition is proved as a theorem.

Section 6 contains proofs that the chosen primitive notions are indeed independent, by proving that for each pair P and Q of primitive notions their are examples where both hold, where P holds and not Q and where Q holds and not P. This material is not intended for inclusion in the theory *orders*, but rather to serve as a check on the chosen formalisation (as, in a different way, do the proofs of Zorn's lemma and the well-ordering principle). It also provides an exercise in semi-automated bulk theorem-proving — the challenge being to formulate and prove the desired results without having to type in all 45 cases and all 45 witnesses.

For the convenience of anyone wanting to run this script on ProofPower, this note is a ProofPower literate script including the code for all the proofs (most of which though present in the source is supressed in the printed document).

The appendices include listings of the theory set up by the code in this document. The theory (optionally) includes the existing theorems from the theory *orders*. These comprise a body of results about down-sets which amount to the proof of the Dedekind-MacNeille completion theorem and a collection of facts about orders induced on a set equipped with a function mapping it to an ordered set. These are all there to support the original aim of constructing the type of real numbers.

These existing theorems are followed by a couple of results that make our definition of the minimum condition and of well-foundedness easier to work with. Then come two lemmas leading up to Zorn's lemma (for which all the hard work has been done elsewhere). This is then used (together with some lemmas about subsets of ordered sets and extensions of well-ordered sets) to prove the well-ordering principle. This material is followed by a selection of facts about the minimum condition and well-foundedness to do with descending sequences and induction. This is then specialised to the case where the field of the relation in question is expected to be the universe of a type. Analogues of all the properties are introduced for this case and the main results are specialised to it. Finally come results that build up to the statement that our six primitive notions are logically independent as mentioned above.

This document requires version 2.7.1 of ProofPower — earlier versions of the theory *orders* defined constants *PartialOrder* and *LinearOrder* (now called *StrictPartialOrder* and *StrictLinearOrder*) which conflict with constants defined here; later versions are likely to include the definitions and many of the theorems already.

2 SOME PROPERTIES OF RELATIONS

The definitions below follow the spirit of the existing ones by not requiring reflexivity or irreflexivity except where necessary. The intention is to make the primitive notions logically independent of one another.

2.1 Definitions of Primitive Properties

The primitive notions that are there already are: irreflexivity, antisymmetry, transitivity and trichotomy together with other notions such as density that are more specific to the Dedekind-MacNeille completion construction and its use in constructing the real numbers. To these we add the notions of: reflexivity, and the "weak minimum condition" (which says that every non-empty set contains a minimal element in a certain sense). In terms of these primitives we define derived notions such as well-foundedness.

SML

```
open_theory "orders";
force_new_theory "ordered_sets";
new_parent "set_thms";
new_parent "rbjmisc";

val existing_defs = map get_spec (get_consts "orders");

set_merge_pcs["basic_hol", "'sets_alg", "'savedthm_cs_∃_proof"];
```

Now we give definitions for the primitives not already defined.

This one, we now abstain from providing since it is in theory *equiv_rel* which is now a parent.

```
ⓈHOLCONST
│ ⌊Refl⌋ : ('a SET × ('a → 'a → BOOL)) → BOOL
├──────
│ ∀ (X, $<<)•
│       Refl(X, $<<)
│ ⇔    ∀x•x ∈ X ⇒ x << x
■
```

We follow Cohn's Universal Algebra in using the term "minimum condition" for the condition that gives well-foundedness of the ordered set formulated so that the ordering relation need not be irreflexive. This is intended to accommodate the varying conventions that are typically used in the literature for well-ordered sets on the one hand and recursion with respect to a well-founded relation on the other. However, this condition entails that the relation is antisymmetric, so we take as primitive the following weaker version.

HOL Constant

```
WeakMinCond : ('a SET × ('a → 'a → BOOL)) → BOOL
──────────────────────
∀ (X, $<<)•
        WeakMinCond(X, $<<)
⇔      ∀A• A ⊆ X ∧ ¬A = {} ⇒
        ∃x• x ∈ A ∧ ∀y• y ∈ A ∧ y << x ∧ ¬ y = x ⇒ x << y
```

We now have our full set of primitives.

2.2 Definitions of Derived Properties

The following non-primitive properties are of interest.

HOL Constant

$$\boldsymbol{PartialOrder} : ('a\ SET \times ('a \to 'a \to BOOL)) \to BOOL$$

$$\forall\ (X,\ \$<<)\bullet\quad PartialOrder(X,\ \$<<) \Leftrightarrow Antisym(X,\ \$<<) \land Trans(X,\ \$<<)$$

HOL Constant

$$\boldsymbol{LinearOrder} : ('a\ SET \times ('a \to 'a \to BOOL)) \to BOOL$$

$$\forall\ (X,\ \$<<)\bullet\ LinearOrder(X,\ \$<<) \Leftrightarrow PartialOrder(X,\ \$<<) \land Trich(X,\ \$<<)$$

HOL Constant

$$\boldsymbol{MinCond} : ('a\ SET \times ('a \to 'a \to BOOL)) \to BOOL$$

$$\forall\ (X,\ \$<<)\bullet\quad MinCond(X,\ \$<<) \Leftrightarrow Antisym(X,\ \$<<) \land WeakMinCond(X,\ \$<<)$$

HOL Constant

$$\boldsymbol{WellOrdering} : ('a\ SET \times ('a \to 'a \to BOOL)) \to BOOL$$

$$\forall\ (X,\ \$<<)\bullet\quad WellOrdering(X,\ \$<<) \Leftrightarrow LinearOrder(X,\ \$<<) \land WeakMinCond(X,\ \$<<)$$

The general policy heretofore has been to express properties of relations in a manner indifferent to whether the relation is reflexive or not. The concept of well-foundedness is important for recursive definitions and inductive reasoning, but for these purposes a strict inequality is needed.

A well-founded relation is therefore an irreflexive relation which satisfies *MinCond*:

HOL Constant

$$\boldsymbol{WellFounded} : ('a\ SET \times ('a \to 'a \to BOOL)) \to BOOL$$

$$\forall\ (X,\ \$<<)\bullet\quad WellFounded(X,\ \$<<) \Leftrightarrow Irrefl(X,\ \$<<) \land MinCond(X,\ \$<<)$$

The inductive applications of well founded relations depend upon the equivalence of this definition with an induction principle. Such an induction principle is needed for reasoning about well-orderings so further results about well-orderings will be deferred until the induction principle is in place.

2.3 Using the Definitions

In this section we do a little preening on the definitions, including a recasting of the definitions for the minimum condition and well-foundedness.

A relation has the minimum condition iff. every non-empty subset has a minimal element (in the more usual sense). We will almost invariably use this formulation in the sequel. Well-foundedness can be rendered similarly, in this case expression of the minimality condition is simplified by the fact that well-founded relations are by definition irreflexive.

```
min_cond_def_thm = ⊢ ∀X $<<• MinCond(X, $<<) ⇔
        (∀ A• A ⊆ X ∧ ¬ A = {}
                ⇒ (∃ x• x ∈ A ∧ (∀ y• y ∈ A ∧ ¬ y = x ⇒ ¬ y << x)))

well_founded_thm = ⊢ ∀X $<<• WellFounded(X, $<<) ⇔
        (∀ A• A ⊆ X ∧ ¬ A = {}
                ⇒ (∃ x• x ∈ A ∧ (∀ y• y ∈ A ⇒ ¬ y << x)))
```

The following is intended for use in conjunction with *min_cond_thm*.

```
well_ordering_def_thm = ⊢ ∀X $<<• WellOrdering(X, $<<)
        ⇔ LinearOrder(X, $<<) ∧ MinCond(X, $<<)
```

2.4 Order Morphisms

An order morphism is an order preserving map between partial orders. The existence of such morphisms determines a partial pre-order over partial orders (the equivalence classes being order-types), and a pre-well-order over well-orderings.

SML

```
declare_infix(210, "<<<");
declare_infix(210, "≤ot");
```

HOL Constant

$$
\begin{array}{l}
\boldsymbol{OrderMorphism} : ('a\ SET \times ('a \to 'a \to BOOL)) \\
\qquad \to ('b\ SET \times ('b \to 'b \to BOOL)) \\
\qquad \to ('a \to 'b) \\
\qquad \to BOOL \\
\hline
\forall\ m\ (X,\ \$<<)\ (Y,\ \$<<<)\bullet \\
\quad OrderMorphism\ (X,\ \$<<)\ (Y,\ \$<<<)\ m \\
\Leftrightarrow \quad \forall x1\ x2\bullet\ x1 \in X \land x2 \in X \Rightarrow m\ x1 \in Y \land m\ x2 \in Y \\
\qquad \land (x1 << x2 \Leftrightarrow m\ x1 <<< m\ x2)
\end{array}
$$

When considering the collection of orderings of a single domain the followng notion of morphism might be used.

HOL Constant

$$
\begin{array}{l}
\$\boldsymbol{OrdMorph} : 'a\ SET \\
\qquad \to ('a \to 'a \to BOOL) \\
\qquad \to ('a \to 'a \to BOOL) \\
\qquad \to BOOL \\
\hline
\forall\ X\ \$<<\ \$<<<\bullet \\
\quad OrdMorph\ X\ \$<<\ \$<<< \\
= \quad \exists m\bullet\ OrderMorphism\ (X,\ \$<<)\ (X,\ \$<<<)\ m
\end{array}
$$

The existence of order morphisms induces the following relation over order types.

HOL Constant

$$
\begin{array}{l}
\$\leqslant_{\boldsymbol{ot}} : ('a\ SET \times ('a \to 'a \to BOOL)) \\
\qquad \to ('b\ SET \times ('b \to 'b \to BOOL)) \\
\qquad \to BOOL \\
\hline
\forall\ (X,\ \$<<)\ (Y,\ \$<<<)\bullet \\
\quad (X,\ \$<<) \leqslant_{ot} (Y,\ \$<<<) \\
\Leftrightarrow \quad \exists m\bullet\ OrderMorphism\ (X,\ \$<<)\ (Y,\ \$<<<)\ m
\end{array}
$$

My main reason for introducing these is that I want to be able to use least well-orderings. Ideally this would be supported by a theorem to the effect that the above is a well-founded ordering of order types, but I can't just now see how to prove that, and may instead settle for showing that the well-orderings of a set are well-founded.

3 WELL-ORDERING

3.1 Zorn's Lemma for Properties of Relations

We already have Zorn's lemma in the special case where the ordered set is some subset of the lattice of subsets of a set (and the ordering relation is inclusion). We just need to "interface" with that theorem by looking at the chains contained in an arbitrary relation.

$$\begin{array}{l} zorn\_thm2 = \vdash \forall X\ \$<<\bullet \\ \qquad Trans(X, \$<<) \\ \quad \land \quad Antisym(X, \$<<) \\ \quad \land \quad (\forall C\bullet\ C \subseteq X \land Trich(C, \$<<) \\ \qquad\quad \Rightarrow \exists x\bullet x \in X \land UpperBound(C, \$<<, x)) \\ \quad \Rightarrow \quad \exists x\bullet\ x \in X \land \forall y\bullet\ y \in X \land \neg y = x \Rightarrow \neg x << y \end{array}$$

3.2 Lemmas about Subsets of Ordered Sets

These lemmas build up to the fact that a subset of a well-ordered set is well-ordered (which we use in the proof of the well-ordering principle).

$$\begin{array}{l} subrel\_well\_ordering\_thm = \vdash \forall X\ Y\ \$<<\bullet \\ \qquad Y \subseteq X \land WellOrdering(X, \$<<) \Rightarrow WellOrdering(Y, \$<<) \\ \\ subrel\_refl\_thm = \\ \qquad \vdash \forall\ X\ Y\ \$<<\bullet\ Y \subseteq X \land Refl\ (X, \$<<) \Rightarrow Refl\ (Y, \$<<) \end{array}$$

3.3 The Well-Ordering Principle

The following lemma is needed in the proof of the well-ordering principle. It shows that extending a well-ordering by adding an extra element above all the existing elements gives a well-ordering.

$$\begin{array}{l} well\_ordering\_extension\_lemma = \vdash \forall A\ x\ y\ z\ \$<<<\bullet \\ \qquad WellOrdering(A, \$<<<) \\ \land \qquad \neg x \in A \\ \Rightarrow \qquad WellOrdering(\\ \qquad\qquad A \cup \{x\}, \\ \qquad\qquad \lambda\ a\ b\bullet\ a \in A \land b \in A \land a <<< b \lor a \in A \land b = x \lor a = x \land b = x)\urcorner \end{array}$$

The proof of the well-ordering is much as per Halmos. Some care is needed to get the construction right and to avoid an embarrassment of case splits. We have to exhibit a well-ordering on some arbitrary set X. Since subsets of well-ordered sets are well-ordered it suffices to well-order the universe of the type of the elements of X. To do this we define U to be the set of all relations that well-order their field and are reflexive on their field:

SML

```
val u_def = ⌜∃U•
        U =
        {       R : 'a → 'a → BOOL
        |       WellOrdering({a | R a a}, R) ∧ ∀a b• R a b ⇒ R a a ∧ R b b}⌝;
```

U is partial ordered by saying $R<<<S$ iff. R is an Initial segment of S:

SML

```
val segment_def = ⌜∃$<<<• ∀R S: 'a → 'a → BOOL•
                R <<< S
        ⇔       (∀a b•R a b ⇒ S a b)
        ∧       (∀a b•R a a ∧ ¬R b b ∧ S b b ⇒ S a b)⌝;
```

We then verify that U and $<<<$ satisfy the hypotheses of Zorn's lemma:

SML

```
val zorn_hyps = ⌜
                Trans(U : ('a → 'a → BOOL) SET, $<<<)
        ∧       Antisym(U, $<<<)
        ∧       (∀ C• C ⊆ U ∧ Trich (C, $<<<) ⇒
                        (∃ x• x ∈ U ∧ UpperBound (C, $<<<, x)))⌝;
```

This verification, done naively would involve a large number of case splits in the situation where we have two members R and S of a chain, so that either R precedes S or S precedes R — one part of the formal proof would branch into 16 cases, if we proceeded in the naive way. Instead, we use the following lemma which shows that any two elements, R and S, of a chain are both contained (viewed as sets of pairs) in some other element, Q. Of course, Q can be taken to be one or other of R and S, but that does not matter for all but one of the properties we have to prove.. This property is that the upper bounds of chains satisfy the minimum condition, and the case split in question is unavoidable since the arguments are essentially different for the two cases.

SML

```
val key_lemma =⌜∀R S : 'a → 'a → BOOL•
                R ∈ C ∧ S ∈ C
        ⇒       ∃Q•
                Q ∈ C
        ∧       (∀a b•R a b ⇒ Q a b )
        ∧       (∀a b•S a b ⇒ Q a b )⌝;
```

Once we have the hypotheses of Zorn's lemma, we apply it to produce a maximal element for the initial segment relation and then prove that the field of such a maximal element must be the whole set. Most of the work for this has been done in the previous theorem about single element extensions of well-orderings.

```
well_ordering_thm = ⊢ ∀X :'a SET• ∃ $<<• WellOrdering(X, $<<)
```

3.4 Lemmas About Well-Orderings

3.5 Least Well-Orderings

There will be several distinct well-orderings of any given set. The order morphisms between these well-orderings well-order the order types of the well-orderings, and there will therefore be least well-orderings under this ordering. These least well-orderings correspond to enumerations by the initial ordinal corresponding to the cardinality of the set.

If we directly formalised the above sketch then we would be involved in developing parts of the theory of ordinals and cardinals to obtain the result that each set has a least well-ordering. However, once the basic idea is in place it can be re-engineered with less equipment.

The elements of any set, together with an arbitrary well-ordering of the set, will serve as surrogates for the ordinals sufficient for our purposes. If this arbitrary well-ordering is not the least, then all shorter well-orderings will be order-isomorpic to an initial segment of the given well-ordering, each of which may be represented by its strict supremum in the set. Such an inital segment of the well-ordering will be isomorphic to a well-ordering of the whole if it has the same cardinality as the whole, i.e. if there is an injection from the whole into the subset. The ordering of the whole can be obtained from the ordering of the part through the injection.

If we take the subset of elements of our original well-ordered set which have the same cardinality as the whole, then this subset will have a minimal element under the well-ordering, and the subset determined by that minimal element will yield a minimal well-ordering of the whole.

The purpose of this section is to prove the existence of such a minimal well-ordering of any set.

HOL Constant

```
MinimalWellOrdering : ('a SET × ('a → 'a → BOOL))
                → BOOL
────────────────────────────
∀ (X, $<<)•
        MinimalWellOrdering (X, $<<)
⇔      ∀$<<<• WellOrdering (X, $<<<) ⇒ (X, $<<) ≤_ot (X, $<<<)
```

4 WELL-FOUNDEDNESS

4.1 Strictness

The concept of well-foundedness is distinctive here in applying only to strict, i.e. irreflexive relations. This usage is adopted because well-foundedness is used as a criterion for judging the consistency of recursive definitions, which is secured if the values referred to on the right of a recursive definition are less than the value on the left under some strictly well-founded relationship. For this reason we require a well-founded relation to be irreflexive.

In this context the variant of the well-ordering theorem which delivers a strict (and hence well-founded) well-order may be of use, and more generally therefore the theorems which show that strict (irreflexive) versions of relations have most of the properties of interest here which are possessed by the original.

HOL Constant

$\boldsymbol{Strict} : ('a\ SET \times ('a \to 'a \to BOOL)) \to ('a\ SET \times ('a \to 'a \to BOOL))$

$\forall\ (X,\ \$<<)\bullet\ Strict\ (X,\ \$<<) = (X,\ \lambda x\ y \bullet\ x << y \wedge \neg\ x = y)$

[This is a bad definition, because it meddles with the relation outside its domain of relevance, and so may change a relation which is already irreflexive over its explicit domain. It would be better to define strict over the relation only, though perhaps it would be better not to work with this concept of relation at all and just to stick with the native HOL conception of a relation as curried property of pairs.]

A few lemmas for proving properties of strict relations:

IrreflStrict_thm =
$\vdash \forall\ (X,\ \$<<)\bullet\ Irrefl\ (Strict\ (X,\ \$<<))$

PartialOrderStrict_thm =
$\vdash \forall\ (X,\ \$<<)\bullet\ PartialOrder\ (X,\ \$<<) \Rightarrow PartialOrder\ (Strict\ (X,\ \$<<))$

TrichStrict_thm =
$\vdash \forall\ (X,\ \$<<)\bullet\ Trich\ (X,\ \$<<) \Rightarrow Trich\ (Strict\ (X,\ \$<<))$

LinearOrderStrict_thm =
$\vdash \forall\ (X,\ \$<<)\bullet\ LinearOrder\ (X,\ \$<<) \Rightarrow LinearOrder\ (Strict\ (X,\ \$<<))$

AntisymStrict_thm =
$\vdash \forall\ (X,\ \$<<)\bullet\ Antisym\ (X,\ \$<<) \Rightarrow Antisym\ (Strict\ (X,\ \$<<))$

WeakMinCondStrict_thm =
$\vdash \forall\ (X,\ \$<<)\bullet\ WeakMinCond\ (X,\ \$<<) \Rightarrow WeakMinCond\ (Strict\ (X,\ \$<<))$

WellOrderingStrict_thm =
$\vdash \forall\ (X,\ \$<<)\bullet\ WellOrdering\ (X,\ \$<<) \Rightarrow WellOrdering\ (Strict\ (X,\ \$<<))$

MinCondStrict_thm =
$\vdash \forall\ (X,\ \$<<)\bullet\ LinearOrder\ (X,\ \$<<) \wedge WeakMinCond\ (X,\ \$<<) \Rightarrow MinCond\ (Strict\ (X,\ \$<<))$

WellFoundedStrict_thm =
$\vdash \forall\ (X,\ \$<<)\bullet\ WellOrdering\ (X,\ \$<<) \Rightarrow WellFounded\ (Strict\ (X,\ \$<<))$

Then a wrinkle on the well-ordering theorem:

wf_well_ordering_thm =
$\vdash \forall X :'a\ SET\bullet\ \exists\ \$<<\bullet\ WellOrdering(X,\ \$<<) \wedge WellFounded(X,\ \$<<)$

We might as well have the concept of strict well-ordering.

HOL Constant

$$\textbf{StrictWellOrdering} : ('a\ SET \times ('a \rightarrow 'a \rightarrow BOOL)) \rightarrow BOOL$$

$$\forall\ (X,\ \${<<})\bullet\ StrictWellOrdering\ (X,\ \${<<})$$
$$\Leftrightarrow Irrefl\ (X,\ \${<<})\ \wedge\ WellOrdering\ (X,\ \${<<})$$

$$\textbf{StrictWellOrdering\_thm1} =$$
$$\vdash \forall\ (X,\ \${<<})\bullet\ WellOrdering\ (X,\ \${<<}) \Rightarrow StrictWellOrdering\ (Strict\ (X,\ \${<<}))$$

4.2 The Minimum Conditions and Well-Founded Relations

In this section we show the equivalence of our definitions with some other useful conditions. This selection of equivalences is currently just a sample. We could, for example, come up with induction principles characterising the minimum conditions.

A relation has the minimum condition iff. every descending sequence stabilises:

$$min\_cond\_descending\_sequence\_thm = \vdash \forall X\ \${<<}\bullet$$
$$MinCond(X,\ \${<<})$$
$$\Leftrightarrow\quad \forall f\bullet\ (\forall n\bullet f\ n \in X)\ \wedge\ (\forall n\bullet f(n+1)\ <<\ f\ n)$$
$$\Rightarrow \exists m\bullet\forall n\bullet m < n \Rightarrow f\ n = f\ m$$

A relation is well-founded iff. there are no infinite descending sequences:

$$well\_founded\_descending\_sequence\_thm = \vdash \forall X\ \${<<}\bullet$$
$$WellFounded(X,\ \${<<})$$
$$\Leftrightarrow\quad \neg\exists f\bullet\ (\forall n\bullet f\ n \in X)\ \wedge\ (\forall n\bullet f(n+1)\ <<\ f\ n)$$

A relation is well-founded iff. it enjoys the Noetherian(?) induction principle:

$$well\_founded\_induction\_thm = \vdash \forall X\ \${<<}\bullet$$
$$WellFounded(X,\ \${<<})$$
$$\Leftrightarrow\quad \forall p\bullet\ (\forall x\bullet x \in X\ \wedge\ (\forall y\bullet y \in X\ \wedge\ y << x \Rightarrow p\ y) \Rightarrow p\ x)$$
$$\Rightarrow (\forall x\bullet x \in X \Rightarrow p\ x)$$

A relation is a reflexive well-ordering iff. every non-empty subset has a unique lower bound:

$$refl\_well\_ordering\_lower\_bounds\_thm = \vdash \forall X\ \${<<}\bullet$$
$$Refl(X,\ \${<<})$$
$$\wedge\quad WellOrdering(X,\ \${<<})$$
$$\Leftrightarrow\quad \forall A\bullet A \subseteq X\ \wedge\ \neg A = \{\} \Rightarrow \exists_1\ x\bullet\ x \in A\ \wedge\ \forall y\bullet y \in A \Rightarrow x << y$$

4.3 Transitive Closure

First the definition:

HOL Constant

$$\textbf{\textit{TranClsr}} : ('a\ SET \times ('a \to 'a \to BOOL)) \to ('a\ SET \times ('a \to 'a \to BOOL))$$

$$\forall\ (X,\ \$<<)\bullet\ TranClsr\ (X,\ \$<<) = (X,\ \lambda x\ y\bullet$$
$$\forall r\bullet\ (Trans\ (X,\ r) \wedge \forall v\ w\bullet\ v \in X \wedge w \in X \wedge v << w \Rightarrow r\ v\ w) \Rightarrow r\ x\ y)$$

HOL Constant

$$\textbf{\textit{RefTranClsr}} : ('a\ SET \times ('a \to 'a \to BOOL)) \to ('a\ SET \times ('a \to 'a \to BOOL))$$

$$\forall\ (X,\ \$<<)\bullet\ RefTranClsr\ (X,\ \$<<) = (X,\ \lambda x\ y\bullet$$
$$x = y \vee Snd\ (TranClsr\ (X,\ \$<<))\ x\ y)$$

The following elementary facts about transitive closure prove useful:

$$trans\_tc\_thm = \quad \vdash \forall\ (X,\ \$<<)\bullet\ Trans\ (TranClsr\ (X,\ \$<<))$$

$$trans\_tc\_thm2 = \quad \vdash \forall\ (X,\ \$<<)\ x\ y\ z\bullet\ x \in X \wedge y \in X \wedge z \in X$$
$$\wedge\ Snd(TranClsr\ (X,\ \$<<))\ x\ y$$
$$\wedge\ Snd(TranClsr\ (X,\ \$<<))\ y\ z$$
$$\Rightarrow Snd(TranClsr\ (X,\ \$<<))\ x\ z$$

$$tc\_incr\_thm2 = \quad \vdash \forall\ (X,\ \$<<)\ x\ y\bullet\ x \in X \wedge y \in X$$
$$\wedge\ x << y$$
$$\Rightarrow Snd(TranClsr\ (X,\ \$<<))\ x\ y$$

$$tc\_decompose\_thm = \quad \vdash \forall\ (X,\ \$<<)\ x\ y\bullet\ x \in X \wedge y \in X$$
$$\wedge\ Snd\ (TranClsr\ (X,\ \$<<))\ x\ y$$
$$\wedge\ \neg\ x << y$$
$$\Rightarrow \exists z\bullet\ z \in X$$
$$\wedge\ Snd\ (TranClsr\ (X,\ \$<<))\ x\ z$$
$$\wedge\ z << y$$

$$tc\_mono\_thm = \vdash \forall\ (X,\ r1)\ (X,\ r2)\bullet$$
$$(\forall\ x\ y\bullet\ x \in X \wedge y \in X$$
$$\wedge\ r1\ x\ y$$
$$\Rightarrow r2\ x\ y)$$
$$\Rightarrow (\forall\ x\ y\bullet\ x \in X \wedge y \in X$$
$$\wedge\ Snd\ (TranClsr\ (X,\ r1))\ x\ y$$
$$\Rightarrow Snd\ (TranClsr\ (X,\ r2))\ x\ y)$$

When reasoning using well-founded induction a primitive induction principle can be converted into "course of values" induction by using the theorem which states that a relation is well-founded iff its transitive closure is well-founded.

In preparation for proving this theorem it is convenient to have a more general notion of inclusion of relations than that of section 3.2. This is helpful because a relation is a subrelation of its transitive closure, not a restriction of it to a smaller field. This suffices to establish that it is well founded if its transitive closure is. One might as well get it via the general result that subrelations of well-founded relation is well-founded, but it can't be done using a subrel theorem of the kind presented in section 3.2 earlier.

SML

```
declare_infix (300, "⊆_r");
```

HOL Constant

```
$⊆_r: ('a SET × ('a → 'a → BOOL)) → ('a SET × ('a → 'a → BOOL)) → BOOL
────────────────────────────────
∀ (X, r1) (Y, r2)• (X, r1) ⊆_r (Y, r2) ⇔
        ∀ x y• x ∈ X ∧ y ∈ X ∧ r1 x y
        ⇒ x ∈ Y ∧ y ∈ Y ∧ r2 x y
```

Allowing us to state concisely (and prove) that a relation is contained in its transitive closure:

```
r_⊆_r_tcr_thm =      ⊢ ∀r• r ⊆_r TranClsr r
```

And that various properties of relations also hold of their subrelations:

```
subrel_irrefl_thm2 =          ⊢ ∀r s• s ⊆_r r ∧ Irrefl r ⇒ Irrefl s
subrel_antisym_thm2 =         ⊢ ∀r s• s ⊆_r r ∧ Antisym r ⇒ Antisym s
subrel_min_cond_thm2 =        ⊢ ∀r s• s ⊆_r r ∧ MinCond r ⇒ MinCond s
subrel_well_founded_thm2 =    ⊢ ∀r s• s ⊆_r r ∧ WellFounded r ⇒ WellFounded s
```

A relation is well-founded iff its transitive closure is well-founded.

In the right to left direction the result follows from the fact that a relation is a subrelation of its transitive closure and that a subrelation of a well-founded relation is well-founded. In the left to right direction the proof involves showing by well-founded induction that there is no descending sequence in the transitive closure.

```
wf_iff_wftc_thm = ⊢ ∀ (X, $<<)•
        WellFounded (X, $<<) ⇔ WellFounded (TranClsr (X, $<<))
```

4.4 The Well-founded Part of a Relation

Relations which are not well-founded may have useful well-founded subrelations.

The well-founded part of a relation is the restriction of the relation to the subdomain of elements which are not the first element of any infinite descending sequence.

HOL Constant

$\boldsymbol{WfPart}$: $('a\ SET \times ('a \to 'a \to BOOL)) \to ('a\ SET \times ('a \to 'a \to BOOL))$

$\forall\ (X,\ \$<<)\bullet\ WfPart\ (X,\ \$<<) = ($
$\quad \{x \mid x \in X \land \neg\ \exists f\bullet\ f\ 0 = x \land \forall n\bullet\ (f\ n \in X \land (f\ (n+1)) << f\ n)\},$
$\quad \$<<)$

The well-founded part of a relation is of course, well-founded.

$wf\_part\_wf\_thm = \vdash \forall r\bullet\ WellFounded\ (WfPart\ r)$

4.5 The Recursion Theorem

The recursion theorem asserts the existence of fixed points of functionals under certain circumstances. This is helpful in establishing the consistency of inductive or recursive definitions of functions. The basic idea is that a recursive definition will be consistent if the recursion is well-founded.

Exactly how the result is best formulated depends on how you want to use it.

In HOL a suitable version of the recursion theorem will be of general applicability in consistency proofs for recursive definitions of HOL functions, which are of course, total over some type, so this is a prime application.

A special variant on this is when a new type is being introduced, corresponding to a subset of some existing type) and one of the primitive operators on the new type is defined recursively. Since the operator is primitive, it must be defined over a subset of the representation type and a version of the recursion theorem which yields a partial fixedpoint will be useful (a partial fixedpoint in this context is a total function which is fixed under the functional over some subset of its domain type). Since this is a generalisation of the recursion theorem more commonly applicable, this is the version I prove here.

A third case of interest is the recursive specification of functions in Z. In this case functions are represented as many-one relations. Its not sensible to address this variant here because it needs to be done in a context suitable for reasoning about Z, best done in a separate document.

The existence of a fixed point of a functional depends upon the recursion embodied in the functional being well-founded. The following two definitions allow us to talk about this formally.

A functional "FunctRespects" (informally, respects) a relation over some set if it delivers a function whose value at some point in that set is dependent only on the values of the operand function at the points in the set related to x under the relation.

HOL Constant

$$\textbf{FunctRespects}: (('a \to 'b) \to ('a \to 'b)) \to ('a\ SET \times ('a \to 'a \to BOOL)) \to BOOL$$

$$\forall G\ (X,\ \$<<)\bullet\ FunctRespects\ G\ (X,\ \$<<) \Leftrightarrow \forall g\ h\ x\bullet\ x \in X \Rightarrow (\forall y\bullet\ y \in X \land y << x \Rightarrow g\ y = h\ y) \Rightarrow G\ g\ x = G\ h\ x$$

A functional is well-founded if it respects some well-founded relation, and in that case will have a fixed point.

It would be nice if one could obtain from a functional a relation which that functional respects. I have unable to discover a way in which this can be done in the general case (other than the useless everywhere-true relationship, which has an empty well-founded part).

To express a recursion theorem giving partial fixed points of a functional we define the notion of partial equivalence of functions.

HOL Constant

$$\textbf{PartFunEquiv}: 'a\ SET \to ('a \to 'b) \to ('a \to 'b) \to BOOL$$

$$\forall X\ f\ g\bullet\ PartFunEquiv\ X\ f\ g \Leftrightarrow \forall x\bullet\ x \in X \Rightarrow f\ x = g\ x$$

which has the elementary property:

$$part\_fun\_equiv\_lemma1\ = \vdash \forall X\ Y\ f\ g\bullet\ PartFunEquiv\ X\ f\ g \land Y \subseteq X \Rightarrow PartFunEquiv\ Y\ f\ g$$

The following variant of reflexive transitive closure (yielding a set from an element) proves useful:

HOL Constant

$$\textbf{TcUpTo}: ('a\ SET \times ('a \to 'a \to BOOL)) \to 'a \to 'a\ SET$$

$$\forall (X,\ \$<<)\ x\bullet\ TcUpTo\ (X,\ \$<<)\ x = \{y \mid y \in X \land x \in X \land x = y \lor Snd\ (TranClsr\ (X,\ \$<<))\ y\ x\}$$

$$tcupto\_inc\_lemma1\ = \vdash \forall (X,\ \$<<)\ x\ y\bullet\ x \in X \land y \in X \land y << x \Rightarrow (TcUpTo\ (X,\ \$<<)\ y) \subseteq (TcUpTo\ (X,\ \$<<)\ x)$$

$$tcupto\_inc\_lemma2\ = \vdash \forall (X,\ \$<<)\ x\ y\bullet\ y \in X \land x \in (TcUpTo\ (X,\ \$<<)\ y) \Rightarrow (TcUpTo\ (X,\ \$<<)\ x) \subseteq (TcUpTo\ (X,\ \$<<)\ y)$$

The following two lemmas about *PartFunEquiv* are then immediate:

$$
\begin{array}{l}
part\_fun\_equiv\_lemma2\ =\ \vdash\ \forall X\ f\ g\ x\ y\bullet\ x \in X\ \wedge\ y \in X\\
\qquad \wedge\ PartFunEquiv\ (TcUpTo\ (X,\ \$<<)\ y)\ f\ g\\
\qquad \wedge\ x\ <<\ y\\
\qquad \Rightarrow\ PartFunEquiv\ (TcUpTo\ (X,\ \$<<)\ x)\ f\ g\\
\\
part\_fun\_equiv\_lemma3\ =\ \vdash\ \forall X\ Y\ f\ g\ x\ y\bullet\ y \in X\\
\qquad \wedge\ PartFunEquiv\ (TcUpTo\ (X,\ \$<<)\ y)\ f\ g\\
\qquad \wedge\ x \in (TcUpTo\ (X,\ \$<<)\ y)\\
\qquad \Rightarrow\ PartFunEquiv\ (TcUpTo\ (X,\ \$<<)\ x)\ f\ g
\end{array}
$$

HOL Constant

$$
\boldsymbol{UniquePartFixp}:\ 'a\ SET \rightarrow (('a \rightarrow 'b) \rightarrow ('a \rightarrow 'b)) \rightarrow BOOL
$$

$$
\begin{array}{l}
\forall\ X\ \ G\bullet\ UniquePartFixp\ X\ G \Leftrightarrow \exists\ f\bullet\\
\qquad PartFunEquiv\ X\ (G\ f)\ f\\
\qquad \wedge\ \forall g\bullet\ PartFunEquiv\ X\ (G\ g)\ g \Rightarrow PartFunEquiv\ X\ f\ g
\end{array}
$$

The proof of the recursion theorem is by well-founded induction on the transitive closure of a relation respected by the functional. The induction hypothesis is exposed in the following lemma:

$$
\begin{array}{l}
recursion\_theorem\_lemma1\ =\ \vdash\ \forall G\ (X,\ \$<<)\bullet\\
\qquad FunctRespects\ G\ (X,\ \$<<)\ \wedge\ WellFounded\ (X,\ \$<<)\\
\qquad \Rightarrow \forall x\bullet\ x \in X\ \Rightarrow\ UniquePartFixp\ (TcUpTo\ (X,\ \$<<)\ x)\ G
\end{array}
$$

The recursion theorem follows:

$$
\begin{array}{l}
recursion\_theorem\ =\ \vdash\ \forall (X,\ \$<<)\ G\bullet\\
\qquad FunctRespects\ G\ (X,\ \$<<)\ \wedge\ WellFounded\ (X,\ \$<<)\ \Rightarrow\ UniquePartFixp\ X\ G
\end{array}
$$

from which is readily obtained the specialisation to total functions:

$$
\begin{array}{l}
tf\_recursion\_thm\ =\ \vdash\ \forall \$<<\ G\bullet\\
\qquad FunctRespects\ G\ (Universe,\ \$<<)\ \wedge\ WellFounded\ (Universe,\ \$<<)\\
\qquad \Rightarrow\ \exists f\bullet\ (G\ f)\ =\ f
\end{array}
$$

It may be helpful to have a proforma which facilitates proof that a function respects some well-founded relation. One way of approaching this is to define a function which restricts some other function to the part of its domain strictly below some value. These will be the values which are accessible to a function computing its value at that point which respects the relation.

Here I assume the relation is over the whole type.

SML

```
declare_infix(400, "⟨◁");
```

HOL Constant

$\$\langle\lhd$: $('a \times ('a \to 'a \to BOOL)) \to ('a \to 'b) \to ('a \to 'b)$

$\forall\ x\ \${<<}\ f\bullet\ (x,\ \${<<})\ \langle\lhd\ f\ =\ \lambda y\bullet\ if\ \ y\ <<\ x\ \ then\ f\ \ y\ \ else\ \epsilon z\bullet T$

$\langle\lhd$_***fc_thm*** $= \vdash \forall\ y\ x\ \${<<}\ f\bullet\ y\ <<\ x \Rightarrow ((x,\ \${<<})\ \langle\lhd\ f)\ y = f\ y$

4.6 Minimal Well Orderings

The term *InitialWellOrdering* is used here for a function which returns the least well-ordering of some set. Every set has a least well-ordering, and the order type is that of an initial ordinal, hence the choice of name.

However, this fact has not yet been proven, so the definition is couched in such a way as to avoid making that assumption.

HOL Constant

MinWellOrdering : $('a\ SET \times ('a \to 'a \to BOOL)) \to BOOL$

$\forall\ X\ R\bullet\ MinWellOrdering\ (X,\ R) \Leftrightarrow WellOrdering\ (X,\ R)$
$\quad \land \quad \forall (Y, S{:}('a \to 'a \to BOOL))\bullet\ WellOrdering\ (Y,\ S)$
$\quad\quad \Rightarrow (X,\ R) \leqslant_{ot} (Y, S)$

The following definition is written so that from it one can establish that the *InitialStrictWellOrdering* of a set is a well-founded well-ordering without having to prove the existence of least or initial well-orderings.

HOL Constant

InitialStrictWellOrdering : $('a\ SET \times ('a \to 'a \to BOOL)) \to BOOL$

$\forall\ X\ R\bullet\ InitialStrictWellOrdering\ (X,\ R) \Leftrightarrow$
$\quad WellOrdering\ (X,\ R) \land WellFounded\ (X,\ R)$
$\quad \land ((\exists S\bullet\ MinWellOrdering\ (X,\ S)) \Rightarrow MinWellOrdering\ (X,\ R))$

We now obtain the somewhat fraudulent theorem that every non-empty set has an InitialStrictWellOrdering:

HOL Constant

AnInitialStrictWellOrdering : $'a\ SET \to ('a \to 'a \to BOOL)$

$\forall\ X\bullet\ AnInitialStrictWellOrdering\ X\ =$
$\quad \epsilon R\bullet\quad InitialStrictWellOrdering\ (X,\ R)$

We can now define a constant which is an initial strict well-ordering of its type.

SML

```
declare_infix(210, "<_iwo");
```

HOL Constant

$\$<_{iwo} : ('a \to 'a \to BOOL)$

$InitialStrictWellOrdering(Universe, \$<_{iwo})$

5 RELATIONS OVER A TYPE

The material in the next two sections is placed in a new theory *U_orders*.

All of our definitions are explicit about the ordered set X on which the property of interest is required to hold. In many applications, X will be the universe of a type and the definitions and the statements of some of the theorems simplify. The relevant definitions may be given concisely as follows:

HOL Constant

$UIrrefl : ('a \to 'a \to BOOL) \to BOOL;$
$UAntisym : ('a \to 'a \to BOOL) \to BOOL;$
$UTrans : ('a \to 'a \to BOOL) \to BOOL;$
$UStrictPartialOrder : ('a \to 'a \to BOOL) \to BOOL;$
$UTrich : ('a \to 'a \to BOOL) \to BOOL;$
$UStrictLinearOrder : ('a \to 'a \to BOOL) \to BOOL;$
$UComplete : ('a \to 'a \to BOOL) \to BOOL;$
$URefl : ('a \to 'a \to BOOL) \to BOOL;$
$UPartialOrder : ('a \to 'a \to BOOL) \to BOOL;$
$ULinearOrder : ('a \to 'a \to BOOL) \to BOOL;$
$UWeakMinCond : ('a \to 'a \to BOOL) \to BOOL;$
$UMinCond : ('a \to 'a \to BOOL) \to BOOL;$
$UWellOrdering : ('a \to 'a \to BOOL) \to BOOL;$
$UWellFounded : ('a \to 'a \to BOOL) \to BOOL$

$UIrrefl = Curry\ Irrefl\ Universe$
$\land \quad UAntisym = Curry\ Antisym\ Universe$
$\land \quad UTrans = Curry\ Trans\ Universe$
$\land \quad UStrictPartialOrder = Curry\ StrictPartialOrder\ Universe$
$\land \quad UTrich = Curry\ Trich\ Universe$
$\land \quad UStrictLinearOrder = Curry\ StrictLinearOrder\ Universe$
$\land \quad UComplete = Curry\ Complete\ Universe$
$\land \quad URefl = Curry\ Refl\ Universe$
$\land \quad UPartialOrder = Curry\ PartialOrder\ Universe$

| $\wedge$ $ULinearOrder = Curry\ LinearOrder\ Universe$
| $\wedge$ $UWeakMinCond = Curry\ WeakMinCond\ Universe$
| $\wedge$ $UMinCond = Curry\ MinCond\ Universe$
| $\wedge$ $UWellOrdering = Curry\ WellOrdering\ Universe$
| $\wedge$ $UWellFounded = Curry\ WellFounded\ Universe$

Now we automatically derive the defining properties of all of these constants in terms of primitive notions or other constants in the set.

6 INDEPENDENCE

We want to show that for any pair of the primitive notions, P and Q, there are relations for which both hold, P holds but not Q and Q holds but not P. Somewhat more subtly, we want to do this without assuming that the primitive notions are distinct. (I.e., it is not good enough to assume that reflexivity and transitivity are distinct notions, for example, when exhibiting a relation that has one property but not the other).

The examples will all use relations on the natural numbers. The weak minimum condition turns out to be the hardest property to prove in the examples; the following lemma serves in all but one case (viz. weak minimum condition and not antisymmetry, for which, happily, the relation that relates any two natural numbers does the job).

For the cases where we need to show that the primitive properties are compatible, our example are be (with one exception), the usual order structure on the natural numbers in its reflexive or irreflexive guise. The exception turns out to be when we have to exhibit a relation which is both reflexive and irreflexive.

A The Theory ordered_sets

A.1 Parents

rbjmisc *set_thms* *orders*

A.2 Children

ba *membership* *U_orders*

A.3 Constants

WeakMinCond $'a\ \mathbb{P} \times ('a \to 'a \to BOOL) \to BOOL$

PartialOrder $'a\ \mathbb{P} \times ('a \to 'a \to BOOL) \to BOOL$

LinearOrder $'a\ \mathbb{P} \times ('a \to 'a \to BOOL) \to BOOL$

MinCond $'a\ \mathbb{P} \times ('a \to 'a \to BOOL) \to BOOL$

WellOrdering $'a\ \mathbb{P} \times ('a \to 'a \to BOOL) \to BOOL$

WellFounded $'a\ \mathbb{P} \times ('a \to 'a \to BOOL) \to BOOL$

OrderMorphism
$'a\ \mathbb{P} \times ('a \to 'a \to BOOL)$
$\to 'b\ \mathbb{P} \times ('b \to 'b \to BOOL)$
$\to ('a \to 'b)$
$\to BOOL$

OrdMorph $'a\ \mathbb{P} \to ('a \to 'a \to BOOL) \to ('a \to 'a \to BOOL) \to BOOL$

$\$\leqslant_{ot}$ $'a\ \mathbb{P} \times ('a \to 'a \to BOOL)$
$\to 'b\ \mathbb{P} \times ('b \to 'b \to BOOL)$
$\to BOOL$

MinimalWellOrdering
$'a\ \mathbb{P} \times ('a \to 'a \to BOOL) \to BOOL$

Strict $'a\ \mathbb{P} \times ('a \to 'a \to BOOL) \to 'a\ \mathbb{P} \times ('a \to 'a \to BOOL)$

StrictWellOrdering
$'a\ \mathbb{P} \times ('a \to 'a \to BOOL) \to BOOL$

TranClsr $'a\ \mathbb{P} \times ('a \to 'a \to BOOL) \to 'a\ \mathbb{P} \times ('a \to 'a \to BOOL)$

RefTranClsr $'a\ \mathbb{P} \times ('a \to 'a \to BOOL) \to 'a\ \mathbb{P} \times ('a \to 'a \to BOOL)$

$\$\subseteq_{r}$ $'a\ \mathbb{P} \times ('a \to 'a \to BOOL)$
$\to 'a\ \mathbb{P} \times ('a \to 'a \to BOOL)$
$\to BOOL$

WfPart $'a\ \mathbb{P} \times ('a \to 'a \to BOOL) \to 'a\ \mathbb{P} \times ('a \to 'a \to BOOL)$

FunctRespects
$(('a \to 'b) \to 'a \to 'b) \to 'a\ \mathbb{P} \times ('a \to 'a \to BOOL) \to BOOL$

PartFunEquiv $'a\ \mathbb{P} \to ('a \to 'b) \to ('a \to 'b) \to BOOL$

TcUpTo $'a\ \mathbb{P} \times ('a \to 'a \to BOOL) \to 'a \to 'a\ \mathbb{P}$

UniquePartFixp
$'a\ \mathbb{P} \to (('a \to 'b) \to 'a \to 'b) \to BOOL$

$\$\langle\lhd$ $'a \times ('a \to 'a \to BOOL) \to ('a \to 'b) \to 'a \to 'b$

MinWellOrdering
$'a\ \mathbb{P} \times ('a \to 'a \to BOOL) \to BOOL$

InitialStrictWellOrdering
$'a\ \mathbb{P} \times ('a \to 'a \to BOOL) \to BOOL$

AnInitialStrictWellOrdering $'a\ \mathbb{P} \to 'a \to 'a \to BOOL$

$\$<_{iwo}$ $'a \to 'a \to BOOL$

A.4 Fixity

Right Infix 210:

$<<<\quad <_{iwo}\quad \leqslant_{ot}$

Right Infix 300:

$\subseteq_r$

Right Infix 400:

$\langle\lhd$

A.5 Definitions

WeakMinCond

$$\begin{array}{l} \vdash \forall\ (X,\ \$<<) \\ \quad \bullet\ WeakMinCond\ (X,\ \$<<) \\ \qquad \Leftrightarrow (\forall\ A \\ \qquad \bullet\ A \subseteq X \land \neg\ A = \{\} \\ \qquad\quad \Rightarrow (\exists\ x \\ \qquad\quad \bullet\ x \in A \\ \qquad\qquad \land (\forall\ y \\ \qquad\qquad \bullet\ y \in A \land y << x \land \neg\ y = x \Rightarrow x << y))) \end{array}$$

PartialOrder

$$\begin{array}{l} \vdash \forall\ (X,\ \$<<) \\ \quad \bullet\ PartialOrder\ (X,\ \$<<) \\ \qquad \Leftrightarrow Antisym\ (X,\ \$<<) \land Trans\ (X,\ \$<<) \end{array}$$

LinearOrder

$$\begin{array}{l} \vdash \forall\ (X,\ \$<<) \\ \quad \bullet\ LinearOrder\ (X,\ \$<<) \\ \qquad \Leftrightarrow PartialOrder\ (X,\ \$<<) \land Trich\ (X,\ \$<<) \end{array}$$

MinCond

$$\begin{array}{l} \vdash \forall\ (X,\ \$<<) \\ \quad \bullet\ MinCond\ (X,\ \$<<) \\ \qquad \Leftrightarrow Antisym\ (X,\ \$<<) \land WeakMinCond\ (X,\ \$<<) \end{array}$$

WellOrdering

$$\begin{array}{l} \vdash \forall\ (X,\ \$<<) \\ \quad \bullet\ WellOrdering\ (X,\ \$<<) \\ \qquad \Leftrightarrow LinearOrder\ (X,\ \$<<) \land WeakMinCond\ (X,\ \$<<) \end{array}$$

WellFounded

$$\begin{array}{l} \vdash \forall\ (X,\ \$<<) \\ \quad \bullet\ WellFounded\ (X,\ \$<<) \\ \qquad \Leftrightarrow Irrefl\ (X,\ \$<<) \land MinCond\ (X,\ \$<<) \end{array}$$

OrderMorphism

$$\begin{array}{l} \vdash \forall\ m\ (X,\ \$<<)\ (Y,\ \$<<<) \\ \quad \bullet\ OrderMorphism\ (X,\ \$<<)\ (Y,\ \$<<<)\ m \\ \qquad \Leftrightarrow (\forall\ x1\ x2 \\ \qquad \bullet\ x1 \in X \land x2 \in X \\ \qquad\quad \Rightarrow m\ x1 \in Y \\ \qquad\qquad \land m\ x2 \in Y \\ \qquad\qquad \land (x1 << x2 \Leftrightarrow m\ x1 <<< m\ x2)) \end{array}$$

OrdMorph

$$\begin{array}{l} \vdash \forall\ X\ \$<<\ \$<<< \\ \quad \bullet\ OrdMorph\ X\ \$<<\ \$<<< \\ \qquad \Leftrightarrow (\exists\ m \bullet\ OrderMorphism\ (X,\ \$<<)\ (X,\ \$<<<)\ m) \end{array}$$

$\leqslant_{ot}$

$$\vdash \forall\ (X,\ \$<<)\ (Y,\ \$<<<)$$

$\bullet\ (X,\ \$<<) \leqslant_{ot} (Y,\ \$<<<)$
$\quad \Leftrightarrow (\exists\ m \bullet\ \mathit{OrderMorphism}\ (X,\ \$<<)\ (Y,\ \$<<<)\ m)$

MinimalWellOrdering
$\vdash \forall\ (X,\ \$<<)$
$\bullet\ \mathit{MinimalWellOrdering}\ (X,\ \$<<)$
$\quad \Leftrightarrow (\forall\ \$<<<$
$\quad \bullet\ \mathit{WellOrdering}\ (X,\ \$<<<)$
$\qquad \Rightarrow (X,\ \$<<) \leqslant_{ot} (X,\ \$<<<))$

Strict $\vdash \forall\ (X,\ \$<<)$
$\bullet\ \mathit{Strict}\ (X,\ \$<<) = (X,\ (\lambda\ x\ y \bullet\ x << y \wedge \neg\ x = y))$

StrictWellOrdering
$\vdash \forall\ (X,\ \$<<)$
$\bullet\ \mathit{StrictWellOrdering}\ (X,\ \$<<)$
$\quad \Leftrightarrow \mathit{Irrefl}\ (X,\ \$<<) \wedge \mathit{WellOrdering}\ (X,\ \$<<)$

TranClsr $\vdash \forall\ (X,\ \$<<)$
$\bullet\ \mathit{TranClsr}\ (X,\ \$<<)$
$\quad = (X,$
$\quad\ (\lambda\ x\ y$
$\qquad \bullet\ \forall\ r$
$\qquad\ \bullet\ \mathit{Trans}\ (X,\ r)$
$\qquad\qquad \wedge (\forall\ v\ w$
$\qquad\qquad \bullet\ v \in X \wedge w \in X \wedge v << w \Rightarrow r\ v\ w)$
$\qquad\quad \Rightarrow r\ x\ y))$

RefTranClsr $\vdash \forall\ (X,\ \$<<)$
$\bullet\ \mathit{RefTranClsr}\ (X,\ \$<<)$
$\quad = (X,$
$\quad\ (\lambda\ x\ y \bullet\ x = y \vee \mathit{Snd}\ (\mathit{TranClsr}\ (X,\ \$<<))\ x\ y))$

$\subseteq_{\boldsymbol{r}}$ $\vdash \forall\ (X,\ r1)\ (Y,\ r2)$
$\bullet\ (X,\ r1) \subseteq_r (Y,\ r2)$
$\quad \Leftrightarrow (\forall\ x\ y$
$\quad \bullet\ x \in X \wedge y \in X \wedge r1\ x\ y$
$\qquad \Rightarrow x \in Y \wedge y \in Y \wedge r2\ x\ y)$

WfPart $\vdash \forall\ (X,\ \$<<)$
$\bullet\ \mathit{WfPart}\ (X,\ \$<<)$
$\quad = (\{x$
$\qquad | x \in X$
$\qquad\quad \wedge \neg\ (\exists\ f$
$\qquad\qquad \bullet\ f\ 0 = x$
$\qquad\qquad\quad \wedge (\forall\ n$
$\qquad\qquad\quad \bullet\ f\ n \in X \wedge f\ (n + 1) << f\ n))\},$
$\quad\ \$<<)$

FunctRespects
$\vdash \forall\ G\ (X,\ \$<<)$
$\bullet\ \mathit{FunctRespects}\ G\ (X,\ \$<<)$
$\quad \Leftrightarrow (\forall\ g\ h\ x$
$\quad \bullet\ x \in X$
$\qquad \Rightarrow (\forall\ y \bullet\ y \in X \wedge y << x \Rightarrow g\ y = h\ y)$
$\qquad \Rightarrow G\ g\ x = G\ h\ x)$

PartFunEquiv $\vdash \forall\ X\ f\ g$
$\bullet\ \mathit{PartFunEquiv}\ X\ f\ g \Leftrightarrow (\forall\ x \bullet\ x \in X \Rightarrow f\ x = g\ x)$

TcUpTo $\vdash \forall\ (X,\ \$<<)\ x$

```
          • TcUpTo (X, $<<) x
              = {y
              |y ∈ X ∧ x ∈ X ∧ x = y
                  ∨ Snd (TranClsr (X, $<<)) y x}
```

UniquePartFixp

```
        ⊢ ∀ X G
          • UniquePartFixp X G
              ⇔ (∃ f
              • PartFunEquiv X (G f) f
                  ∧ (∀ g
                  • PartFunEquiv X (G g) g
                      ⇒ PartFunEquiv X f g))
```

⟨◁

```
        ⊢ ∀ x $<< f
          • (x, $<<) ⟨◁ f
              = (λ y• if y << x then f y else ε z• T)
```

MinWellOrdering

```
        ⊢ ∀ X R
          • MinWellOrdering (X, R)
              ⇔ WellOrdering (X, R)
                ∧ (∀ (Y, S)
                • WellOrdering (Y, S) ⇒ (X, R) ≤_ot (Y, S))
```

InitialStrictWellOrdering

```
        ⊢ ∀ X R
          • InitialStrictWellOrdering (X, R)
              ⇔ WellOrdering (X, R)
                ∧ WellFounded (X, R)
                ∧ ((∃ S• MinWellOrdering (X, S))
                  ⇒ MinWellOrdering (X, R))
```

AnInitialStrictWellOrdering

```
        ⊢ ∀ X
          • AnInitialStrictWellOrdering X
              = (ε R• InitialStrictWellOrdering (X, R))
```

$<_{iwo}$

```
        ⊢ ConstSpec
          (λ <_iwo'
            • InitialStrictWellOrdering (Universe, <_iwo'))
          $<_iwo
```

A.6 Theorems

min_cond_def_thm

```
        ⊢ ∀ X $<<
          • MinCond (X, $<<)
              ⇔ (∀ A
              • A ⊆ X ∧ ¬ A = {}
                  ⇒ (∃ x
                  • x ∈ A ∧ (∀ y• y ∈ A ∧ ¬ y = x ⇒ ¬ y << x)))
```

well_founded_thm

```
        ⊢ ∀ X $<<
          • WellFounded (X, $<<)
              ⇔ (∀ A
              • A ⊆ X ∧ ¬ A = {}
```

```
                    ⇒ (∃ x• x ∈ A ∧ (∀ y• y ∈ A ⇒ ¬ y << x)))
well_ordering_def_thm
              ⊢ ∀ X $<<
                • WellOrdering (X, $<<)
                  ⇔ LinearOrder (X, $<<) ∧ MinCond (X, $<<)
chain_extension_thm
              ⊢ ∀ X $<< C D
                • Trans (X, $<<)
                      ∧ C ⊆ X
                      ∧ Trich (C, $<<)
                      ∧ D ⊆ X
                      ∧ Trich (D, $<<)
                      ∧ (∀ x y• x ∈ C ∧ y ∈ D ⇒ x << y)
                    ⇒ Trich (C ∪ D, $<<)
chain_singleton_thm
              ⊢ ∀ $<< x• Trich ({x}, $<<)
chain_singleton_extension_thm
              ⊢ ∀ X $<< C x
                • Trans (X, $<<)
                      ∧ C ⊆ X
                      ∧ Trich (C, $<<)
                      ∧ x ∈ X
                      ∧ (∀ y• y ∈ C ⇒ y << x)
                    ⇒ Trich (C ∪ {x}, $<<)
zorn_thm2     ⊢ ∀ X $<<
                • Trans (X, $<<)
                      ∧ Antisym (X, $<<)
                      ∧ (∀ C
                      • C ⊆ X ∧ Trich (C, $<<)
                          ⇒ (∃ x• x ∈ X ∧ UpperBound (C, $<<, x)))
                    ⇒ (∃ x
                    • x ∈ X ∧ (∀ y• y ∈ X ∧ ¬ y = x ⇒ ¬ x << y))
subrel_irrefl_thm
              ⊢ ∀ X Y $<<• Y ⊆ X ∧ Irrefl (X, $<<) ⇒ Irrefl (Y, $<<)
subrel_refl_thm
              ⊢ ∀ X Y $<<• Y ⊆ X ∧ Refl (X, $<<) ⇒ Refl (Y, $<<)
subrel_antisym_thm
              ⊢ ∀ X Y $<<
                • Y ⊆ X ∧ Antisym (X, $<<) ⇒ Antisym (Y, $<<)
subrel_trans_thm
              ⊢ ∀ X Y $<<• Y ⊆ X ∧ Trans (X, $<<) ⇒ Trans (Y, $<<)
subrel_trich_thm
              ⊢ ∀ X Y $<<• Y ⊆ X ∧ Trich (X, $<<) ⇒ Trich (Y, $<<)
subrel_partial_order_thm
              ⊢ ∀ X Y $<<
                • Y ⊆ X ∧ PartialOrder (X, $<<)
                  ⇒ PartialOrder (Y, $<<)
subrel_linear_order_thm
              ⊢ ∀ X Y $<<
                • Y ⊆ X ∧ LinearOrder (X, $<<) ⇒ LinearOrder (Y, $<<)
subrel_min_cond_thm
```

$$\vdash \forall\ X\ \ Y\ \$<< \bullet\ Y \subseteq X \land MinCond\ (X, \$<<) \Rightarrow MinCond\ (Y, \$<<)$$

subrel_well_ordering_thm

$$\vdash \forall\ X\ \ Y\ \$<< \bullet\ Y \subseteq X \land WellOrdering\ (X, \$<<) \Rightarrow WellOrdering\ (Y, \$<<)$$

well_ordering_extension_lemma

$$\vdash \forall\ A\ x\ y\ z\ \$<<< \bullet\ WellOrdering\ (A, \$<<<) \land \lnot\ x \in A \Rightarrow WellOrdering\ (A \cup \{x\}, (\lambda\ a\ b \bullet a \in A \land b \in A \land a <<< b \lor a \in A \land b = x \lor a = x \land b = x))$$

well_ordering_thm

$$\vdash \forall\ X \bullet \exists\ \$<< \bullet\ WellOrdering\ (X, \$<<)$$

IrreflStrict_thm

$$\vdash \forall\ (X, \$<<) \bullet\ Irrefl\ (Strict\ (X, \$<<))$$

PartialOrderStrict_thm

$$\vdash \forall\ (X, \$<<) \bullet\ PartialOrder\ (X, \$<<) \Rightarrow PartialOrder\ (Strict\ (X, \$<<))$$

TrichStrict_thm

$$\vdash \forall\ (X, \$<<) \bullet\ Trich\ (X, \$<<) \Rightarrow Trich\ (Strict\ (X, \$<<))$$

LinearOrderStrict_thm

$$\vdash \forall\ (X, \$<<) \bullet\ LinearOrder\ (X, \$<<) \Rightarrow LinearOrder\ (Strict\ (X, \$<<))$$

AntisymStrict_thm

$$\vdash \forall\ (X, \$<<) \bullet\ Antisym\ (X, \$<<) \Rightarrow Antisym\ (Strict\ (X, \$<<))$$

WeakMinCondStrict_thm

$$\vdash \forall\ (X, \$<<) \bullet\ WeakMinCond\ (X, \$<<) \Rightarrow WeakMinCond\ (Strict\ (X, \$<<))$$

WellOrderingStrict_thm

$$\vdash \forall\ (X, \$<<) \bullet\ WellOrdering\ (X, \$<<) \Rightarrow WellOrdering\ (Strict\ (X, \$<<))$$

MinCondStrict_thm

$$\vdash \forall\ (X, \$<<) \bullet\ LinearOrder\ (X, \$<<) \land WeakMinCond\ (X, \$<<) \Rightarrow MinCond\ (Strict\ (X, \$<<))$$

WellFoundedStrict_thm

$$\vdash \forall\ (X, \$<<) \bullet\ WellOrdering\ (X, \$<<) \Rightarrow WellFounded\ (Strict\ (X, \$<<))$$

wf_well_ordering_thm

$$\vdash \forall\ X \bullet \exists\ \$<< \bullet\ WellOrdering\ (X, \$<<) \land WellFounded\ (X, \$<<)$$

StrictWellOrdering_thm1

$\vdash \forall (X, \$<<)$
$\bullet$ *WellOrdering* $(X, \$<<)$
$\Rightarrow$ *StrictWellOrdering* (*Strict* $(X, \$<<))$

min_cond_descending_sequence_thm

$\vdash \forall X\ \$<<$
$\bullet$ *MinCond* $(X, \$<<)$
$\Leftrightarrow (\forall f$
$\bullet (\forall n \bullet f\ n \in X) \wedge (\forall n \bullet f\ (n + 1) << f\ n)$
$\Rightarrow (\exists m \bullet \forall n \bullet m < n \Rightarrow f\ n = f\ m))$

well_founded_descending_sequence_thm

$\vdash \forall X\ \$<<$
$\bullet$ *WellFounded* $(X, \$<<)$
$\Leftrightarrow \neg (\exists f$
$\bullet (\forall n \bullet f\ n \in X) \wedge (\forall n \bullet f\ (n + 1) << f\ n))$

well_founded_induction_thm

$\vdash \forall X\ \$<<$
$\bullet$ *WellFounded* $(X, \$<<)$
$\Leftrightarrow (\forall p$
$\bullet (\forall x$
$\bullet x \in X \wedge (\forall y \bullet y \in X \wedge y << x \Rightarrow p\ y) \Rightarrow p\ x)$
$\Rightarrow (\forall x \bullet x \in X \Rightarrow p\ x))$

refl_well_ordering_lower_bounds_thm

$\vdash \forall X\ \$<<$
$\bullet$ *Refl* $(X, \$<<) \wedge$ *WellOrdering* $(X, \$<<)$
$\Leftrightarrow (\forall A$
$\bullet A \subseteq X \wedge \neg A = \{\}$
$\Rightarrow (\exists_1 x \bullet x \in A \wedge (\forall y \bullet y \in A \Rightarrow x << y)))$

trans_tc_thm $\vdash \forall (X, \$<<) \bullet$ *Trans* (*TranClsr* $(X, \$<<))$

fst_tc_lemma $\vdash \forall (X, \$<<) \bullet$ *Fst* (*TranClsr* $(X, \$<<)) = X$

tran_clsr_thm

$\vdash \forall r$
$\bullet$ *TranClsr* r
$= ($*Fst* r,
$(\lambda x\ y$
$\bullet \forall s$
$\bullet$ *Trans* (*Fst* r, s)
$\wedge (\forall v\ w$
$\bullet v \in$ *Fst* $r \wedge w \in$ *Fst* $r \wedge$ *Snd* $r\ v\ w$
$\Rightarrow s\ v\ w)$
$\Rightarrow s\ x\ y))$

trans_tc_thm2

$\vdash \forall (X, \$<<)\ x\ y\ z$
$\bullet x \in X$
$\wedge y \in X$
$\wedge z \in X$
$\wedge$ *Snd* (*TranClsr* $(X, \$<<))\ x\ y$
$\wedge$ *Snd* (*TranClsr* $(X, \$<<))\ y\ z$
$\Rightarrow$ *Snd* (*TranClsr* $(X, \$<<))\ x\ z$

tc_incr_thm2 $\vdash \forall (X, \$<<)\ x\ y$
$\bullet x \in X \wedge y \in X \wedge x << y$

$\Rightarrow Snd\ (TranClsr\ (X, \${<<}))\ x\ y$

tc_decompose_thm

$\vdash \forall\ (X, \${<<})\ x\ y$
$\bullet\ x \in X$
$\land\ y \in X$
$\land\ Snd\ (TranClsr\ (X, \${<<}))\ x\ y$
$\land\ \neg\ x << y$
$\Rightarrow (\exists\ z$
$\bullet\ z \in X\ \land\ Snd\ (TranClsr\ (X, \${<<}))\ x\ z\ \land\ z << y)$

tc_mono_thm $\vdash \forall\ (X, r1)\ (X, r2)$
$\bullet\ (\forall\ x\ y \bullet\ x \in X\ \land\ y \in X\ \land\ r1\ x\ y \Rightarrow r2\ x\ y)$
$\Rightarrow (\forall\ x\ y$
$\bullet\ x \in X\ \land\ y \in X\ \land\ Snd\ (TranClsr\ (X, r1))\ x\ y$
$\Rightarrow Snd\ (TranClsr\ (X, r2))\ x\ y)$

***tran_clsr_lemma*1**

$\vdash \forall\ (X, \${<<})\ x\ y$
$\bullet\ Snd\ (TranClsr\ (X, \${<<}))\ x\ y \Rightarrow x \in X\ \land\ y \in X$

⊆r_thm $\vdash \forall\ r1\ r2$
$\bullet\ r1 \subseteq_r r2$
$\Leftrightarrow (\forall\ x\ y$
$\bullet\ x \in Fst\ r1\ \land\ y \in Fst\ r1\ \land\ Snd\ r1\ x\ y$
$\Rightarrow x \in Fst\ r2\ \land\ y \in Fst\ r2\ \land\ Snd\ r2\ x\ y)$

r_⊆_tcr_thm $\vdash \forall\ r \bullet\ r \subseteq_r TranClsr\ r$

***subrel_irrefl_thm*2**

$\vdash \forall\ r\ s \bullet\ s \subseteq_r r\ \land\ Irrefl\ r \Rightarrow Irrefl\ s$

***subrel_antisym_thm*2**

$\vdash \forall\ r\ s \bullet\ s \subseteq_r r\ \land\ Antisym\ r \Rightarrow Antisym\ s$

***subrel_min_cond_thm*2**

$\vdash \forall\ r\ s \bullet\ s \subseteq_r r\ \land\ MinCond\ r \Rightarrow MinCond\ s$

***subrel_well_founded_thm*2**

$\vdash \forall\ r\ s \bullet\ s \subseteq_r r\ \land\ WellFounded\ r \Rightarrow WellFounded\ s$

tcwf_not_refl_thm

$\vdash \forall\ (X, \${<<})\ x\ y$
$\bullet\ WellFounded\ (X, \${<<})\ \land\ x \in X$
$\Rightarrow \neg\ Snd\ (TranClsr\ (X, \${<<}))\ x\ x$

wf_wftc_thm $\vdash \forall\ r \bullet\ WellFounded\ r \Rightarrow WellFounded\ (TranClsr\ r)$

wftc_wf_thm $\vdash \forall\ r \bullet\ WellFounded\ (TranClsr\ r) \Rightarrow WellFounded\ r$

wf_iff_wftc_thm

$\vdash \forall\ r \bullet\ WellFounded\ r \Leftrightarrow WellFounded\ (TranClsr\ r)$

wfpart_wf_thm

$\vdash \forall\ r \bullet\ WellFounded\ (WfPart\ r)$

***part_fun_equiv_lemma*1**

$\vdash \forall\ X\ Y\ f\ g$
$\bullet\ PartFunEquiv\ X\ f\ g\ \land\ Y \subseteq X \Rightarrow PartFunEquiv\ Y\ f\ g$

tc_up_to_eq_thm

$\vdash \forall\ (X, \${<<})\ x$
$\bullet\ x \in X$
$\Rightarrow TcUpTo\ (X, \${<<})\ x$
$= \{y | Snd\ (RefTranClsr\ (X, \${<<}))\ y\ x\}$

***tcupto_inc_lemma*1**

$\vdash \forall\ (X, \${<<})\ x\ y$

$\bullet\ x \in X \wedge y \in X \wedge y << x$
$\Rightarrow TcUpTo\ (X, \$<<)\ y \subseteq TcUpTo\ (X, \$<<)\ x$

tcupto_inc_lemma2
$\vdash \forall\ (X, \$<<)\ x\ y$
$\bullet\ y \in X \wedge x \in TcUpTo\ (X, \$<<)\ y$
$\Rightarrow TcUpTo\ (X, \$<<)\ x \subseteq TcUpTo\ (X, \$<<)\ y$

part_fun_equiv_lemma2
$\vdash \forall\ X\ f\ g\ x\ y$
$\bullet\ x \in X$
$\wedge\ y \in X$
$\wedge\ PartFunEquiv\ (TcUpTo\ (X, \$<<)\ y)\ f\ g$
$\wedge\ x << y$
$\Rightarrow PartFunEquiv\ (TcUpTo\ (X, \$<<)\ x)\ f\ g$

part_fun_equiv_lemma3
$\vdash \forall\ X\ Y\ f\ g\ x\ y$
$\bullet\ y \in X$
$\wedge\ PartFunEquiv\ (TcUpTo\ (X, \$<<)\ y)\ f\ g$
$\wedge\ x \in TcUpTo\ (X, \$<<)\ y$
$\Rightarrow PartFunEquiv\ (TcUpTo\ (X, \$<<)\ x)\ f\ g$

unique_val_lemma
$\vdash \exists\ UniqueVal$
$\bullet\ \forall\ (X, \$<<)\ G\ x$
$\bullet\ x \in X \wedge UniquePartFixp\ (TcUpTo\ (X, \$<<)\ x)\ G$
$\Rightarrow (\forall\ f$
$\bullet\ PartFunEquiv\ (TcUpTo\ (X, \$<<)\ x)\ (G\ f)\ f$
$\Rightarrow UniqueVal\ (X, \$<<)\ G\ x = f\ x)$

recursion_theorem_lemma1
$\vdash \forall\ G\ (X, \$<<)$
$\bullet\ FunctRespects\ G\ (X, \$<<) \wedge WellFounded\ (X, \$<<)$
$\Rightarrow (\forall\ x$
$\bullet\ x \in X \Rightarrow UniquePartFixp\ (TcUpTo\ (X, \$<<)\ x)\ G)$

recursion_theorem
$\vdash \forall\ (X, \$<<)\ G$
$\bullet\ FunctRespects\ G\ (X, \$<<) \wedge WellFounded\ (X, \$<<)$
$\Rightarrow UniquePartFixp\ X\ G$

tf_recursion_thm
$\vdash \forall\ \$<<\ G$
$\bullet\ FunctRespects\ G\ (Universe, \$<<)$
$\wedge\ WellFounded\ (Universe, \$<<)$
$\Rightarrow (\exists_1\ f \bullet\ G\ f = f)$

⟨◁_fc_thm $\vdash \forall\ y\ x\ \$<<\ f \bullet\ y << x \Rightarrow ((x, \$<<)\ \langle\lhd\ f)\ y = f\ y$

tf_rec_thm2 $\vdash \forall\ \$<<$
$\bullet\ WellFounded\ (Universe, \$<<)$
$\Rightarrow (\forall\ af \bullet\ \exists\ f \bullet\ \forall\ x \bullet\ f\ x = af\ ((x, \$<<)\ \langle\lhd\ f)\ x)$

B The Theory U_orders

B.1 Parents

ordered_sets

B.2 Children

fixp   *gst−ax*

B.3 Constants

UWellFounded	$('a \to 'a \to BOOL) \to BOOL$
UWellOrdering	$('a \to 'a \to BOOL) \to BOOL$
UMinCond	$('a \to 'a \to BOOL) \to BOOL$
UWeakMinCond	$('a \to 'a \to BOOL) \to BOOL$
ULinearOrder	$('a \to 'a \to BOOL) \to BOOL$
UPartialOrder	$('a \to 'a \to BOOL) \to BOOL$
URefl	$('a \to 'a \to BOOL) \to BOOL$
UComplete	$('a \to 'a \to BOOL) \to BOOL$
UStrictLinearOrder	$('a \to 'a \to BOOL) \to BOOL$
UTrich	$('a \to 'a \to BOOL) \to BOOL$
UStrictPartialOrder	$('a \to 'a \to BOOL) \to BOOL$
UTrans	$('a \to 'a \to BOOL) \to BOOL$
UAntisym	$('a \to 'a \to BOOL) \to BOOL$
UIrrefl	$('a \to 'a \to BOOL) \to BOOL$

B.4 Definitions

UIrrefl
UAntisym
UTrans
UStrictPartialOrder
UTrich
UStrictLinearOrder
UComplete
URefl
UPartialOrder
ULinearOrder
UWeakMinCond
UMinCond
UWellOrdering
UWellFounded

$$\vdash UIrrefl = Curry\ Irrefl\ Universe$$
$$\land\ UAntisym = Curry\ Antisym\ Universe$$
$$\land\ UTrans = Curry\ Trans\ Universe$$

```
∧ UStrictPartialOrder
  = Curry StrictPartialOrder Universe
∧ UTrich = Curry Trich Universe
∧ UStrictLinearOrder
  = Curry StrictLinearOrder Universe
∧ UComplete = Curry Complete Universe
∧ URefl = Curry Refl Universe
∧ UPartialOrder = Curry PartialOrder Universe
∧ ULinearOrder = Curry LinearOrder Universe
∧ UWeakMinCond = Curry WeakMinCond Universe
∧ UMinCond = Curry MinCond Universe
∧ UWellOrdering = Curry WellOrdering Universe
∧ UWellFounded = Curry WellFounded Universe
```

B.5 Theorems

u_irrefl_def_thm

```
⊢ ∀ $<<• UIrrefl $<< ⇔ (∀ x• ¬ x << x)
```

u_antisym_def_thm

```
⊢ ∀ $<<
  • UAntisym $<<
      ⇔ (∀ x y• ¬ x = y ⇒ ¬ (x << y ∧ y << x))
```

u_trans_def_thm

```
⊢ ∀ $<<
  • UTrans $<< ⇔ (∀ x y z• x << y ∧ y << z ⇒ x << z)
```

u_strict_partial_order_def_thm

```
⊢ ∀ $<<
  • UStrictPartialOrder $<<
      ⇔ UIrrefl $<< ∧ UAntisym $<< ∧ UTrans $<<
```

u_trich_def_thm

```
⊢ ∀ $<<
  • UTrich $<< ⇔ (∀ x y• ¬ x = y ⇒ x << y ∨ y << x)
```

u_strict_linear_order_def_thm

```
⊢ ∀ $<<
  • UStrictLinearOrder $<<
      ⇔ UStrictPartialOrder $<< ∧ UTrich $<<
```

u_complete_def_thm

```
⊢ ∀ $<<
  • UComplete $<<
      ⇔ (∀ A x
      • ¬ A = {}
            ∧ UnboundedAbove (A, $<<)
            ∧ UpperBound (A, $<<, x)
          ⇒ (∃ y• A HasSupremum (y, Universe, $<<)))
```

u_refl_def_thm

```
⊢ ∀ $<<• URefl $<< ⇔ (∀ x• x << x)
```

u_partial_order_def_thm

```
⊢ ∀ $<<• UPartialOrder $<< ⇔ UAntisym $<< ∧ UTrans $<<
```

u_linear_order_def_thm

```
⊢ ∀ $<<
  • ULinearOrder $<< ⇔ UPartialOrder $<< ∧ UTrich $<<
```

u_weak_min_cond_def_thm

```
⊢ ∀ $<<
  • UWeakMinCond $<<
      ⇔ (∀ A
      • ¬ A = {}
          ⇒ (∃ x
          • x ∈ A
              ∧ (∀ y
              • y ∈ A ∧ y << x ∧ ¬ y = x ⇒ x << y)))
```

u_min_cond_def_thm

```
⊢ ∀ $<<
  • UMinCond $<<
      ⇔ (∀ A
      • ¬ A = {}
          ⇒ (∃ x
          • x ∈ A ∧ (∀ y• y ∈ A ∧ ¬ y = x ⇒ ¬ y << x)))
```

u_well_ordering_def_thm

```
⊢ ∀ $<<
  • UWellOrdering $<< ⇔ ULinearOrder $<< ∧ UMinCond $<<
```

u_well_founded_def_thm

```
⊢ ∀ $<<
  • UWellFounded $<<
      ⇔ (∀ A
      • ¬ A = {}
          ⇒ (∃ x• x ∈ A ∧ (∀ y• y ∈ A ⇒ ¬ y << x)))
```

u_zorn_thm2

```
⊢ ∀ $<<
  • UTrans $<<
        ∧ UAntisym $<<
        ∧ (∀ C
        • Trich (C, $<<)
            ⇒ (∃ x• UpperBound (C, $<<, x)))
      ⇒ (∃ x• ∀ y• ¬ y = x ⇒ ¬ x << y)
```

u_well_ordering_thm

```
⊢ ∃ $<<• UWellOrdering $<<
```

u_min_cond_descending_sequence_thm

```
⊢ ∀ $<<
  • UMinCond $<<
      ⇔ (∀ f
      • (∀ n• f (n + 1) << f n)
          ⇒ (∃ m• ∀ n• m < n ⇒ f n = f m))
```

u_well_founded_descending_sequence_thm

```
⊢ ∀ $<<
  • UWellFounded $<< ⇔ ¬ (∃ f• ∀ n• f (n + 1) << f n)
```

u_well_founded_induction_thm

```
⊢ ∀ $<<
  • UWellFounded $<<
      ⇔ (∀ p
      • (∀ x• (∀ y• y << x ⇒ p y) ⇒ p x) ⇒ (∀ x• p x))
```

u_refl_well_ordering_lower_bounds_thm

```
⊢ ∀ $<<
  • URefl $<< ∧ UWellOrdering $<<
```

```
                    ⇔ (∀ A
                    • ¬ A = {}
                        ⇒ (∃₁ x• x ∈ A ∧ (∀ y• y ∈ A ⇒ x << y)))
weak_min_cond_⩽_lemma
                ⊢ ∀ X $<<
                  • (∀ i j• i << j ⇒ i ⩽ j) ⇒ WeakMinCond (X, $<<)
⩽_less_lemma    ⊢ Refl (Universe, $⩽)
                    ∧ Irrefl (Universe, $<)
                    ∧ ¬ Irrefl (Universe, $⩽)
                    ∧ ¬ Refl (Universe, $<)
                    ∧ Antisym (Universe, $⩽)
                    ∧ Antisym (Universe, $<)
                    ∧ Trans (Universe, $⩽)
                    ∧ Trans (Universe, $<)
                    ∧ Trich (Universe, $⩽)
                    ∧ Trich (Universe, $<)
                    ∧ WeakMinCond (Universe, $⩽)
                    ∧ WeakMinCond (Universe, $<)
independence_lemma1
                ⊢ ∀ m n
                  • m ∈ {0; 1; 2; 3; 4} ∧ n ∈ {0; 1; 2; 3; 4} ∧ ¬ m = n
                    ⇒ (let props m
                          = (if m = 0
                            then Refl
                            else if m = 1
                            then Antisym
                            else if m = 2
                            then Trans
                            else if m = 3
                            then Trich
                            else WeakMinCond)
                      in ∃ X $<<
                      • props m (X, $<<) ∧ ¬ props n (X, $<<))
independence_lemma2
                ⊢ ∀ m
                  • m ∈ {1; 2; 3; 4}
                    ⇒ (let props m
                          = (if m = 1
                            then Antisym
                            else if m = 2
                            then Trans
                            else if m = 3
                            then Trich
                            else WeakMinCond)
                      in (∃ X $<<
                        • Refl (X, $<<) ∧ ¬ props m (X, $<<))
                        ⇒ (∃ X $<<
                        • Irrefl (X, $<<) ∧ ¬ props m (X, $<<)))
independence_lemma3
                ⊢ ∀ m
                  • m ∈ {1; 2; 3; 4}
```

```
        ⇒ (let props m
              = (if m = 1
                then Antisym
                else if m = 2
                then Trans
                else if m = 3
                then Trich
                else WeakMinCond)
          in (∃ X $<<
            • props m (X, $<<) ∧ ¬ Refl (X, $<<))
            ⇒ (∃ X $<<
            • props m (X, $<<) ∧ ¬ Irrefl (X, $<<)))
```

independence_thm

```
    ⊢ ∀ m n
      • m ∈ {0; 1; 2; 3; 4; 5} ∧ n ∈ {0; 1; 2; 3; 4; 5}
        ⇒ (let props m
              = (if m = 0
                then Refl
                else if m = 1
                then Antisym
                else if m = 2
                then Trans
                else if m = 3
                then Trich
                else if m = 4
                then WeakMinCond
                else Irrefl)
          in (¬ m = n
              ⇒ (∃ X $<<
              • props m (X, $<<) ∧ ¬ props n (X, $<<)))
            ∧ (m < n
              ⇒ (∃ X $<<
              • props m (X, $<<) ∧ props n (X, $<<))))
```

independence_clauses

```
    ⊢ (∃ X $<<• Antisym (X, $<<) ∧ Irrefl (X, $<<))
        ∧ (∃ X $<<• Antisym (X, $<<) ∧ ¬ Irrefl (X, $<<))
        ∧ (∃ X $<<• Antisym (X, $<<) ∧ ¬ Refl (X, $<<))
        ∧ (∃ X $<<• Antisym (X, $<<) ∧ Trans (X, $<<))
        ∧ (∃ X $<<• Antisym (X, $<<) ∧ ¬ Trans (X, $<<))
        ∧ (∃ X $<<• Antisym (X, $<<) ∧ Trich (X, $<<))
        ∧ (∃ X $<<• Antisym (X, $<<) ∧ ¬ Trich (X, $<<))
        ∧ (∃ X $<<
        • Antisym (X, $<<) ∧ WeakMinCond (X, $<<))
        ∧ (∃ X $<<
        • Antisym (X, $<<) ∧ ¬ WeakMinCond (X, $<<))
        ∧ (∃ X $<<• Irrefl (X, $<<) ∧ ¬ Antisym (X, $<<))
        ∧ (∃ X $<<• Irrefl (X, $<<) ∧ ¬ Refl (X, $<<))
        ∧ (∃ X $<<• Irrefl (X, $<<) ∧ ¬ Trans (X, $<<))
        ∧ (∃ X $<<• Irrefl (X, $<<) ∧ ¬ Trich (X, $<<))
        ∧ (∃ X $<<
        • Irrefl (X, $<<) ∧ ¬ WeakMinCond (X, $<<))
```

∧ (∃ X $<<• *Refl* (X, $<<) ∧ *Antisym* (X, $<<))
∧ (∃ X $<<• *Refl* (X, $<<) ∧ ¬ *Antisym* (X, $<<))
∧ (∃ X $<<• *Refl* (X, $<<) ∧ *Irrefl* (X, $<<))
∧ (∃ X $<<• *Refl* (X, $<<) ∧ ¬ *Irrefl* (X, $<<))
∧ (∃ X $<<• *Refl* (X, $<<) ∧ *Trans* (X, $<<))
∧ (∃ X $<<• *Refl* (X, $<<) ∧ ¬ *Trans* (X, $<<))
∧ (∃ X $<<• *Refl* (X, $<<) ∧ *Trich* (X, $<<))
∧ (∃ X $<<• *Refl* (X, $<<) ∧ ¬ *Trich* (X, $<<))
∧ (∃ X $<<• *Refl* (X, $<<) ∧ *WeakMinCond* (X, $<<))
∧ (∃ X $<<• *Refl* (X, $<<) ∧ ¬ *WeakMinCond* (X, $<<))
∧ (∃ X $<<• *Trans* (X, $<<) ∧ ¬ *Antisym* (X, $<<))
∧ (∃ X $<<• *Trans* (X, $<<) ∧ *Irrefl* (X, $<<))
∧ (∃ X $<<• *Trans* (X, $<<) ∧ ¬ *Irrefl* (X, $<<))
∧ (∃ X $<<• *Trans* (X, $<<) ∧ ¬ *Refl* (X, $<<))
∧ (∃ X $<<• *Trans* (X, $<<) ∧ *Trich* (X, $<<))
∧ (∃ X $<<• *Trans* (X, $<<) ∧ ¬ *Trich* (X, $<<))
∧ (∃ X $<<• *Trans* (X, $<<) ∧ *WeakMinCond* (X, $<<))
∧ (∃ X $<<
• *Trans* (X, $<<) ∧ ¬ *WeakMinCond* (X, $<<))
∧ (∃ X $<<• *Trich* (X, $<<) ∧ ¬ *Antisym* (X, $<<))
∧ (∃ X $<<• *Trich* (X, $<<) ∧ *Irrefl* (X, $<<))
∧ (∃ X $<<• *Trich* (X, $<<) ∧ ¬ *Irrefl* (X, $<<))
∧ (∃ X $<<• *Trich* (X, $<<) ∧ ¬ *Refl* (X, $<<))
∧ (∃ X $<<• *Trich* (X, $<<) ∧ ¬ *Trans* (X, $<<))
∧ (∃ X $<<• *Trich* (X, $<<) ∧ *WeakMinCond* (X, $<<))
∧ (∃ X $<<
• *Trich* (X, $<<) ∧ ¬ *WeakMinCond* (X, $<<))
∧ (∃ X $<<
• *WeakMinCond* (X, $<<) ∧ ¬ *Antisym* (X, $<<))
∧ (∃ X $<<• *WeakMinCond* (X, $<<) ∧ *Irrefl* (X, $<<))
∧ (∃ X $<<
• *WeakMinCond* (X, $<<) ∧ ¬ *Irrefl* (X, $<<))
∧ (∃ X $<<• *WeakMinCond* (X, $<<) ∧ ¬ *Refl* (X, $<<))
∧ (∃ X $<<
• *WeakMinCond* (X, $<<) ∧ ¬ *Trans* (X, $<<))
∧ (∃ X $<<
• *WeakMinCond* (X, $<<) ∧ ¬ *Trich* (X, $<<))

C INDEX