Equivalents of the Axiom of Choice

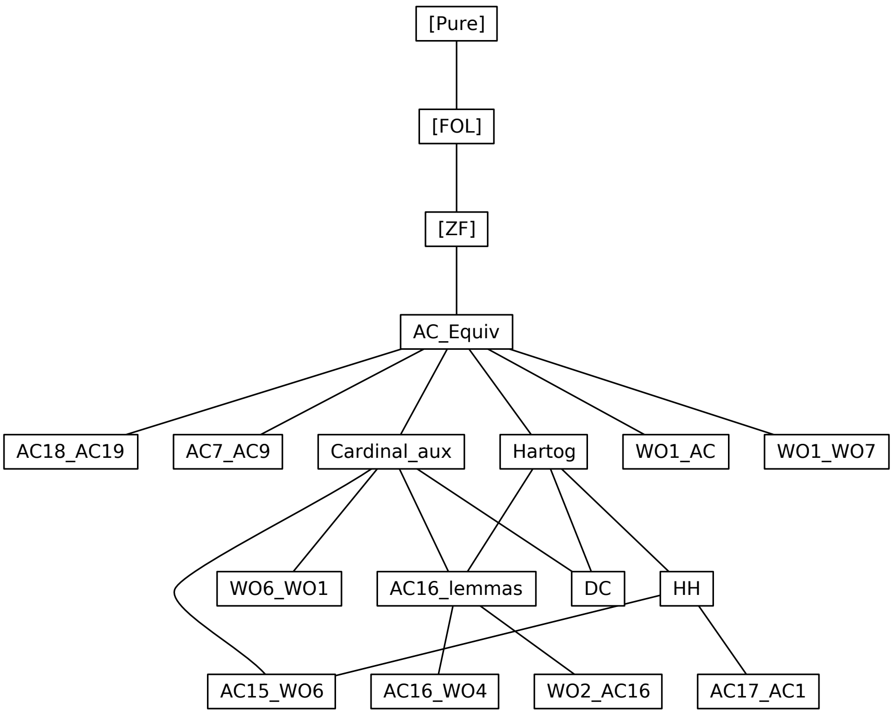
Krzysztof Grąbczewski

March 13, 2025

Abstract

This development [1] proves the equivalence of seven formulations of the well-ordering theorem and twenty formulations of the axiom of choice. It formalizes the first two chapters of the monograph *Equivalents of the Axiom of Choice* by Rubin and Rubin [2]. Some of this mmaterial involves extremely complex techniques.

Contents



```
theory AC_Equiv
imports ZF
begin



definition
    "WO1 ≡ ∀A. ∃R. well_ord(A,R)"

definition
    "WO2 ≡ ∀A. ∃a. Ord(a) ∧ A≈a"

definition
    "WO3 ≡ ∀A. ∃a. Ord(a) ∧ (∃b. b ⊆ a ∧ A≈b)"

definition
    "WO4(m) ≡ ∀A. ∃a f. Ord(a) ∧ domain(f)=a ∧
                         (⋃b<a. f`b) = A ∧ (∀b<a. f`b ≲ m)"

definition
    "WO5 ≡ ∃m ∈ nat. 1≤m ∧ WO4(m)"

definition
    "WO6 ≡ ∀A. ∃m ∈ nat. 1≤m ∧ (∃a f. Ord(a) ∧ domain(f)=a
                              ∧ (⋃b<a. f`b) = A ∧ (∀b<a. f`b ≲ m))"

definition
    "WO7 ≡ ∀A. Finite(A) ⟷ (∀R. well_ord(A,R) ⟶ well_ord(A,converse(R)))"

definition
    "WO8 ≡ ∀A. (∃f. f ∈ (∏X ∈ A. X)) ⟶ (∃R. well_ord(A,R))"


definition

  pairwise_disjoint :: "i ⇒ o"  where
    "pairwise_disjoint(A) ≡ ∀A1 ∈ A. ∀A2 ∈ A. A1 ∩ A2 ≠ 0 ⟶ A1=A2"

definition
  sets_of_size_between :: "[i, i, i] ⇒ o"  where
    "sets_of_size_between(A,m,n) ≡ ∀B ∈ A. m ≲ B ∧ B ≲ n"



definition
    "AC0 ≡ ∀A. ∃f. f ∈ (∏X ∈ Pow(A)-{0}. X)"
```

definition
```
"AC1 ≡ ∀A. 0∉A ⟶ (∃f. f ∈ (∏X ∈ A. X))"
```

definition
```
"AC2 ≡ ∀A. 0∉A ∧ pairwise_disjoint(A)
           ⟶ (∃C. ∀B ∈ A. ∃y. B ∩ C = {y})"
```

definition
```
"AC3 ≡ ∀A B. ∀f ∈ A->B. ∃g. g ∈ (∏x ∈ {a ∈ A. f`a≠0}. f`x)"
```

definition
```
"AC4 ≡ ∀R A B. (R ⊆ A*B ⟶ (∃f. f ∈ (∏x ∈ domain(R). R``{x})))"
```

definition
```
"AC5 ≡ ∀A B. ∀f ∈ A->B. ∃g ∈ range(f)->A. ∀x ∈ domain(g). f`(g`x)
= x"
```

definition
```
"AC6 ≡ ∀A. 0∉A ⟶ (∏B ∈ A. B)≠0"
```

definition
```
"AC7 ≡ ∀A. 0∉A ∧ (∀B1 ∈ A. ∀B2 ∈ A. B1≈B2) ⟶ (∏B ∈ A. B) ≠
0"
```

definition
```
"AC8 ≡ ∀A. (∀B ∈ A. ∃B1 B2. B=⟨B1,B2⟩ ∧ B1≈B2)
           ⟶ (∃f. ∀B ∈ A. f`B ∈ bij(fst(B),snd(B)))"
```

definition
```
"AC9 ≡ ∀A. (∀B1 ∈ A. ∀B2 ∈ A. B1≈B2) ⟶
           (∃f. ∀B1 ∈ A. ∀B2 ∈ A. f`⟨B1,B2⟩ ∈ bij(B1,B2))"
```

definition
```
"AC10(n) ≡  ∀A. (∀B ∈ A. ¬Finite(B)) ⟶
           (∃f. ∀B ∈ A. (pairwise_disjoint(f`B) ∧
           sets_of_size_between(f`B, 2, succ(n)) ∧ ⋃(f`B)=B))"
```

definition
```
"AC11 ≡ ∃n ∈ nat. 1≤n ∧ AC10(n)"
```

definition
```
"AC12 ≡ ∀A. (∀B ∈ A. ¬Finite(B)) ⟶
         (∃n ∈ nat. 1≤n ∧ (∃f. ∀B ∈ A. (pairwise_disjoint(f`B)
∧
              sets_of_size_between(f`B, 2, succ(n)) ∧ ⋃(f`B)=B)))"
```

definition
```
"AC13(m) ≡ ∀A. 0∉A ⟶ (∃f. ∀B ∈ A. f`B≠0 ∧ f`B ⊆ B ∧ f`B ≲
m)"
```

definition
```
    "AC14 ≡ ∃m ∈ nat. 1≤m ∧ AC13(m)"
```

definition
```
    "AC15 ≡ ∀A. 0∉A ⟶
                (∃m ∈ nat. 1≤m ∧ (∃f. ∀B ∈ A. f`B≠0 ∧ f`B ⊆ B ∧
f`B ≲ m))"
```

definition
```
    "AC16(n, k)  ≡
      ∀A. ¬Finite(A) ⟶
          (∃T. T ⊆ {X ∈ Pow(A). X≈succ(n)} ∧
          (∀X ∈ {X ∈ Pow(A). X≈succ(k)}. ∃! Y. Y ∈ T ∧ X ⊆ Y))"
```

definition
```
    "AC17 ≡ ∀A. ∀g ∈ (Pow(A)-{0} -> A) -> Pow(A)-{0}.
                  ∃f ∈ Pow(A)-{0} -> A. f`(g`f) ∈ g`f"
```

locale *AC18* =
```
  assumes AC18: "A≠0 ∧ (∀a ∈ A. B(a) ≠ 0) ⟶
    ((⋂a ∈ A. ⋃b ∈ B(a). X(a,b)) =
      (⋃f ∈ ∏a ∈ A. B(a). ⋂a ∈ A. X(a, f`a)))"
```
— AC18 cannot be expressed within the object-logic

definition
```
    "AC19 ≡ ∀A. A≠0 ∧ 0∉A ⟶ ((⋂a ∈ A. ⋃b ∈ a. b) =
                  (⋃f ∈ (∏B ∈ A. B). ⋂a ∈ A. f`a))"
```

lemma *rvimage_id: "rvimage(A,id(A),r) = r ∩ A*A"*
⟨*proof*⟩

lemma *ordertype_Int:*
```
     "well_ord(A,r) ⟹ ordertype(A, r ∩ A*A) = ordertype(A,r)"
```
⟨*proof*⟩

lemma *lam_sing_bij: "(λx ∈ A. {x}) ∈ bij(A, {{x}. x ∈ A})"*
⟨*proof*⟩

lemma *inj_strengthen_type:*
```
     "⟦f ∈ inj(A, B);  ⋀a. a ∈ A ⟹ f`a ∈ C⟧ ⟹ f ∈ inj(A,C)"
```
⟨*proof*⟩

lemma *ex1_two_eq: "⟦∃! x. P(x); P(x); P(y)⟧ ⟹ x=y"*
⟨*proof*⟩

lemma *first_in_B:*
"⟦well_ord(⋃(A),r); 0 ∉ A; B ∈ A⟧ ⟹ (THE b. first(b,B,r)) ∈ B"
⟨*proof*⟩

lemma *ex_choice_fun: "⟦well_ord(⋃(A), R); 0 ∉ A⟧ ⟹ ∃f. f ∈ (∏X ∈ A. X)"*
⟨*proof*⟩

lemma *ex_choice_fun_Pow: "well_ord(A, R) ⟹ ∃f. f ∈ (∏X ∈ Pow(A)-{0}. X)"*
⟨*proof*⟩

lemma *lepoll_m_imp_domain_lepoll_m:*
"⟦m ∈ nat; u ≲ m⟧ ⟹ domain(u) ≲ m"
⟨*proof*⟩

lemma *rel_domain_ex1:*
"⟦succ(m) ≲ domain(r); r ≲ succ(m); m ∈ nat⟧ ⟹ function(r)"
⟨*proof*⟩

lemma *rel_is_fun:*
"⟦succ(m) ≲ domain(r); r ≲ succ(m); m ∈ nat;
*r ⊆ A*B; A=domain(r)⟧ ⟹ r ∈ A->B"*
⟨*proof*⟩

end

theory *Cardinal_aux* **imports** *AC_Equiv* **begin**

lemma *Diff_lepoll*: "⟦A ≲ succ(m); B ⊆ A; B≠0⟧ ⟹ A-B ≲ m"
⟨*proof*⟩

lemma *lepoll_imp_ex_le_eqpoll*:
"⟦A ≲ i; Ord(i)⟧ ⟹ ∃j. j ≤ i ∧ A ≈ j"
⟨*proof*⟩

lemma *lesspoll_imp_ex_lt_eqpoll*:
"⟦A ≺ i; Ord(i)⟧ ⟹ ∃j. j<i ∧ A ≈ j"
⟨*proof*⟩

lemma *Un_eqpoll_Inf_Ord*:
assumes *A*: "A ≈ i" **and** *B*: "B ≈ i" **and** *NFI*: "¬ Finite(i)" **and** *i*: "Ord(i)"
shows "A ∪ B ≈ i"
⟨*proof*⟩

schematic_goal *paired_bij*: "?f ∈ bij({{y,z}. y ∈ x}, x)"
⟨*proof*⟩

lemma *paired_eqpoll*: "{{y,z}. y ∈ x} ≈ x"
⟨*proof*⟩

lemma *ex_eqpoll_disjoint*: "∃B. B ≈ A ∧ B ∩ C = 0"
⟨*proof*⟩

lemma *Un_lepoll_Inf_Ord*:
"⟦A ≲ i; B ≲ i; ¬Finite(i); Ord(i)⟧ ⟹ A ∪ B ≲ i"
⟨*proof*⟩

lemma *Least_in_Ord*: "⟦P(i); i ∈ j; Ord(j)⟧ ⟹ (μ i. P(i)) ∈ j"
⟨*proof*⟩

lemma *Diff_first_lepoll*:
"⟦well_ord(x,r); y ⊆ x; y ≲ succ(n); n ∈ nat⟧
⟹ y - {THE b. first(b,y,r)} ≲ n"
⟨*proof*⟩

lemma *UN_subset_split:*
"(⋃x ∈ X. P(x)) ⊆ (⋃x ∈ X. P(x)-Q(x)) ∪ (⋃x ∈ X. Q(x))"
⟨*proof*⟩

lemma *UN_sing_lepoll:* "Ord(a) ⟹ (⋃x ∈ a. {P(x)}) ≲ a"
⟨*proof*⟩

lemma *UN_fun_lepoll_lemma [rule_format]:*
"⟦well_ord(T, R); ¬Finite(a); Ord(a); n ∈ nat⟧
⟹ ∀f. (∀b ∈ a. f‘b ≲ n ∧ f‘b ⊆ T) ⟶ (⋃b ∈ a. f‘b) ≲ a"
⟨*proof*⟩

lemma *UN_fun_lepoll:*
"⟦∀b ∈ a. f‘b ≲ n ∧ f‘b ⊆ T; well_ord(T, R);
¬Finite(a); Ord(a); n ∈ nat⟧ ⟹ (⋃b ∈ a. f‘b) ≲ a"
⟨*proof*⟩

lemma *UN_lepoll:*
"⟦∀b ∈ a. F(b) ≲ n ∧ F(b) ⊆ T; well_ord(T, R);
¬Finite(a); Ord(a); n ∈ nat⟧
⟹ (⋃b ∈ a. F(b)) ≲ a"
⟨*proof*⟩

lemma *UN_eq_UN_Diffs:*
"Ord(a) ⟹ (⋃b ∈ a. F(b)) = (⋃b ∈ a. F(b) - (⋃c ∈ b. F(c)))"
⟨*proof*⟩

lemma *lepoll_imp_eqpoll_subset:*
"a ≲ X ⟹ ∃Y. Y ⊆ X ∧ a ≈ Y"
⟨*proof*⟩

lemma *Diff_lesspoll_eqpoll_Card_lemma:*
"⟦A≈a; ¬Finite(a); Card(a); B ≺ a; A-B ≺ a⟧ ⟹ P"
⟨*proof*⟩

lemma *Diff_lesspoll_eqpoll_Card:*
"⟦A ≈ a; ¬Finite(a); Card(a); B ≺ a⟧ ⟹ A - B ≈ a"
⟨*proof*⟩

end

theory *WO6_WO1*
imports *Cardinal_aux*
begin

definition
```
NN :: "i ⇒ i" where
  "NN(y) ≡ {m ∈ nat. ∃a. ∃f. Ord(a) ∧ domain(f)=a ∧
                  (⋃b<a. f`b) = y ∧ (∀b<a. f`b ≲ m)}"
```

definition
```
uu :: "[i, i, i, i] ⇒ i" where
  "uu(f, beta, gamma, delta) ≡ (f`beta * f`gamma) ∩ f`delta"
```

definition
```
vv1 :: "[i, i, i] ⇒ i" where
   "vv1(f,m,b) ≡
        let g = μ g. (∃d. Ord(d) ∧ (domain(uu(f,b,g,d)) ≠ 0 ∧
                                 domain(uu(f,b,g,d)) ≲ m));
            d = μ d. domain(uu(f,b,g,d)) ≠ 0 ∧
                         domain(uu(f,b,g,d)) ≲ m
        in  if f`b ≠ 0 then domain(uu(f,b,g,d)) else 0"
```

definition
```
ww1 :: "[i, i, i] ⇒ i" where
   "ww1(f,m,b) ≡ f`b - vv1(f,m,b)"
```

definition
```
gg1 :: "[i, i, i] ⇒ i" where
   "gg1(f,a,m) ≡ λb ∈ a++a. if b<a then vv1(f,m,b) else ww1(f,m,b--a)"
```

definition
```
vv2 :: "[i, i, i, i] ⇒ i" where
   "vv2(f,b,g,s) ≡
        if f`g ≠ 0 then {uu(f, b, g, μ d. uu(f,b,g,d) ≠ 0)`s} else
0"
```

definition
```
ww2 :: "[i, i, i, i] ⇒ i" where
   "ww2(f,b,g,s) ≡ f`g - vv2(f,b,g,s)"
```

definition
```
gg2 :: "[i, i, i, i] ⇒ i" where
   "gg2(f,a,b,s) ≡
          λg ∈ a++a. if g<a then vv2(f,b,g,s) else ww2(f,b,g--a,s)"
```

lemma `WO2_WO3: "WO2 ⟹ WO3"`

⟨proof⟩

lemma *WO3_WO1: "WO3 ⟹ WO1"*
⟨proof⟩

lemma *WO1_WO2: "WO1 ⟹ WO2"*
⟨proof⟩

lemma *lam_sets: "f ∈ A->B ⟹ (λx ∈ A. {f‘x}): A -> {{b}. b ∈ B}"*
⟨proof⟩

lemma *surj_imp_eq': "f ∈ surj(A,B) ⟹ (⋃a ∈ A. {f‘a}) = B"*
⟨proof⟩

lemma *surj_imp_eq: "⟦f ∈ surj(A,B); Ord(A)⟧ ⟹ (⋃a<A. {f‘a}) = B"*
⟨proof⟩

lemma *WO1_WO4: "WO1 ⟹ WO4(1)"*
⟨proof⟩

lemma *WO4_mono: "⟦m≤n; WO4(m)⟧ ⟹ WO4(n)"*
⟨proof⟩

lemma *WO4_WO5: "⟦m ∈ nat; 1≤m; WO4(m)⟧ ⟹ WO5"*
⟨proof⟩

lemma *WO5_WO6: "WO5 ⟹ WO6"*
⟨proof⟩

lemma *lt_oadd_odiff_disj:*
```
      "⟦k < i++j;  Ord(i);  Ord(j)⟧
       ⟹ k < i  |  (¬ k<i ∧ k = i ++ (k--i) ∧ (k--i)<j)"
```
⟨proof⟩

```
lemma domain_uu_subset: "domain(uu(f,b,g,d)) ⊆ f‘b"
⟨proof⟩

lemma quant_domain_uu_lepoll_m:
     "∀b<a. f‘b ≲ m ⟹ ∀b<a. ∀g<a. ∀d<a. domain(uu(f,b,g,d)) ≲ m"
⟨proof⟩

lemma uu_subset1: "uu(f,b,g,d) ⊆ f‘b * f‘g"
⟨proof⟩

lemma uu_subset2: "uu(f,b,g,d) ⊆ f‘d"
⟨proof⟩

lemma uu_lepoll_m: "⟦∀b<a. f‘b ≲ m;  d<a⟧ ⟹ uu(f,b,g,d) ≲ m"
⟨proof⟩
```

```
lemma cases:
     "∀b<a. ∀g<a. ∀d<a. u(f,b,g,d) ≲ m
      ⟹ (∀b<a. f‘b ≠ 0 ⟶
                 (∃g<a. ∃d<a. u(f,b,g,d) ≠ 0 ∧ u(f,b,g,d) ≺ m))
        | (∃b<a. f‘b ≠ 0 ∧ (∀g<a. ∀d<a. u(f,b,g,d) ≠ 0 ⟶
                                       u(f,b,g,d) ≈ m))"
  ⟨proof⟩
```

```
lemma UN_oadd: "Ord(a) ⟹ (⋃b<a++a. C(b)) = (⋃b<a. C(b) ∪ C(a++b))"
⟨proof⟩
```

```
lemma vv1_subset: "vv1(f,m,b) ⊆ f‘b"
⟨proof⟩
```

lemma *UN_gg1_eq:*
  "⟦Ord(a); m ∈ nat⟧ ⟹ (⋃b<a++a. gg1(f,a,m)`b) = (⋃b<a. f`b)"
⟨*proof*⟩

lemma *domain_gg1: "domain(gg1(f,a,m)) = a++a"*
⟨*proof*⟩

lemma *nested_LeastI:*
    "⟦P(a, b); Ord(a); Ord(b);
        Least_a = (μ a. ∃x. Ord(x) ∧ P(a, x))⟧
     ⟹ P(Least_a, μ b. P(Least_a, b))"
⟨*proof*⟩

lemmas *nested_Least_instance =*
      nested_LeastI [of "λg d. domain(uu(f,b,g,d)) ≠ 0 ∧
                              domain(uu(f,b,g,d)) ≲ m"] **for** *f b m*

lemma *gg1_lepoll_m:*
    "⟦Ord(a); m ∈ nat;
        ∀b<a. f`b ≠0 ⟶
            (∃g<a. ∃d<a. domain(uu(f,b,g,d)) ≠ 0 ∧
                        domain(uu(f,b,g,d)) ≲ m);
        ∀b<a. f`b ≲ succ(m); b<a++a⟧
     ⟹ gg1(f,a,m)`b ≲ m"
⟨*proof*⟩

lemma *ex_d_uu_not_empty:*
    "⟦b<a; g<a; f`b≠0; f`g≠0;
        *y*y ⊆ y; (⋃b<a. f`b)=y⟧*
     ⟹ ∃d<a. uu(f,b,g,d) ≠ 0"
⟨*proof*⟩

lemma *uu_not_empty:*
    *"⟦b<a; g<a; f`b≠0; f`g≠0; y*y ⊆ y; (⋃b<a. f`b)=y⟧*

```
     ⟹ uu(f,b,g,μ d. (uu(f,b,g,d) ≠ 0)) ≠ 0"
⟨proof⟩

lemma not_empty_rel_imp_domain: "⟦r ⊆ A*B; r≠0⟧ ⟹ domain(r)≠0"
⟨proof⟩

lemma Least_uu_not_empty_lt_a:
    "⟦b<a; g<a; f`b≠0; f`g≠0; y*y ⊆ y; (⋃b<a. f`b)=y⟧
     ⟹ (μ d. uu(f,b,g,d) ≠ 0) < a"
⟨proof⟩

lemma subset_Diff_sing: "⟦B ⊆ A; a∉B⟧ ⟹ B ⊆ A-{a}"
⟨proof⟩


lemma supset_lepoll_imp_eq:
    "⟦A ≲ m; m ≲ B; B ⊆ A; m ∈ nat⟧ ⟹ A=B"
⟨proof⟩

lemma uu_Least_is_fun:
    "⟦∀g<a. ∀d<a. domain(uu(f, b, g, d))≠0 ⟶
         domain(uu(f, b, g, d)) ≈ succ(m);
         ∀b<a. f`b ≲ succ(m);  y*y ⊆ y;
         (⋃b<a. f`b)=y;  b<a;  g<a;  d<a;
         f`b≠0;  f`g≠0;  m ∈ nat;  s ∈ f`b⟧
     ⟹ uu(f, b, g, μ d. uu(f,b,g,d)≠0) ∈ f`b -> f`g"
⟨proof⟩

lemma vv2_subset:
    "⟦∀g<a. ∀d<a. domain(uu(f, b, g, d))≠0 ⟶
                   domain(uu(f, b, g, d)) ≈ succ(m);
       ∀b<a. f`b ≲ succ(m); y*y ⊆ y;
       (⋃b<a. f`b)=y;  b<a;  g<a;  m ∈ nat;  s ∈ f`b⟧
     ⟹ vv2(f,b,g,s) ⊆ f`g"
⟨proof⟩




lemma UN_gg2_eq:
    "⟦∀g<a. ∀d<a. domain(uu(f,b,g,d)) ≠ 0 ⟶
             domain(uu(f,b,g,d)) ≈ succ(m);
       ∀b<a. f`b ≲ succ(m); y*y ⊆ y;
       (⋃b<a. f`b)=y;  Ord(a);  m ∈ nat;  s ∈ f`b;  b<a⟧
     ⟹ (⋃g<a++a. gg2(f,a,b,s) ` g) = y"
  ⟨proof⟩

lemma domain_gg2: "domain(gg2(f,a,b,s)) = a++a"
⟨proof⟩
```

lemma *vv2_lepoll:* "⟦m ∈ nat; m≠0⟧ ⟹ vv2(f,b,g,s) ≲ m"
⟨*proof*⟩

lemma *ww2_lepoll:*
"⟦∀b<a. f‘b ≲ succ(m); g<a; m ∈ nat; vv2(f,b,g,d) ⊆ f‘g⟧
⟹ ww2(f,b,g,d) ≲ m"
⟨*proof*⟩

lemma *gg2_lepoll_m:*
"⟦∀g<a. ∀d<a. domain(uu(f,b,g,d)) ≠ 0 ⟶
domain(uu(f,b,g,d)) ≈ succ(m);
∀b<a. f‘b ≲ succ(m); y*y ⊆ y;
(⋃b<a. f‘b)=y; b<a; s ∈ f‘b; m ∈ nat; m≠ 0; g<a++a⟧
⟹ gg2(f,a,b,s) ‘ g ≲ m"
⟨*proof*⟩

lemma *lemma_ii:* "⟦succ(m) ∈ NN(y); y*y ⊆ y; m ∈ nat; m≠0⟧ ⟹ m ∈ NN(y)"
⟨*proof*⟩

lemma *z_n_subset_z_succ_n:*
"∀n ∈ nat. rec(n, x, λk r. r ∪ r*r) ⊆ rec(succ(n), x, λk r. r ∪ r*r)"
⟨*proof*⟩

lemma *le_subsets:*
"⟦∀n ∈ nat. f(n)<=f(succ(n)); n≤m; n ∈ nat; m ∈ nat⟧
⟹ f(n)<=f(m)"
⟨*proof*⟩

lemma *le_imp_rec_subset:*

"⟦$n \le m$; $m \in nat$⟧
$\Longrightarrow$ rec(n, x, λk r. r $\cup$ r*r) $\subseteq$ rec(m, x, λk r. r $\cup$ r*r)"
⟨*proof*⟩

lemma *lemma_iv:* "$\exists$y. x $\cup$ y*y $\subseteq$ y"
⟨*proof*⟩

lemma *WO6_imp_NN_not_empty:* "WO6 $\Longrightarrow$ NN(y) $\neq$ 0"
⟨*proof*⟩

lemma *lemma1:*
"⟦($\bigcup$b<a. f`b)=y; x $\in$ y; $\forall$b<a. f`b $\lesssim$ 1; Ord(a)⟧ $\Longrightarrow$ $\exists$c<a. f`c = {x}"
⟨*proof*⟩

lemma *lemma2:*
"⟦($\bigcup$b<a. f`b)=y; x $\in$ y; $\forall$b<a. f`b $\lesssim$ 1; Ord(a)⟧
$\Longrightarrow$ f` ($\mu$ i. f`i = {x}) = {x}"
⟨*proof*⟩

lemma *NN_imp_ex_inj:* "1 $\in$ NN(y) $\Longrightarrow$ $\exists$a f. Ord(a) $\wedge$ f $\in$ inj(y, a)"
⟨*proof*⟩

lemma *y_well_ord:* "⟦y*y $\subseteq$ y; 1 $\in$ NN(y)⟧ $\Longrightarrow$ $\exists$r. well_ord(y, r)"
⟨*proof*⟩

```
lemma rev_induct_lemma [rule_format]:
     "⟦n ∈ nat; ⋀m. ⟦m ∈ nat; m≠0; P(succ(m))⟧ ⟹ P(m)⟧
      ⟹ n≠0 ⟶ P(n) ⟶ P(1)"
⟨proof⟩

lemma rev_induct:
     "⟦n ∈ nat;  P(n);  n≠0;
         ⋀m. ⟦m ∈ nat; m≠0; P(succ(m))⟧ ⟹ P(m)⟧
      ⟹ P(1)"
⟨proof⟩

lemma NN_into_nat: "n ∈ NN(y) ⟹ n ∈ nat"
⟨proof⟩

lemma lemma3: "⟦n ∈ NN(y); y*y ⊆ y; n≠0⟧ ⟹ 1 ∈ NN(y)"
⟨proof⟩

lemma NN_y_0: "0 ∈ NN(y) ⟹ y=0"
  ⟨proof⟩

lemma WO6_imp_WO1: "WO6 ⟹ WO1"
  ⟨proof⟩

end

theory WO1_WO7
imports AC_Equiv
begin

definition
    "LEMMA ≡
     ∀X. ¬Finite(X) ⟶ (∃R. well_ord(X,R) ∧ ¬well_ord(X,converse(R)))"

lemma WO7_iff_LEMMA: "WO7 ⟷ LEMMA"
  ⟨proof⟩
```

lemma *LEMMA_imp_WO1: "LEMMA ⟹ WO1"*
⟨*proof*⟩

lemma *converse_Memrel_not_wf_on:*
"⟦Ord(a); ¬Finite(a)⟧ ⟹ ¬wf[a](converse(Memrel(a)))"
⟨*proof*⟩

lemma *converse_Memrel_not_well_ord:*
"⟦Ord(a); ¬Finite(a)⟧ ⟹ ¬well_ord(a,converse(Memrel(a)))"
⟨*proof*⟩

lemma *well_ord_rvimage_ordertype:*
"well_ord(A,r) ⟹
rvimage (ordertype(A,r), converse(ordermap(A,r)),r) =
Memrel(ordertype(A,r))"
⟨*proof*⟩

lemma *well_ord_converse_Memrel:*
"⟦well_ord(A,r); well_ord(A,converse(r))⟧
⟹ well_ord(ordertype(A,r), converse(Memrel(ordertype(A,r))))"

⟨*proof*⟩

lemma *WO1_imp_LEMMA: "WO1 ⟹ LEMMA"*
⟨*proof*⟩

lemma *WO1_iff_WO7: "WO1 ⟷ WO7"*
⟨*proof*⟩

lemma *WO1_WO8: "WO1 ⟹ WO8"*
⟨*proof*⟩

lemma *WO8_WO1: "WO8 ⟹ WO1"*
⟨*proof*⟩

end

theory *AC7_AC9*
imports *AC_Equiv*
begin

lemma *Sigma_fun_space_not0: "⟦0∉A; B ∈ A⟧ ⟹ (nat->⋃(A)) * B ≠ 0"*
⟨*proof*⟩

lemma *inj_lemma:*
```
"C ∈ A ⟹ (λg ∈ (nat->⋃(A))*C.
        (λn ∈ nat. if(n=0, snd(g), fst(g)`(n #- 1))))
       ∈ inj((nat->⋃(A))*C, (nat->⋃(A)) ) "
```
⟨*proof*⟩

lemma *Sigma_fun_space_eqpoll:*
```
"⟦C ∈ A; 0∉A⟧ ⟹ (nat->⋃(A)) * C ≈ (nat->⋃(A))"
```
⟨*proof*⟩

lemma *AC6_AC7: "AC6 ⟹ AC7"*
⟨*proof*⟩

lemma *lemma1_1: "y ∈ (∏B ∈ A. Y*B) ⟹ (λB ∈ A. snd(y`B)) ∈ (∏B ∈*

A. B)"
⟨*proof*⟩

lemma *lemma1_2:*
*"y ∈ (∏B ∈ {Y*C. C ∈ A}. B) ⟹ (λB ∈ A. y‘(Y*B)) ∈ (∏B ∈ A. Y*B)"*
⟨*proof*⟩

lemma *AC7_AC6_lemma1:*
*"(∏B ∈ {(nat->⋃(A))*C. C ∈ A}. B) ≠ 0 ⟹ (∏B ∈ A. B) ≠ 0"*
⟨*proof*⟩

lemma *AC7_AC6_lemma2: "0 ∉ A ⟹ 0 ∉ {(nat -> ⋃(A)) * C. C ∈ A}"*
⟨*proof*⟩

lemma *AC7_AC6: "AC7 ⟹ AC6"*
⟨*proof*⟩

lemma *AC1_AC8_lemma1:*
"∀B ∈ A. ∃B1 B2. B=⟨B1,B2⟩ ∧ B1 ≈ B2
⟹ 0 ∉ { bij(fst(B),snd(B)). B ∈ A }"
⟨*proof*⟩

lemma *AC1_AC8_lemma2:*
"⟦f ∈ (∏X ∈ RepFun(A,p). X); D ∈ A⟧ ⟹ (λx ∈ A. f‘p(x))‘D ∈ p(D)"
⟨*proof*⟩

lemma *AC1_AC8: "AC1 ⟹ AC8"*
⟨*proof*⟩

lemma *AC8_AC9_lemma:*
"∀B1 ∈ A. ∀B2 ∈ A. B1 ≈ B2
*⟹ ∀B ∈ A*A. ∃B1 B2. B=⟨B1,B2⟩ ∧ B1 ≈ B2"*
⟨*proof*⟩

lemma *AC8_AC9: "AC8 ⟹ AC9"*

⟨proof⟩

lemma *snd_lepoll_SigmaI:* "b ∈ B ⟹ X ≲ B × X"
⟨proof⟩

lemma *nat_lepoll_lemma:*
"⟦0 ∉ A; B ∈ A⟧ ⟹ nat ≲ ((nat → ⋃(A)) × B) × nat"
⟨proof⟩

lemma *AC9_AC1_lemma1:*
"⟦0∉A; A≠0;
C = {((nat->⋃(A))*B)*nat. B ∈ A} ∪
{cons(0,((nat->⋃(A))*B)*nat). B ∈ A};
B1 ∈ C; B2 ∈ C⟧
⟹ B1 ≈ B2"
⟨proof⟩

lemma *AC9_AC1_lemma2:*
"∀B1 ∈ {(F*B)*N. B ∈ A} ∪ {cons(0,(F*B)*N). B ∈ A}.
∀B2 ∈ {(F*B)*N. B ∈ A} ∪ {cons(0,(F*B)*N). B ∈ A}.
f‘⟨B1,B2⟩ ∈ bij(B1, B2)
⟹ (λB ∈ A. snd(fst((f‘<cons(0,(F*B)*N),(F*B)*N>)‘0))) ∈ (∏X
∈ A. X)"
⟨proof⟩

lemma *AC9_AC1:* "AC9 ⟹ AC1"
⟨proof⟩

end

theory *WO1_AC*
imports *AC_Equiv*
begin

theorem *WO1_AC1:* "WO1 ⟹ AC1"

⟨*proof*⟩

lemma *lemma1: "⟦WO1; ∀B ∈ A. ∃C ∈ D(B). P(C,B)⟧ ⟹ ∃f. ∀B ∈ A. P(f‘B,B)"*
⟨*proof*⟩

lemma *lemma2_1: "⟦¬Finite(B); WO1⟧ ⟹ |B| + |B| ≈ B"*
⟨*proof*⟩

lemma *lemma2_2:*
"f ∈ bij(D+D, B) ⟹ {{f‘Inl(i), f‘Inr(i)}. i ∈ D} ∈ Pow(Pow(B))"
⟨*proof*⟩

lemma *lemma2_3:*
"f ∈ bij(D+D, B) ⟹ pairwise_disjoint({{f‘Inl(i), f‘Inr(i)}.
i ∈ D})"
⟨*proof*⟩

lemma *lemma2_4:*
"⟦f ∈ bij(D+D, B); 1≤n⟧
⟹ sets_of_size_between({{f‘Inl(i), f‘Inr(i)}. i ∈ D}, 2, succ(n))"
⟨*proof*⟩

lemma *lemma2_5:*
"f ∈ bij(D+D, B) ⟹ ⋃({{f‘Inl(i), f‘Inr(i)}. i ∈ D})=B"
⟨*proof*⟩

lemma *lemma2:*
"⟦WO1; ¬Finite(B); 1≤n⟧
⟹ ∃C ∈ Pow(Pow(B)). pairwise_disjoint(C) ∧
sets_of_size_between(C, 2, succ(n)) ∧
⋃(C)=B"
⟨*proof*⟩

theorem *WO1_AC10: "⟦WO1; 1≤n⟧ ⟹ AC10(n)"*
⟨*proof*⟩

end

theory *Hartog*
imports *AC_Equiv*
begin

definition

```
  Hartog :: "i ⇒ i"  where
   "Hartog(X) ≡ μ i. ¬ i ≲ X"

lemma Ords_in_set: "∀a. Ord(a) ⟶ a ∈ X ⟹ P"
⟨proof⟩

lemma Ord_lepoll_imp_ex_well_ord:
     "⟦Ord(a); a ≲ X⟧
      ⟹ ∃Y. Y ⊆ X ∧ (∃R. well_ord(Y,R) ∧ ordertype(Y,R)=a)"
  ⟨proof⟩

lemma Ord_lepoll_imp_eq_ordertype:
     "⟦Ord(a); a ≲ X⟧ ⟹ ∃Y. Y ⊆ X ∧ (∃R. R ⊆ X*X ∧ ordertype(Y,R)=a)"
⟨proof⟩

lemma Ords_lepoll_set_lemma:
     "(∀a. Ord(a) ⟶ a ≲ X) ⟹
       ∀a. Ord(a) ⟶
        a ∈ {b. Z ∈ Pow(X)*Pow(X*X), ∃Y R. Z=⟨Y,R⟩ ∧ ordertype(Y,R)=b}"
⟨proof⟩

lemma Ords_lepoll_set: "∀a. Ord(a) ⟶ a ≲ X ⟹ P"
⟨proof⟩

lemma ex_Ord_not_lepoll: "∃a. Ord(a) ∧ ¬a ≲ X"
⟨proof⟩

lemma not_Hartog_lepoll_self: "¬ Hartog(A) ≲ A"
  ⟨proof⟩

lemmas Hartog_lepoll_selfE = not_Hartog_lepoll_self [THEN notE]

lemma Ord_Hartog: "Ord(Hartog(A))"
⟨proof⟩

lemma less_HartogE1: "⟦i < Hartog(A); ¬ i ≲ A⟧ ⟹ P"
⟨proof⟩

lemma less_HartogE: "⟦i < Hartog(A); i ≈ Hartog(A)⟧ ⟹ P"
⟨proof⟩

lemma Card_Hartog: "Card(Hartog(A))"
⟨proof⟩

end


theory HH
imports AC_Equiv Hartog
```

begin

definition
  HH :: "[i, i, i] ⇒ i" **where**
    "HH(f,x,a) ≡ transrec(a, λb r. let z = x - (⋃c ∈ b. r‘c)
                                    in if f‘z ∈ Pow(z)-{0} then f‘z else {x})"

0.1 Lemmas useful in each of the three proofs

lemma *HH_def_satisfies_eq:*
    "HH(f,x,a) = (let z = x - (⋃b ∈ a. HH(f,x,b))
                   in if f‘z ∈ Pow(z)-{0} then f‘z else {x})"
⟨*proof*⟩

lemma *HH_values: "HH(f,x,a) ∈ Pow(x)-{0} | HH(f,x,a)={x}"*
⟨*proof*⟩

lemma *subset_imp_Diff_eq:*
    "B ⊆ A ⟹ X-(⋃a ∈ A. P(a)) = X-(⋃a ∈ A-B. P(a))-(⋃b ∈ B. P(b))"
⟨*proof*⟩

lemma *Ord_DiffE: "⟦c ∈ a-b; b<a⟧ ⟹ c=b | b<c ∧ c<a"*
⟨*proof*⟩

lemma *Diff_UN_eq_self: "(⋀y. y∈A ⟹ P(y) = {x}) ⟹ x - (⋃y ∈ A. P(y)) = x"*
⟨*proof*⟩

lemma *HH_eq: "x - (⋃b ∈ a. HH(f,x,b)) = x - (⋃b ∈ a1. HH(f,x,b))*
           ⟹ HH(f,x,a) = HH(f,x,a1)"
⟨*proof*⟩

lemma *HH_is_x_gt_too: "⟦HH(f,x,b)={x}; b<a⟧ ⟹ HH(f,x,a)={x}"*
⟨*proof*⟩

lemma *HH_subset_x_lt_too:*
    "⟦HH(f,x,a) ∈ Pow(x)-{0}; b<a⟧ ⟹ HH(f,x,b) ∈ Pow(x)-{0}"
⟨*proof*⟩

lemma *HH_subset_x_imp_subset_Diff_UN:*
    "HH(f,x,a) ∈ Pow(x)-{0} ⟹ HH(f,x,a) ∈ Pow(x - (⋃b ∈ a. HH(f,x,b)))-{0}"
⟨*proof*⟩

lemma *HH_eq_arg_lt:*
    "⟦HH(f,x,v)=HH(f,x,w); HH(f,x,v) ∈ Pow(x)-{0}; v ∈ w⟧ ⟹ P"
⟨*proof*⟩

lemma *HH_eq_imp_arg_eq:*

"⟦HH(f,x,v)=HH(f,x,w); HH(f,x,w) ∈ Pow(x)-{0}; Ord(v); Ord(w)⟧ ⟹ v=w"
⟨proof⟩

lemma *HH_subset_x_imp_lepoll:*
"⟦HH(f, x, i) ∈ Pow(x)-{0}; Ord(i)⟧ ⟹ i ≲ Pow(x)-{0}"
⟨proof⟩

lemma *HH_Hartog_is_x:* "HH(f, x, Hartog(Pow(x)-{0})) = {x}"
⟨proof⟩

lemma *HH_Least_eq_x:* "HH(f, x, μ i. HH(f, x, i) = {x}) = {x}"
⟨proof⟩

lemma *less_Least_subset_x:*
"a ∈ (μ i. HH(f,x,i)={x}) ⟹ HH(f,x,a) ∈ Pow(x)-{0}"
⟨proof⟩

0.2 Lemmas used in the proofs of *AC1* ⟹ *WO2* and *AC17* ⟹ *AC1*

lemma *lam_Least_HH_inj_Pow:*
"(λa ∈ (μ i. HH(f,x,i)={x}). HH(f,x,a))
∈ inj(μ i. HH(f,x,i)={x}, Pow(x)-{0})"
⟨proof⟩

lemma *lam_Least_HH_inj:*
"∀a ∈ (μ i. HH(f,x,i)={x}). ∃z ∈ x. HH(f,x,a) = {z}
⟹ (λa ∈ (μ i. HH(f,x,i)={x}). HH(f,x,a))
∈ inj(μ i. HH(f,x,i)={x}, {{y}. y ∈ x})"
⟨proof⟩

lemma *lam_surj_sing:*
"⟦x - (⋃a ∈ A. F(a)) = 0; ∀a ∈ A. ∃z ∈ x. F(a) = {z}⟧
⟹ (λa ∈ A. F(a)) ∈ surj(A, {{y}. y ∈ x})"
⟨proof⟩

lemma *not_emptyI2:* "y ∈ Pow(x)-{0} ⟹ x ≠ 0"
⟨proof⟩

lemma *f_subset_imp_HH_subset:*
"f‘(x - (⋃j ∈ i. HH(f,x,j))) ∈ Pow(x - (⋃j ∈ i. HH(f,x,j)))-{0}

⟹ HH(f, x, i) ∈ Pow(x) - {0}"
⟨proof⟩

lemma *f_subsets_imp_UN_HH_eq_x:*
"∀z ∈ Pow(x)-{0}. f‘z ∈ Pow(z)-{0}
⟹ x - (⋃j ∈ (μ i. HH(f,x,i)={x}). HH(f,x,j)) = 0"
⟨proof⟩

lemma *HH_values2:* "HH(f,x,i) = f`(x - (⋃j ∈ i. HH(f,x,j))) | HH(f,x,i)={x}"
⟨*proof*⟩

lemma *HH_subset_imp_eq:*

```
     "HH(f,x,i): Pow(x)-{0} ⟹ HH(f,x,i)=f`(x - (⋃j ∈ i. HH(f,x,j)))"
```

⟨*proof*⟩

lemma *f_sing_imp_HH_sing:*

```
     "⟦f ∈ (Pow(x)-{0}) -> {{z}. z ∈ x};
         a ∈ (μ i. HH(f,x,i)={x})⟧ ⟹ ∃z ∈ x. HH(f,x,a) = {z}"
```

⟨*proof*⟩

lemma *f_sing_lam_bij:*

```
     "⟦x - (⋃j ∈ (μ i. HH(f,x,i)={x}). HH(f,x,j)) = 0;
         f ∈ (Pow(x)-{0}) -> {{z}. z ∈ x}⟧
      ⟹ (λa ∈ (μ i. HH(f,x,i)={x}). HH(f,x,a))
           ∈ bij(μ i. HH(f,x,i)={x}, {{y}. y ∈ x})"
```

⟨*proof*⟩

lemma *lam_singI:*

```
     "f ∈ (∏X ∈ Pow(x)-{0}. F(X))
      ⟹ (λX ∈ Pow(x)-{0}. {f`X}) ∈ (∏X ∈ Pow(x)-{0}. {{z}. z ∈ F(X)})"
```

⟨*proof*⟩

lemmas *bij_Least_HH_x =*

```
    comp_bij [OF f_sing_lam_bij [OF _ lam_singI]
              lam_sing_bij [THEN bij_converse_bij]]
```

0.3 The proof of *AC1 ⟹ WO2*

lemma *bijection:*

```
     "f ∈ (∏X ∈ Pow(x) - {0}. X)
      ⟹ ∃g. g ∈ bij(x, μ i. HH(λX ∈ Pow(x)-{0}. {f`X}, x, i) = {x})"
```

⟨*proof*⟩

lemma *AC1_WO2:* "AC1 ⟹ WO2"
⟨*proof*⟩

end

theory *AC15_WO6*
imports *HH Cardinal_aux*
begin

lemma *lepoll_Sigma: "A≠0 ⟹ B ≲ A*B"*
⟨*proof*⟩

lemma *cons_times_nat_not_Finite:*
*"0∉A ⟹ ∀B ∈ {cons(0,x*nat). x ∈ A}. ¬Finite(B)"*
⟨*proof*⟩

lemma *lemma1: "⟦⋃(C)=A; a ∈ A⟧ ⟹ ∃B ∈ C. a ∈ B ∧ B ⊆ A"*
⟨*proof*⟩

lemma *lemma2:*
"⟦pairwise_disjoint(A); B ∈ A; C ∈ A; a ∈ B; a ∈ C⟧ ⟹ B=C"
⟨*proof*⟩

lemma *lemma3:*
*"∀B ∈ {cons(0, x*nat). x ∈ A}. pairwise_disjoint(f`B) ∧*
sets_of_size_between(f`B, 2, n) ∧ ⋃(f`B)=B
*⟹ ∀B ∈ A. ∃! u. u ∈ f`cons(0, B*nat) ∧ u ⊆ cons(0, B*nat) ∧*

0 ∈ u ∧ 2 ≲ u ∧ u ≲ n"
⟨*proof*⟩

lemma *lemma4: "⟦A ≲ i; Ord(i)⟧ ⟹ {P(a). a ∈ A} ≲ i"*
⟨*proof*⟩

lemma *lemma5_1:*
"⟦B ∈ A; 2 ≲ u(B)⟧ ⟹ (λx ∈ A. {fst(x). x ∈ u(x)-{0}})`B ≠ 0"
⟨*proof*⟩

lemma *lemma5_2:*
*"⟦B ∈ A; u(B) ⊆ cons(0, B*nat)⟧*
⟹ (λx ∈ A. {fst(x). x ∈ u(x)-{0}})`B ⊆ B"
⟨*proof*⟩

lemma *lemma5_3:*
"⟦n ∈ nat; B ∈ A; 0 ∈ u(B); u(B) ≲ succ(n)⟧
⟹ (λx ∈ A. {fst(x). x ∈ u(x)-{0}})`B ≲ n"
⟨*proof*⟩

lemma *ex_fun_AC13_AC15:*
*"⟦∀B ∈ {cons(0, x*nat). x ∈ A}.*
pairwise_disjoint(f`B) ∧
sets_of_size_between(f`B, 2, succ(n)) ∧ ⋃(f`B)=B;
n ∈ nat⟧

$\Longrightarrow$ ∃f. ∀B ∈ A. f‘B ≠ 0 ∧ f‘B ⊆ B ∧ f‘B ≲ n"
⟨*proof*⟩

theorem *AC10_AC11: "⟦n ∈ nat; 1≤n; AC10(n)⟧ ⟹ AC11"*
⟨*proof*⟩

theorem *AC11_AC12: "AC11 ⟹ AC12"*
⟨*proof*⟩

theorem *AC12_AC15: "AC12 ⟹ AC15"*
⟨*proof*⟩

lemma *OUN_eq_UN: "Ord(x) ⟹ (⋃a<x. F(a)) = (⋃a ∈ x. F(a))"*
⟨*proof*⟩

lemma *AC15_WO6_aux1:*
"∀x ∈ Pow(A)-{0}. f‘x≠0 ∧ f‘x ⊆ x ∧ f‘x ≲ m
⟹ (⋃i<μ x. HH(f,A,x)={A}. HH(f,A,i)) = A"
⟨*proof*⟩

lemma *AC15_WO6_aux2:*
"∀x ∈ Pow(A)-{0}. f‘x≠0 ∧ f‘x ⊆ x ∧ f‘x ≲ m
⟹ ∀x < (μ x. HH(f,A,x)={A}). HH(f,A,x) ≲ m"
⟨*proof*⟩

theorem *AC15_WO6: "AC15 ⟹ WO6"*
⟨*proof*⟩

theorem *AC10_AC13: "⟦n ∈ nat; 1≤n; AC10(n)⟧ ⟹ AC13(n)"*
⟨*proof*⟩

lemma *AC1_AC13: "AC1 ⟹ AC13(1)"*
⟨*proof*⟩

lemma *AC13_mono: "⟦m≤n; AC13(m)⟧ ⟹ AC13(n)"*
⟨*proof*⟩

theorem *AC13_AC14: "⟦n ∈ nat; 1≤n; AC13(n)⟧ ⟹ AC14"*
⟨*proof*⟩

theorem *AC14_AC15: "AC14 ⟹ AC15"*
⟨*proof*⟩

lemma *lemma_aux: "⟦A≠0; A ≲ 1⟧ ⟹ ∃a. A={a}"*
⟨*proof*⟩

lemma *AC13_AC1_lemma:*
"∀B ∈ A. f(B)≠0 ∧ f(B)<=B ∧ f(B) ≲ 1
⟹ (λx ∈ A. THE y. f(x)={y}) ∈ (∏X ∈ A. X)"
⟨*proof*⟩

theorem *AC13_AC1: "AC13(1) ⟹ AC1"*
⟨*proof*⟩

theorem *AC11_AC14: "AC11 ⟹ AC14"*
⟨*proof*⟩

end

theory *AC16_lemmas*
imports *AC_Equiv Hartog Cardinal_aux*
begin

lemma *cons_Diff_eq: "a∉A ⟹ cons(a,A)-{a}=A"*
⟨*proof*⟩

lemma *nat_1_lepoll_iff: "1≲X ⟷ (∃x. x ∈ X)"*
⟨*proof*⟩

lemma *eqpoll_1_iff_singleton: "X≈1 ⟷ (∃x. X={x})"*
⟨*proof*⟩

lemma *cons_eqpoll_succ: "⟦x≈n; y∉x⟧ ⟹ cons(y,x)≈succ(n)"*
⟨*proof*⟩

lemma *subsets_eqpoll_1_eq: "{Y ∈ Pow(X). Y≈1} = {{x}. x ∈ X}"*

⟨proof⟩

lemma *eqpoll_RepFun_sing: "X≈{{x}. x ∈ X}"*
⟨proof⟩

lemma *subsets_eqpoll_1_eqpoll: "{Y ∈ Pow(X). Y≈1}≈X"*
⟨proof⟩

lemma *InfCard_Least_in:*
"⟦InfCard(x); y ⊆ x; y ≈ succ(z)⟧ ⟹ (μ i. i ∈ y) ∈ y"
⟨proof⟩

lemma *subsets_lepoll_lemma1:*
"⟦InfCard(x); n ∈ nat⟧
⟹ {y ∈ Pow(x). y≈succ(succ(n))} ≲ x{y ∈ Pow(x). y≈succ(n)}"*
⟨proof⟩

lemma *set_of_Ord_succ_Union: "(∀y ∈ z. Ord(y)) ⟹ z ⊆ succ(⋃(z))"*
⟨proof⟩

lemma *subset_not_mem: "j ⊆ i ⟹ i ∉ j"*
⟨proof⟩

lemma *succ_Union_not_mem:*
"(⋀y. y ∈ z ⟹ Ord(y)) ⟹ succ(⋃(z)) ∉ z"
⟨proof⟩

lemma *Union_cons_eq_succ_Union:*
"⋃(cons(succ(⋃(z)),z)) = succ(⋃(z))"
⟨proof⟩

lemma *Un_Ord_disj: "⟦Ord(i); Ord(j)⟧ ⟹ i ∪ j = i | i ∪ j = j"*
⟨proof⟩

lemma *Union_eq_Un: "x ∈ X ⟹ ⋃(X) = x ∪ ⋃(X-{x})"*
⟨proof⟩

lemma *Union_in_lemma [rule_format]:*
"n ∈ nat ⟹ ∀z. (∀y ∈ z. Ord(y)) ∧ z≈n ∧ z≠0 ⟶ ⋃(z) ∈ z"
⟨proof⟩

lemma *Union_in: "⟦∀x ∈ z. Ord(x); z≈n; z≠0; n ∈ nat⟧ ⟹ ⋃(z) ∈ z"*
⟨proof⟩

lemma *succ_Union_in_x:*
"⟦InfCard(x); z ∈ Pow(x); z≈n; n ∈ nat⟧ ⟹ succ(⋃(z)) ∈ x"
⟨proof⟩

lemma *succ_lepoll_succ_succ:*

"⟦*InfCard(x); n ∈ nat*⟧
⟹ *{y ∈ Pow(x). y≈succ(n)} ≲ {y ∈ Pow(x). y≈succ(succ(n))}*"
⟨*proof*⟩

lemma *subsets_eqpoll_X:*
"⟦*InfCard(X); n ∈ nat*⟧ ⟹ *{Y ∈ Pow(X). Y≈succ(n)} ≈ X*"
⟨*proof*⟩

lemma *image_vimage_eq:*
"⟦*f ∈ surj(A,B); y ⊆ B*⟧ ⟹ *f``(converse(f)``y) = y*"
⟨*proof*⟩

lemma *vimage_image_eq:* "⟦*f ∈ inj(A,B); y ⊆ A*⟧ ⟹ *converse(f)``(f``y) = y*"
⟨*proof*⟩

lemma *subsets_eqpoll:*
"*A≈B* ⟹ *{Y ∈ Pow(A). Y≈n}≈{Y ∈ Pow(B). Y≈n}*"
⟨*proof*⟩

lemma *WO2_imp_ex_Card:* "*WO2* ⟹ *∃a. Card(a) ∧ X≈a*"
⟨*proof*⟩

lemma *lepoll_infinite:* "⟦*X≲Y; ¬Finite(X)*⟧ ⟹ *¬Finite(Y)*"
⟨*proof*⟩

lemma *infinite_Card_is_InfCard:* "⟦*¬Finite(X); Card(X)*⟧ ⟹ *InfCard(X)*"
⟨*proof*⟩

lemma *WO2_infinite_subsets_eqpoll_X:* "⟦*WO2; n ∈ nat; ¬Finite(X)*⟧
⟹ *{Y ∈ Pow(X). Y≈succ(n)}≈X*"
⟨*proof*⟩

lemma *well_ord_imp_ex_Card:* "*well_ord(X,R)* ⟹ *∃a. Card(a) ∧ X≈a*"
⟨*proof*⟩

lemma *well_ord_infinite_subsets_eqpoll_X:*
"⟦*well_ord(X,R); n ∈ nat; ¬Finite(X)*⟧ ⟹ *{Y ∈ Pow(X). Y≈succ(n)}≈X*"
⟨*proof*⟩

end

theory *WO2_AC16* **imports** *AC_Equiv AC16_lemmas Cardinal_aux* **begin**

definition
recfunAC16 :: "*[i,i,i,i] ⇒ i*" **where**

"recfunAC16(f,h,i,a) ≡
    transrec2(i, 0,
        λg r. if (∃y ∈ r. h‘g ⊆ y) then r
            else r ∪ {f‘(μ i. h‘g ⊆ f‘i ∧
                (∀b<a. (h‘b ⊆ f‘i ⟶ (∀t ∈ r. ¬ h‘b ⊆ t))))})"

lemma *recfunAC16_0: "recfunAC16(f,h,0,a) = 0"*
⟨*proof*⟩

lemma *recfunAC16_succ:*
"recfunAC16(f,h,succ(i),a) =
(if (∃y ∈ recfunAC16(f,h,i,a). h ‘ i ⊆ y) then recfunAC16(f,h,i,a)

else recfunAC16(f,h,i,a) ∪
{f ‘ (μ j. h ‘ i ⊆ f ‘ j ∧
(∀b<a. (h‘b ⊆ f‘j
⟶ (∀t ∈ recfunAC16(f,h,i,a). ¬ h‘b ⊆ t))))})"
⟨*proof*⟩

lemma *recfunAC16_Limit: "Limit(i)*
⟹ recfunAC16(f,h,i,a) = (⋃j<i. recfunAC16(f,h,j,a))"
⟨*proof*⟩

lemma *transrec2_mono_lemma [rule_format]:*
"⟦⋀g r. r ⊆ B(g,r); Ord(i)⟧
⟹ j<i ⟶ transrec2(j, 0, B) ⊆ transrec2(i, 0, B)"
⟨*proof*⟩

lemma *transrec2_mono:*
"⟦⋀g r. r ⊆ B(g,r); j≤i⟧
⟹ transrec2(j, 0, B) ⊆ transrec2(i, 0, B)"
⟨*proof*⟩

lemma *recfunAC16_mono:*
"i≤j ⟹ recfunAC16(f, g, i, a) ⊆ recfunAC16(f, g, j, a)"
⟨*proof*⟩

lemma *lemma3_1:*

```
    "⟦∀y<x. ∀z<a. z<y | (∃Y ∈ F(y). f(z)<=Y) ⟶ (∃! Y. Y ∈ F(y) ∧
f(z)<=Y);
        ∀i j. i≤j ⟶ F(i) ⊆ F(j); j≤i; i<x; z<a;
        V ∈ F(i); f(z)<=V; W ∈ F(j); f(z)<=W⟧
     ⟹ V = W"
```

⟨*proof*⟩

lemma *lemma3:*

```
    "⟦∀y<x. ∀z<a. z<y | (∃Y ∈ F(y). f(z)<=Y) ⟶ (∃! Y. Y ∈ F(y) ∧
f(z)<=Y);
        ∀i j. i≤j ⟶ F(i) ⊆ F(j); i<x; j<x; z<a;
        V ∈ F(i); f(z)<=V; W ∈ F(j); f(z)<=W⟧
     ⟹ V = W"
```

⟨*proof*⟩

lemma *lemma4:*

```
     "⟦∀y<x. F(y) ⊆ X ∧
                (∀x<a. x < y | (∃Y ∈ F(y). h(x) ⊆ Y) ⟶
                 (∃! Y. Y ∈ F(y) ∧ h(x) ⊆ Y));
         x < a⟧
      ⟹ ∀y<x. ∀z<a. z < y | (∃Y ∈ F(y). h(z) ⊆ Y) ⟶
                      (∃! Y. Y ∈ F(y) ∧ h(z) ⊆ Y)"
```

⟨*proof*⟩

lemma *lemma5:*

```
     "⟦∀y<x. F(y) ⊆ X ∧
               (∀x<a. x < y | (∃Y ∈ F(y). h(x) ⊆ Y) ⟶
                (∃! Y. Y ∈ F(y) ∧ h(x) ⊆ Y));
         x < a; Limit(x); ∀i j. i≤j ⟶ F(i) ⊆ F(j)⟧
      ⟹ (⋃x<x. F(x)) ⊆ X ∧
         (∀xa<a. xa < x | (∃x ∈ ⋃x<x. F(x). h(xa) ⊆ x)
               ⟶ (∃! Y. Y ∈ (⋃x<x. F(x)) ∧ h(xa) ⊆ Y))"
```

⟨*proof*⟩

lemma *dbl_Diff_eqpoll_Card:*
"⟦A≈a; Card(a); ¬Finite(a); B≺a; C≺a⟧ ⟹ A - B - C≈a"
⟨*proof*⟩

lemma *Finite_lesspoll_infinite_Ord:*
"⟦Finite(X); ¬Finite(a); Ord(a)⟧ ⟹ X≺a"
⟨*proof*⟩

lemma *Union_lesspoll:*
"⟦∀x ∈ X. x ≲ n ∧ x ⊆ T; well_ord(T, R); X ≲ b;
b<a; ¬Finite(a); Card(a); n ∈ nat⟧
⟹ ⋃(X)≺a"
⟨*proof*⟩

lemma *Un_sing_eq_cons:* "A ∪ {a} = cons(a, A)"
⟨*proof*⟩

lemma *Un_lepoll_succ:* "A ≲ B ⟹ A ∪ {a} ≲ succ(B)"
⟨*proof*⟩

lemma *Diff_UN_succ_empty:* "Ord(a) ⟹ F(a) - (⋃b<succ(a). F(b)) = 0"
⟨*proof*⟩

lemma *Diff_UN_succ_subset:* "Ord(a) ⟹ F(a) ∪ X - (⋃b<succ(a). F(b)) ⊆ X"
⟨*proof*⟩

lemma *recfunAC16_Diff_lepoll_1:*
"Ord(x)
⟹ recfunAC16(f, g, x, a) - (⋃i<x. recfunAC16(f, g, i, a)) ≲ 1"
⟨*proof*⟩

lemma *in_Least_Diff:*
"⟦z ∈ F(x); Ord(x)⟧
⟹ z ∈ F(μ i. z ∈ F(i)) - (⋃j<(μ i. z ∈ F(i)). F(j))"
⟨*proof*⟩

```
lemma Least_eq_imp_ex:
      "⟦(μ i. w ∈ F(i)) = (μ i. z ∈ F(i));
          w ∈ (⋃i<a. F(i)); z ∈ (⋃i<a. F(i))⟧
       ⟹ ∃b<a. w ∈ (F(b) - (⋃c<b. F(c))) ∧ z ∈ (F(b) - (⋃c<b. F(c)))"
⟨proof⟩
```

```
lemma two_in_lepoll_1: "⟦A ≲ 1; a ∈ A; b ∈ A⟧ ⟹ a=b"
⟨proof⟩
```

```
lemma UN_lepoll_index:
      "⟦∀i<a. F(i)-(⋃j<i. F(j)) ≲ 1; Limit(a)⟧
       ⟹ (⋃x<a. F(x)) ≲ a"
⟨proof⟩
```

```
lemma recfunAC16_lepoll_index: "Ord(y) ⟹ recfunAC16(f, h, y, a) ≲
y"
⟨proof⟩
```

```
lemma Union_recfunAC16_lesspoll:
      "⟦recfunAC16(f,g,y,a) ⊆ {X ∈ Pow(A). X≈n};
          A≈a;  y<a;  ¬Finite(a);  Card(a);  n ∈ nat⟧
       ⟹ ⋃(recfunAC16(f,g,y,a))≺a"
⟨proof⟩
```

```
lemma dbl_Diff_eqpoll:
      "⟦recfunAC16(f, h, y, a) ⊆ {X ∈ Pow(A) . X≈succ(k #+ m)};
          Card(a); ¬ Finite(a); A≈a;
          k ∈ nat;  y<a;
          h ∈ bij(a, {Y ∈ Pow(A). Y≈succ(k)})⟧
       ⟹ A - ⋃(recfunAC16(f, h, y, a)) - h`y≈a"
⟨proof⟩
```

```
lemmas disj_Un_eqpoll_nat_sum =
    eqpoll_trans [THEN eqpoll_trans,
                  OF disj_Un_eqpoll_sum sum_eqpoll_cong nat_sum_eqpoll_sum]
```

```
lemma Un_in_Collect: "⟦x ∈ Pow(A - B - h`i); x≈m;
        h ∈ bij(a, {x ∈ Pow(A) . x≈k}); i<a; k ∈ nat; m ∈ nat⟧
        ⟹ h ` i ∪ x ∈ {x ∈ Pow(A) . x≈k #+ m}"
⟨proof⟩
```

lemma *lemma6:*
"⟦∀y<succ(j). F(y)<=X ∧ (∀x<a. x<y | P(x,y) ⟶ Q(x,y)); succ(j)<a⟧
⟹ F(j)<=X ∧ (∀x<a. x<j | P(x,j) ⟶ Q(x,j))"
⟨*proof*⟩

lemma *lemma7:*
"⟦∀x<a. x<j | P(x,j) ⟶ Q(x,j); succ(j)<a⟧
⟹ P(j,j) ⟶ (∀x<a. x≤j | P(x,j) ⟶ Q(x,j))"
⟨*proof*⟩

lemma *ex_subset_eqpoll:*
"⟦A≈a; ¬ Finite(a); Ord(a); m ∈ nat⟧ ⟹ ∃X ∈ Pow(A). X≈m"
⟨*proof*⟩

lemma *subset_Un_disjoint:* "⟦A ⊆ B ∪ C; A ∩ C = 0⟧ ⟹ A ⊆ B"
⟨*proof*⟩

lemma *Int_empty:*
"⟦X ∈ Pow(A - ⋃(B) -C); T ∈ B; F ⊆ T⟧ ⟹ F ∩ X = 0"
⟨*proof*⟩

lemma *subset_imp_eq_lemma:*
"m ∈ nat ⟹ ∀A B. A ⊆ B ∧ m ≲ A ∧ B ≲ m ⟶ A=B"
⟨*proof*⟩

lemma *subset_imp_eq:* "⟦A ⊆ B; m ≲ A; B ≲ m; m ∈ nat⟧ ⟹ A=B"
⟨*proof*⟩

lemma *bij_imp_arg_eq:*
"⟦f ∈ bij(a, {Y ∈ X. Y≈succ(k)}); k ∈ nat; f‘b ⊆ f‘y; b<a; y<a⟧

```
      ⟹ b=y"
⟨proof⟩
```

lemma *ex_next_set:*

```
    "⟦recfunAC16(f, h, y, a) ⊆ {X ∈ Pow(A) . X≈succ(k #+ m)};
       Card(a); ¬ Finite(a); A≈a;
       k ∈ nat; m ∈ nat; y<a;
       h ∈ bij(a, {Y ∈ Pow(A). Y≈succ(k)});
       ¬ (∃Y ∈ recfunAC16(f, h, y, a). h`y ⊆ Y)⟧
      ⟹ ∃X ∈ {Y ∈ Pow(A). Y≈succ(k #+ m)}. h`y ⊆ X ∧
              (∀b<a. h`b ⊆ X ⟶
              (∀T ∈ recfunAC16(f, h, y, a). ¬ h`b ⊆ T))"
⟨proof⟩
```

lemma *ex_next_Ord:*

```
    "⟦recfunAC16(f, h, y, a) ⊆ {X ∈ Pow(A) . X≈succ(k #+ m)};
       Card(a); ¬ Finite(a); A≈a;
       k ∈ nat; m ∈ nat; y<a;
       h ∈ bij(a, {Y ∈ Pow(A). Y≈succ(k)});
       f ∈ bij(a, {Y ∈ Pow(A). Y≈succ(k #+ m)});
       ¬ (∃Y ∈ recfunAC16(f, h, y, a). h`y ⊆ Y)⟧
      ⟹ ∃c<a. h`y ⊆ f`c ∧
              (∀b<a. h`b ⊆ f`c ⟶
              (∀T ∈ recfunAC16(f, h, y, a). ¬ h`b ⊆ T))"
⟨proof⟩
```

lemma *lemma8:*

```
    "⟦∀x<a. x<j | (∃xa ∈ F(j). P(x, xa))
       ⟶ (∃! Y. Y ∈ F(j) ∧ P(x, Y)); F(j) ⊆ X;
       L ∈ X; P(j, L) ∧ (∀x<a. P(x, L) ⟶ (∀xa ∈ F(j). ¬P(x, xa)))⟧

      ⟹ F(j) ∪ {L} ⊆ X ∧
        (∀x<a. x≤j | (∃xa ∈ (F(j) ∪ {L}). P(x, xa)) ⟶
              (∃! Y. Y ∈ (F(j) ∪ {L}) ∧ P(x, Y)))"
⟨proof⟩
```

lemma *main_induct:*

```
"⟦b < a; f ∈ bij(a, {Y ∈ Pow(A) . Y≈succ(k #+ m)});
    h ∈ bij(a, {Y ∈ Pow(A) . Y≈succ(k)});
    ¬Finite(a); Card(a); A≈a; k ∈ nat; m ∈ nat⟧
 ⟹ recfunAC16(f, h, b, a) ⊆ {X ∈ Pow(A) . X≈succ(k #+ m)} ∧
     (∀x<a. x < b | (∃Y ∈ recfunAC16(f, h, b, a). h ‘ x ⊆ Y) ⟶

     (∃! Y. Y ∈ recfunAC16(f, h, b, a) ∧ h ‘ x ⊆ Y))"
```

⟨*proof*⟩

lemma *lemma_simp_induct:*

```
"⟦∀b. b<a ⟶ F(b) ⊆ S ∧ (∀x<a. (x<b | (∃Y ∈ F(b). f‘x ⊆ Y))
                                ⟶ (∃! Y. Y ∈ F(b) ∧ f‘x ⊆ Y));
    f ∈ a->f‘‘(a); Limit(a);
    ∀i j. i≤j ⟶ F(i) ⊆ F(j)⟧
   ⟹ (⋃j<a. F(j)) ⊆ S ∧
       (∀x ∈ f‘‘a. ∃! Y. Y ∈ (⋃j<a. F(j)) ∧ x ⊆ Y)"
```

⟨*proof*⟩

theorem *WO2_AC16:* "⟦WO2; 0<m; k ∈ nat; m ∈ nat⟧ ⟹ AC16(k #+ m,k)"
⟨*proof*⟩

end

theory *AC16_WO4*
imports *AC16_lemmas*
begin

lemma *lemma1:*

```
"⟦Finite(A); 0<m; m ∈ nat⟧
 ⟹ ∃a f. Ord(a) ∧ domain(f) = a ∧
```

(⋃b<a. f‘b) = A ∧ (∀b<a. f‘b ≲ m)"
⟨proof⟩

lemmas *well_ord_paired = paired_bij [THEN bij_is_inj, THEN well_ord_rvimage]*

lemma *lepoll_trans1: "⟦A ≲ B; ¬ A ≲ C⟧ ⟹ ¬ B ≲ C"*
⟨proof⟩

lemmas *lepoll_paired = paired_eqpoll [THEN eqpoll_sym, THEN eqpoll_imp_lepoll]*

lemma *lemma2: "∃y R. well_ord(y,R) ∧ x ∩ y = 0 ∧ ¬y ≲ z ∧ ¬Finite(y)"*
⟨proof⟩

lemma *infinite_Un: "¬Finite(B) ⟹ ¬Finite(A ∪ B)"*
⟨proof⟩

lemma *succ_not_lepoll_lemma:*
"⟦¬(∃x ∈ A. f‘x=y); f ∈ inj(A, B); y ∈ B⟧
⟹ (λa ∈ succ(A). if(a=A, y, f‘a)) ∈ inj(succ(A), B)"
⟨proof⟩

lemma *succ_not_lepoll_imp_eqpoll: "⟦¬A ≈ B; A ≲ B⟧ ⟹ succ(A) ≲ B"*
⟨proof⟩

lemmas *ordertype_eqpoll =*
ordermap_bij [THEN exI [THEN eqpoll_def [THEN def_imp_iff, THEN iffD2]]]

lemma *cons_cons_subset:*
"⟦a ⊆ y; b ∈ y-a; u ∈ x⟧ ⟹ cons(b, cons(u, a)) ∈ Pow(x ∪ y)"
⟨*proof*⟩

lemma *cons_cons_eqpoll:*
"⟦a ≈ k; a ⊆ y; b ∈ y-a; u ∈ x; x ∩ y = 0⟧
⟹ cons(b, cons(u, a)) ≈ succ(succ(k))"
⟨*proof*⟩

lemma *set_eq_cons:*
"⟦succ(k) ≈ A; k ≈ B; B ⊆ A; a ∈ A-B; k ∈ nat⟧ ⟹ A = cons(a, B)"
⟨*proof*⟩

lemma *cons_eqE:* "⟦cons(x,a) = cons(y,a); x ∉ a⟧ ⟹ x = y "
⟨*proof*⟩

lemma *eq_imp_Int_eq:* "A = B ⟹ A ∩ C = B ∩ C"
⟨*proof*⟩

lemma *eqpoll_sum_imp_Diff_lepoll_lemma [rule_format]:*
"⟦k ∈ nat; m ∈ nat⟧
⟹ ∀A B. A ≈ k #+ m ∧ k ≲ B ∧ B ⊆ A ⟶ A-B ≲ m"
⟨*proof*⟩

lemma *eqpoll_sum_imp_Diff_lepoll:*
"⟦A ≈ succ(k #+ m); B ⊆ A; succ(k) ≲ B; k ∈ nat; m ∈ nat⟧
⟹ A-B ≲ m"
⟨*proof*⟩

lemma *eqpoll_sum_imp_Diff_eqpoll_lemma [rule_format]:*
"⟦k ∈ nat; m ∈ nat⟧
⟹ ∀A B. A ≈ k #+ m ∧ k ≈ B ∧ B ⊆ A ⟶ A-B ≈ m"
⟨*proof*⟩

lemma *eqpoll_sum_imp_Diff_eqpoll:*
"⟦A ≈ succ(k #+ m); B ⊆ A; succ(k) ≈ B; k ∈ nat; m ∈ nat⟧
⟹ A-B ≈ m"
⟨*proof*⟩

lemma *subsets_lepoll_0_eq_unit:* "{x ∈ Pow(X). x ≲ 0} = {0}"
⟨*proof*⟩

lemma *subsets_lepoll_succ:*
"n ∈ nat ⟹ {z ∈ Pow(y). z ≲ succ(n)} =
{z ∈ Pow(y). z ≲ n} ∪ {z ∈ Pow(y). z ≈ succ(n)}"
⟨*proof*⟩

lemma *Int_empty:*
"n ∈ nat ⟹ {z ∈ Pow(y). z ≲ n} ∩ {z ∈ Pow(y). z ≈ succ(n)} = 0"
⟨*proof*⟩

locale *AC16* =
fixes x **and** y **and** k **and** l **and** m **and** t_n **and** R **and** MM **and** LL **and** GG **and** s
defines *k_def:* "k ≡ succ(l)"
and *MM_def:* "MM ≡ {v ∈ t_n. succ(k) ≲ v ∩ y}"
and *LL_def:* "LL ≡ {v ∩ y. v ∈ MM}"
and *GG_def:* "GG ≡ λv ∈ LL. (THE w. w ∈ MM ∧ v ⊆ w) - v"
and *s_def:* "s(u) ≡ {v ∈ t_n. u ∈ v ∧ k ≲ v ∩ y}"
assumes *all_ex:* "∀z ∈ {z ∈ Pow(x ∪ y) . z ≈ succ(k)}.
∃! w. w ∈ t_n ∧ z ⊆ w "
and *disjoint[iff]:* "x ∩ y = 0"
and *"includes":* "t_n ⊆ {v ∈ Pow(x ∪ y). v ≈ succ(k #+ m)}"
and *WO_R[iff]:* "well_ord(y,R)"
and *lnat[iff]:* "l ∈ nat"
and *mnat[iff]:* "m ∈ nat"
and *mpos[iff]:* "0<m"
and *Infinite[iff]:* "¬ Finite(y)"
and *noLepoll:* "¬ y ≲ {v ∈ Pow(x). v ≈ m}"
begin

lemma *knat [iff]:* "k ∈ nat"
⟨*proof*⟩

lemma *Diff_Finite_eqpoll:* "⟦l ≈ a; a ⊆ y⟧ ⟹ y - a ≈ y"
⟨*proof*⟩

lemma *s_subset:* "s(u) ⊆ t_n"
⟨*proof*⟩

lemma *sI:*
"⟦w ∈ t_n; cons(b,cons(u,a)) ⊆ w; a ⊆ y; b ∈ y-a; l ≈ a⟧
⟹ w ∈ s(u)"
⟨*proof*⟩

lemma *in_s_imp_u_in:* "v ∈ s(u) ⟹ u ∈ v"
⟨*proof*⟩

lemma *ex1_superset_a:*
"⟦l ≈ a; a ⊆ y; b ∈ y - a; u ∈ x⟧
⟹ ∃! c. c ∈ s(u) ∧ a ⊆ c ∧ b ∈ c"
⟨*proof*⟩

lemma *the_eq_cons:*
"⟦∀ v ∈ s(u). succ(l) ≈ v ∩ y;
l ≈ a; a ⊆ y; b ∈ y - a; u ∈ x⟧
⟹ (THE c. c ∈ s(u) ∧ a ⊆ c ∧ b ∈ c) ∩ y = cons(b, a)"
⟨*proof*⟩

lemma *y_lepoll_subset_s:*
"⟦∀ v ∈ s(u). succ(l) ≈ v ∩ y;
l ≈ a; a ⊆ y; u ∈ x⟧
⟹ y ≲ {v ∈ s(u). a ⊆ v}"
⟨*proof*⟩

lemma *x_imp_not_y [dest]:* "a ∈ x ⟹ a ∉ y"
⟨*proof*⟩

lemma *w_Int_eq_w_Diff:*
"w ⊆ x ∪ y ⟹ w ∩ (x - {u}) = w - cons(u, w ∩ y)"
⟨*proof*⟩

lemma *w_Int_eqpoll_m:*
"⟦w ∈ {v ∈ s(u). a ⊆ v};
l ≈ a; u ∈ x;
∀v ∈ s(u). succ(l) ≈ v ∩ y⟧
⟹ w ∩ (x - {u}) ≈ m"
⟨*proof*⟩

lemma *eqpoll_m_not_empty:* "a ≈ m ⟹ a ≠ 0"
⟨*proof*⟩

lemma *cons_cons_in:*
"⟦z ∈ xa ∩ (x - {u}); l ≈ a; a ⊆ y; u ∈ x⟧
⟹ ∃! w. w ∈ t_n ∧ cons(z, cons(u, a)) ⊆ w"
⟨*proof*⟩

lemma *subset_s_lepoll_w:*
"⟦∀v ∈ s(u). succ(l) ≈ v ∩ y; a ⊆ y; l ≈ a; u ∈ x⟧
⟹ {v ∈ s(u). a ⊆ v} ≲ {v ∈ Pow(x). v ≈ m}"
⟨*proof*⟩

lemma *well_ord_subsets_eqpoll_n:*
"n ∈ nat ⟹ ∃S. well_ord({z ∈ Pow(y) . z ≈ succ(n)}, S)"
⟨*proof*⟩

lemma *well_ord_subsets_lepoll_n:*
"n ∈ nat ⟹ ∃R. well_ord({z ∈ Pow(y). z ≲ n}, R)"
⟨*proof*⟩

lemma *LL_subset:* "LL ⊆ {z ∈ Pow(y). z ≲ succ(k #+ m)}"
⟨*proof*⟩

lemma *well_ord_LL:* "∃S. well_ord(LL,S)"
⟨*proof*⟩

lemma *unique_superset_in_MM:*
"$v \in LL \Longrightarrow \exists! \ w.\ w \in MM \wedge v \subseteq w$"
⟨*proof*⟩

lemma *Int_in_LL:* "$w \in MM \Longrightarrow w \cap y \in LL$"
⟨*proof*⟩

lemma *in_LL_eq_Int:*
"$v \in LL \Longrightarrow v = (THE\ x.\ x \in MM \wedge v \subseteq x) \cap y$"
⟨*proof*⟩

lemma *unique_superset1:* "$a \in LL \Longrightarrow (THE\ x.\ x \in MM \wedge a \subseteq x) \in MM$"
⟨*proof*⟩

lemma *the_in_MM_subset:*
"$v \in LL \Longrightarrow (THE\ x.\ x \in MM \wedge v \subseteq x) \subseteq x \cup y$"
⟨*proof*⟩

lemma *GG_subset:* "$v \in LL \Longrightarrow GG\ `\ v \subseteq x$"
⟨*proof*⟩

lemma *nat_lepoll_ordertype:* "$nat \lesssim ordertype(y, R)$"
⟨*proof*⟩

lemma *ex_subset_eqpoll_n:* "$n \in nat \Longrightarrow \exists z.\ z \subseteq y \wedge n \approx z$"
⟨*proof*⟩

lemma *exists_proper_in_s:* "$u \in x \Longrightarrow \exists v \in s(u).\ succ(k) \lesssim v \cap y$"
⟨*proof*⟩

lemma *exists_in_MM:* "$u \in x \Longrightarrow \exists w \in MM.\ u \in w$"
⟨*proof*⟩

lemma *exists_in_LL:* "$u \in x \Longrightarrow \exists w \in LL.\ u \in GG`w$"
⟨*proof*⟩

lemma *OUN_eq_x:* "$well\_ord(LL,S) \Longrightarrow$

(⋃ b<ordertype(LL,S). GG ‘ (converse(ordermap(LL,S)) ‘ b)) = x"
⟨*proof*⟩

lemma *in_MM_eqpoll_n: "w ∈ MM ⟹ w ≈ succ(k #+ m)"*
⟨*proof*⟩

lemma *in_LL_eqpoll_n: "w ∈ LL ⟹ succ(k) ≲ w"*
⟨*proof*⟩

lemma *in_LL: "w ∈ LL ⟹ w ⊆ (THE x. x ∈ MM ∧ w ⊆ x)"*
⟨*proof*⟩

lemma *all_in_lepoll_m:*
"well_ord(LL,S) ⟹
∀ b<ordertype(LL,S). GG ‘ (converse(ordermap(LL,S)) ‘ b) ≲ m"
⟨*proof*⟩

lemma *"conclusion":*
"∃ a f. Ord(a) ∧ domain(f) = a ∧ (⋃ b<a. f ‘ b) = x ∧ (∀ b<a. f ‘ b ≲ m)"
⟨*proof*⟩

end

theorem *AC16_WO4:*
"⟦AC_Equiv.AC16(k #+ m, k); 0 < k; 0 < m; k ∈ nat; m ∈ nat⟧ ⟹ WO4(m)"
⟨*proof*⟩

end

theory *AC17_AC1*
imports *HH*
begin

lemma *AC0_AC1_lemma: "⟦f:(∏ X ∈ A. X); D ⊆ A⟧ ⟹ ∃ g. g:(∏ X ∈ D.*

X)"
⟨proof⟩

lemma AC0_AC1: "AC0 ⟹ AC1"
⟨proof⟩

lemma AC1_AC0: "AC1 ⟹ AC0"
⟨proof⟩

lemma AC1_AC17_lemma: "f ∈ (∏ X ∈ Pow(A) - {0}. X) ⟹ f ∈ (Pow(A) - {0} -> A)"
⟨proof⟩

lemma AC1_AC17: "AC1 ⟹ AC17"
⟨proof⟩

lemma UN_eq_imp_well_ord:
```
"⟦x - (⋃j ∈ μ i. HH(λX ∈ Pow(x)-{0}. {f`X}, x, i) = {x}.
    HH(λX ∈ Pow(x)-{0}. {f`X}, x, j)) = 0;
    f ∈ Pow(x)-{0} -> x⟧
    ⟹ ∃r. well_ord(x,r)"
```
⟨proof⟩

lemma not_AC1_imp_ex:
```
"¬AC1 ⟹ ∃A. ∀f ∈ Pow(A)-{0} -> A. ∃u ∈ Pow(A)-{0}. f`u ∉ u"
```
⟨proof⟩

lemma AC17_AC1_aux1:
```
"⟦∀f ∈ Pow(x) - {0} -> x. ∃u ∈ Pow(x) - {0}. f`u∉u;
    ∃f ∈ Pow(x)-{0}->x.
        x - (⋃a ∈ (μ i. HH(λX ∈ Pow(x)-{0}. {f`X},x,i)={x}).
        HH(λX ∈ Pow(x)-{0}. {f`X},x,a)) = 0⟧
    ⟹ P"
```
⟨proof⟩

lemma *AC17_AC1_aux2:*
```
"¬ (∃f ∈ Pow(x)-{0}->x. x - F(f) = 0)
  ⟹ (λf ∈ Pow(x)-{0}->x . x - F(f))
      ∈ (Pow(x) -{0} -> x) -> Pow(x) - {0}"
```
⟨*proof*⟩

lemma *AC17_AC1_aux3:*
```
"⟦f‘Z ∈ Z; Z ∈ Pow(x)-{0}⟧
 ⟹ (λX ∈ Pow(x)-{0}. {f‘X})‘Z ∈ Pow(Z)-{0}"
```
⟨*proof*⟩

lemma *AC17_AC1_aux4:*
```
"∃f ∈ F. f‘((λf ∈ F. Q(f))‘f) ∈ (λf ∈ F. Q(f))‘f
 ⟹ ∃f ∈ F. f‘Q(f) ∈ Q(f)"
```
⟨*proof*⟩

lemma *AC17_AC1: "AC17 ⟹ AC1"*
⟨*proof*⟩

lemma *AC1_AC2_aux1:*
```
"⟦f:(∏X ∈ A. X);  B ∈ A;  0∉A⟧ ⟹ {f‘B} ⊆ B ∩ {f‘C. C ∈ A}"
```
⟨*proof*⟩

lemma *AC1_AC2_aux2:*
```
"⟦pairwise_disjoint(A); B ∈ A; C ∈ A; D ∈ B; D ∈ C⟧ ⟹ f‘B
= f‘C"
```
⟨*proof*⟩

lemma *AC1_AC2: "AC1 ⟹ AC2"*
⟨*proof*⟩

lemma *AC2_AC1_aux1: "0∉A ⟹ 0 ∉ {B*{B}. B ∈ A}"*
⟨*proof*⟩

lemma *AC2_AC1_aux2:*
```
"⟦X*{X} ∩ C = {y}; X ∈ A⟧
         ⟹ (THE y. X*{X} ∩ C = {y}): X*A"
```
⟨*proof*⟩

lemma *AC2_AC1_aux3:*
"∀D ∈ {E*{E}. E ∈ A}. ∃y. D ∩ C = {y}
⟹ (λx ∈ A. fst(THE z. (x*{x} ∩ C = {z}))) ∈ (∏X ∈ A. X)"
⟨*proof*⟩

lemma *AC2_AC1: "AC2 ⟹ AC1"*
⟨*proof*⟩

lemma *empty_notin_images: "0 ∉ {R``{x}. x ∈ domain(R)}"*
⟨*proof*⟩

lemma *AC1_AC4: "AC1 ⟹ AC4"*
⟨*proof*⟩

lemma *AC4_AC3_aux1: "f ∈ A->B ⟹ (⋃z ∈ A. {z}*f`z) ⊆ A*⋃(B)"*
⟨*proof*⟩

lemma *AC4_AC3_aux2: "domain(⋃z ∈ A. {z}*f(z)) = {a ∈ A. f(a)≠0}"*
⟨*proof*⟩

lemma *AC4_AC3_aux3: "x ∈ A ⟹ (⋃z ∈ A. {z}*f(z))``{x} = f(x)"*
⟨*proof*⟩

lemma *AC4_AC3: "AC4 ⟹ AC3"*
⟨*proof*⟩

lemma *AC3_AC1_lemma:*
"b∉A ⟹ (∏x ∈ {a ∈ A. id(A)`a≠b}. id(A)`x) = (∏x ∈ A. x)"
⟨*proof*⟩

lemma *AC3_AC1: "AC3 ⟹ AC1"*
⟨*proof*⟩

lemma *AC4_AC5: "AC4 ⟹ AC5"*
⟨*proof*⟩

lemma *AC5_AC4_aux1: "R ⊆ A*B ⟹ (λx ∈ R. fst(x)) ∈ R -> A"*
⟨*proof*⟩

lemma *AC5_AC4_aux2: "R ⊆ A*B ⟹ range(λx ∈ R. fst(x)) = domain(R)"*
⟨*proof*⟩

lemma *AC5_AC4_aux3: "⟦∃ f ∈ A->C. P(f,domain(f)); A=B⟧ ⟹ ∃ f ∈ B->C. P(f,B)"*
⟨*proof*⟩

lemma *AC5_AC4_aux4: "⟦R ⊆ A*B; g ∈ C->R; ∀ x ∈ C. (λz ∈ R. fst(z))‘ (g‘x) = x⟧*
⟹ (λx ∈ C. snd(g‘x)): (∏ x ∈ C. R‘‘{x})"
⟨*proof*⟩

lemma *AC5_AC4: "AC5 ⟹ AC4"*
⟨*proof*⟩

lemma *AC1_iff_AC6: "AC1 ⟷ AC6"*
⟨*proof*⟩

end

theory *AC18_AC19*
imports *AC_Equiv*
begin

definition
uu :: "i ⇒ i" **where**
"uu(a) ≡ {c ∪ {0}. c ∈ a}"

lemma *PROD_subsets:*
"⟦f ∈ (∏b ∈ {P(a). a ∈ A}. b); ∀a ∈ A. P(a)<=Q(a)⟧
⟹ (λa ∈ A. f`P(a)) ∈ (∏a ∈ A. Q(a))"
⟨*proof*⟩

lemma *lemma_AC18:*
"⟦∀A. 0 ∉ A ⟶ (∃f. f ∈ (∏X ∈ A. X)); A ≠ 0⟧
⟹ (⋂a ∈ A. ⋃b ∈ B(a). X(a, b)) ⊆
(⋃f ∈ ∏a ∈ A. B(a). ⋂a ∈ A. X(a, f`a))"
⟨*proof*⟩

lemma *AC1_AC18: "AC1 ⟹ PROP AC18"*
⟨*proof*⟩

theorem **(in** *AC18)* *AC19*
⟨*proof*⟩

lemma *RepRep_conj:*
"⟦A ≠ 0; 0 ∉ A⟧ ⟹ {uu(a). a ∈ A} ≠ 0 ∧ 0 ∉ {uu(a). a ∈ A}"
⟨*proof*⟩

lemma *lemma1_1: "⟦c ∈ a; x = c ∪ {0}; x ∉ a⟧ ⟹ x - {0} ∈ a"*
⟨*proof*⟩

lemma *lemma1_2:*
"⟦f`(uu(a)) ∉ a; f ∈ (∏B ∈ {uu(a). a ∈ A}. B); a ∈ A⟧
⟹ f`(uu(a))-{0} ∈ a"
⟨*proof*⟩

lemma *lemma1: "∃f. f ∈ (∏B ∈ {uu(a). a ∈ A}. B) ⟹ ∃f. f ∈ (∏B ∈ A. B)"*
⟨*proof*⟩

lemma *lemma2_1: "a≠0 ⟹ 0 ∈ (⋃b ∈ uu(a). b)"*
⟨*proof*⟩

lemma *lemma2: "⟦A≠0; 0∉A⟧ ⟹ (⋂x ∈ {uu(a). a ∈ A}. ⋃b ∈ x. b) ≠*

0"
⟨proof⟩

lemma *AC19_AC1: "AC19 ⟹ AC1"*
⟨proof⟩

end

theory *DC*
imports *AC_Equiv Hartog Cardinal_aux*
begin

lemma *RepFun_lepoll: "Ord(a) ⟹ {P(b). b ∈ a} ≲ a"*
⟨proof⟩

Trivial in the presence of AC, but here we need a wellordering of X

lemma *image_Ord_lepoll: "⟦f ∈ X->Y; Ord(X)⟧ ⟹ f``X ≲ X"*
⟨proof⟩

lemma *range_subset_domain:*
*"⟦R ⊆ X*X; ⋀g. g ∈ X ⟹ ∃u. ⟨g,u⟩ ∈ R⟧*
⟹ range(R) ⊆ domain(R)"
⟨proof⟩

lemma *cons_fun_type: "g ∈ n->X ⟹ cons(⟨n,x⟩, g) ∈ succ(n) -> cons(x, X)"*
⟨proof⟩

lemma *cons_fun_type2:*
"⟦g ∈ n->X; x ∈ X⟧ ⟹ cons(⟨n,x⟩, g) ∈ succ(n) -> X"
⟨proof⟩

lemma *cons_image_n: "n ∈ nat ⟹ cons(⟨n,x⟩, g)``n = g``n"*
⟨proof⟩

lemma *cons_val_n: "g ∈ n->X ⟹ cons(⟨n,x⟩, g)`n = x"*
⟨proof⟩

lemma *cons_image_k: "k ∈ n ⟹ cons(⟨n,x⟩, g)``k = g``k"*
⟨proof⟩

lemma *cons_val_k: "⟦k ∈ n; g ∈ n->X⟧ ⟹ cons(⟨n,x⟩, g)`k = g`k"*
⟨proof⟩

lemma *domain_cons_eq_succ: "domain(f)=x ⟹ domain(cons(⟨x,y⟩, f)) = succ(x)"*
⟨proof⟩

```
lemma restrict_cons_eq: "g ∈ n->X ⟹ restrict(cons(⟨n,x⟩, g), n) =
g"
⟨proof⟩

lemma succ_in_succ: "⟦Ord(k); i ∈ k⟧ ⟹ succ(i) ∈ succ(k)"
⟨proof⟩

lemma restrict_eq_imp_val_eq:
      "⟦restrict(f, domain(g)) = g; x ∈ domain(g)⟧
       ⟹ f`x = g`x"
⟨proof⟩

lemma domain_eq_imp_fun_type: "⟦domain(f)=A; f ∈ B->C⟧ ⟹ f ∈ A->C"
⟨proof⟩

lemma ex_in_domain: "⟦R ⊆ A * B; R ≠ 0⟧ ⟹ ∃x. x ∈ domain(R)"
⟨proof⟩


definition
  DC  :: "i ⇒ o"  where
    "DC(a) ≡ ∀X R. R ⊆ Pow(X)*X  ∧
                    (∀Y ∈ Pow(X). Y ≺ a ⟶ (∃x ∈ X. ⟨Y,x⟩ ∈ R))
                    ⟶ (∃f ∈ a->X. ∀b<a. <f``b,f`b> ∈ R)"

definition
  DC0 :: o  where
    "DC0 ≡ ∀A B R. R ⊆ A*B ∧ R≠0 ∧ range(R) ⊆ domain(R)
                    ⟶ (∃f ∈ nat->domain(R). ∀n ∈ nat. <f`n,f`succ(n)>:R)"

definition
  ff  :: "[i, i, i, i] ⇒ i"  where
    "ff(b, X, Q, R) ≡
           transrec(b, λc r. THE x. first(x, {x ∈ X. <r``c, x> ∈ R},
Q))"


locale DC0_imp =
  fixes XX and RR and X and R

  assumes all_ex: "∀Y ∈ Pow(X). Y ≺ nat ⟶ (∃x ∈ X. ⟨Y, x⟩ ∈ R)"

  defines XX_def: "XX ≡ (⋃n ∈ nat. {f ∈ n->X. ∀k ∈ n. <f``k, f`k> ∈
R})"
     and RR_def:  "RR ≡ {⟨z1,z2⟩:XX*XX. domain(z2)=succ(domain(z1))
                                   ∧ restrict(z2, domain(z1)) = z1}"
begin
```

lemma *lemma1_1: "RR ⊆ XX*XX"*
⟨*proof*⟩

lemma *lemma1_2: "RR ≠ 0"*
⟨*proof*⟩

lemma *lemma1_3: "range(RR) ⊆ domain(RR)"*
⟨*proof*⟩

lemma *lemma2:*
```
"⟦∀n ∈ nat. <f`n, f`succ(n)> ∈ RR;  f ∈ nat -> XX;  n ∈ nat⟧
 ⟹ ∃k ∈ nat. f`succ(n) ∈ k -> X ∧ n ∈ k
                ∧ <f`succ(n)``n, f`succ(n)`n> ∈ R"
```
⟨*proof*⟩

lemma *lemma3_1:*
```
"⟦∀n ∈ nat. <f`n, f`succ(n)> ∈ RR;  f ∈ nat -> XX;  m ∈ nat⟧
 ⟹  {f`succ(x)`x. x ∈ m} = {f`succ(m)`x. x ∈ m}"
```
⟨*proof*⟩

lemma *lemma3:*
```
"⟦∀n ∈ nat. <f`n, f`succ(n)> ∈ RR;  f ∈ nat -> XX;  m ∈ nat⟧
 ⟹ (λx ∈ nat. f`succ(x)`x) `` m = f`succ(m)``m"
```
⟨*proof*⟩

```
end

theorem DC0_imp_DC_nat: "DC0 ⟹ DC(nat)"
⟨proof⟩




lemma singleton_in_funs:
 "x ∈ X ⟹ {⟨0,x⟩} ∈
           (⋃n ∈ nat. {f ∈ succ(n)->X. ∀k ∈ n. <f`k, f`succ(k)> ∈
R})"
⟨proof⟩


locale imp_DC0 =
  fixes XX and RR and x and R and f and allRR
  defines XX_def: "XX ≡ (⋃n ∈ nat.
                      {f ∈ succ(n)->domain(R). ∀k ∈ n. <f`k, f`succ(k)>
∈ R})"
      and RR_def:
         "RR ≡ {⟨z1,z2⟩:Fin(XX)*XX.
                  (domain(z2)=succ(⋃f ∈ z1. domain(f))
                    ∧ (∀f ∈ z1. restrict(z2, domain(f)) = f))
                  | (¬ (∃g ∈ XX. domain(g)=succ(⋃f ∈ z1. domain(f))

                     ∧ (∀f ∈ z1. restrict(g, domain(f)) = f)) ∧ z2={⟨0,x⟩})}"
      and allRR_def:
        "allRR ≡ ∀b<nat.
                   <f``b, f`b> ∈
                    {⟨z1,z2⟩∈Fin(XX)*XX. (domain(z2)=succ(⋃f ∈ z1. domain(f))
                                   ∧ (⋃f ∈ z1. domain(f)) = b
                                   ∧ (∀f ∈ z1. restrict(z2,domain(f))
= f))}"
begin

lemma lemma4:
      "⟦range(R) ⊆ domain(R);  x ∈ domain(R)⟧
       ⟹ RR ⊆ Pow(XX)*XX ∧
            (∀Y ∈ Pow(XX). Y ≺ nat ⟶ (∃x ∈ XX. ⟨Y,x⟩:RR))"
⟨proof⟩

lemma UN_image_succ_eq:
      "⟦f ∈ nat->X; n ∈ nat⟧
       ⟹ (⋃x ∈ f``succ(n). P(x)) =  P(f`n) ∪ (⋃x ∈ f``n. P(x))"
⟨proof⟩

lemma UN_image_succ_eq_succ:
```

```
"⟦(⋃x ∈ f``n. P(x)) = y; P(f`n) = succ(y);
    f ∈ nat -> X; n ∈ nat⟧ ⟹ (⋃x ∈ f``succ(n). P(x)) = succ(y)"
```
⟨*proof*⟩

lemma *apply_domain_type:*
```
"⟦h ∈ succ(n) -> D;  n ∈ nat; domain(h)=succ(y)⟧ ⟹ h`y ∈ D"
```
⟨*proof*⟩

lemma *image_fun_succ:*
```
"⟦h ∈ nat -> X; n ∈ nat⟧ ⟹ h``succ(n) = cons(h`n, h``n)"
```
⟨*proof*⟩

lemma *f_n_type:*
```
"⟦domain(f`n) = succ(k); f ∈ nat -> XX;  n ∈ nat⟧
 ⟹ f`n ∈ succ(k) -> domain(R)"
```
⟨*proof*⟩

lemma *f_n_pairs_in_R [rule_format]:*
```
"⟦h ∈ nat -> XX;  domain(h`n) = succ(k);  n ∈ nat⟧
 ⟹ ∀i ∈ k. <h`n`i, h`n`succ(i)> ∈ R"
```
⟨*proof*⟩

lemma *restrict_cons_eq_restrict:*
```
"⟦restrict(h, domain(u))=u;  h ∈ n->X;  domain(u) ⊆ n⟧
 ⟹ restrict(cons(⟨n, y⟩, h), domain(u)) = u"
```
⟨*proof*⟩

lemma *all_in_image_restrict_eq:*
```
"⟦∀x ∈ f``n. restrict(f`n, domain(x))=x;
    f ∈ nat -> XX;
    n ∈ nat;  domain(f`n) = succ(n);
    (⋃x ∈ f``n. domain(x)) ⊆ n⟧
 ⟹ ∀x ∈ f``succ(n). restrict(cons(<succ(n),y>, f`n), domain(x))
= x"
```
⟨*proof*⟩

lemma *simplify_recursion:*
```
"⟦∀b<nat. <f``b, f`b> ∈ RR;
    f ∈ nat -> XX; range(R) ⊆ domain(R); x ∈ domain(R)⟧
 ⟹ allRR"
```
⟨*proof*⟩

lemma *lemma2:*
```
"⟦allRR; f ∈ nat->XX; range(R) ⊆ domain(R); x ∈ domain(R); n ∈
nat⟧
 ⟹ f`n ∈ succ(n) -> domain(R) ∧ (∀i ∈ n. <f`n`i, f`n`succ(i)>:R)"
```
⟨*proof*⟩

lemma *lemma3:*
```
"⟦allRR; f ∈ nat->XX; n∈nat; range(R) ⊆ domain(R);  x ∈ domain(R)⟧
 ⟹ f`n`n = f`succ(n)`n"
```
⟨*proof*⟩

end

theorem *DC_nat_imp_DC0:* `"DC(nat) ⟹ DC0"`
⟨*proof*⟩

lemma *fun_Ord_inj:*
```
"⟦f ∈ a->X;  Ord(a);
    ⋀b c. ⟦b<c; c ∈ a⟧ ⟹ f`b≠f`c⟧
 ⟹ f ∈ inj(a, X)"
```
⟨*proof*⟩

lemma *value_in_image:* `"⟦f ∈ X->Y; A ⊆ X; a ∈ A⟧ ⟹ f`a ∈ f``A"`
⟨*proof*⟩

lemma *lesspoll_lemma:* `"⟦¬ A ≺ B; C ≺ B⟧ ⟹ A - C ≠ 0"`
⟨*proof*⟩

theorem *DC_WO3:* `"(∀K. Card(K) ⟶ DC(K)) ⟹ WO3"`
⟨*proof*⟩

lemma *images_eq:*
```
"⟦∀x ∈ A. f`x=g`x; f ∈ Df->Cf; g ∈ Dg->Cg; A ⊆ Df; A ⊆ Dg⟧
 ⟹ f``A = g``A"
```
⟨*proof*⟩

lemma *lam_images_eq:*
```
"⟦Ord(a); b ∈ a⟧ ⟹ (λx ∈ a. h(x))``b = (λx ∈ b. h(x))``b"
```
⟨*proof*⟩

lemma *lam_type_RepFun:* `"(λb ∈ a. h(b)) ∈ a -> {h(b). b ∈ a}"`
⟨*proof*⟩

lemma *lemmaX:*
```
"⟦∀Y ∈ Pow(X). Y ≺ K ⟶ (∃x ∈ X. ⟨Y, x⟩ ∈ R);
   b ∈ K; Z ∈ Pow(X); Z ≺ K⟧
```

```
      ⟹ {x ∈ X. ⟨Z,x⟩ ∈ R} ≠ 0"
⟨proof⟩
```

lemma *WO1_DC_lemma:*

```
     "⟦Card(K); well_ord(X,Q);
        ∀Y ∈ Pow(X). Y ≺ K ⟶ (∃x ∈ X. ⟨Y, x⟩ ∈ R); b ∈ K⟧
      ⟹ ff(b, X, Q, R) ∈ {x ∈ X. <(λc ∈ b. ff(c, X, Q, R))``b, x> ∈
R}"
⟨proof⟩
```

theorem *WO1_DC_Card:* "WO1 ⟹ ∀K. Card(K) ⟶ DC(K)"
  ⟨proof⟩

end

References

[1] Lawrence C. Paulson and Krzysztof Grąbczewski. Mechanizing set theory: Cardinal arithmetic and the axiom of choice. *Journal of Automated Reasoning*, 17(3):291–323, December 1996.

[2] Herman Rubin and Jean E. Rubin. *Equivalents of the Axiom of Choice, II.* North-Holland, 1985.