Library Theorems
(DRAFT)

Rob Arthan
International Computers Limited,
Eskdale Road, Winnersh, Berks, England, RG11 5TT.
Phone: +44 734 693131,
E-mail: R.B.Jones@win0109.uucp, rda@win.icl.co.uk

Abstract

This document contains a miscellany of theorems about the objects defined in the general purpose library of HOL theories (the theories *bin_rel*, *fun_rel*, *seq*, and *fin_set*).

1 CONTENTS

2 INTRODUCTION

This document contains a miscellany of theorems about the objects defined in the general purpose library of HOL theories (the theories *bin_rel*, *fun_rel*, *seq*, and *fin_set*).

The source of the document is intended to be supplied with early ProofPower releases as example material for the users. Theorems based on the ones here, but more systematically organised may be included as part of future releases.

3 PREAMBLE

SML
```
open_theory"fin_thms";
new_theory"lib_thms";
set_pc"hol";
```

SML
```
val rel_ext_thm = save_thm("rel_ext_thm", (
push_goal([], ⌜∀ r s• r = s ⇔ (∀ x y• (x, y) ∈ r ⇔ (x, y) ∈ s)⌝);
a(REPEAT strip_tac);
(* *** Goal "1" *** *)
a(DROP_ASM_T ⌜r = s⌝ (asm_rewrite_thm_tac o eq_sym_rule));
(* *** Goal "2" *** *)
a(asm_rewrite_tac[]);
(* *** Goal "3" *** *)
a(rewrite_tac[sets_ext_clauses] THEN strip_tac);
a(LIST_SPEC_NTH_ASM_T 1 [⌜Fst x⌝, ⌜Snd x⌝] (strip_asm_tac o rewrite_rule[])
        THEN REPEAT strip_tac);
pop_thm()
));
```

SML

```
val rev_⌒_thm = save_thm("rev_⌒_thm", (
push_goal([], ⌜
                ∀list x• Rev(list ⌒ [x]) = Cons x (Rev list)
⌝);
a(strip_tac THEN list_induction_tac⌜list:'a LIST⌝
        THEN asm_rewrite_tac[rev_def, ⌒_def]);
pop_thm()
));
```

SML

```
val rev_rev_thm = save_thm("rev_rev_thm", (
push_goal([], ⌜
                ∀list• Rev(Rev list) = list
⌝);
a(strip_tac THEN list_induction_tac⌜list:'a LIST⌝
        THEN asm_rewrite_tac[rev_def, rev_⌒_thm]);
pop_thm()
));
```

SML

```
val rev_list_induction_thm = save_thm("rev_list_induction_thm", (
push_goal([], ⌜
                ∀p•     p [] ∧ (∀ list• p list ⇒ ∀ x• p (list ⌒ [x]))
                ⇒       ∀ list• p list
⌝);
a(REPEAT strip_tac);
a(lemma_tac⌜∀l•(p o Rev)l⌝);
(* *** Goal "1" *** *)
a(strip_tac THEN list_induction_tac⌜l:'a LIST⌝);
(* *** Goal "1.1" *** *)
a(asm_rewrite_tac[o_def, rev_def]);
(* *** Goal "1.2" *** *)
a(POP_ASM_T ante_tac THEN rewrite_tac[o_def, rev_def] THEN strip_tac);
a(asm_fc_tac[]);
(* *** Goal "2" *** *)
a(POP_ASM_T (accept_tac o rewrite_rule[o_def, rev_rev_thm] o ∀_elim⌜Rev list⌝));
pop_thm()
));
```

SML

```
val rev_list_induction_tac = gen_induction_tac rev_list_induction_thm;
```

SML

```
val length_⌢_thm = save_thm("length_⌢_thm", (
push_goal([], ⌜
        ∀list1 list2• Length (list1 ⌢ list2) = Length list1 + Length list2
⌝);
a(strip_tac);
a(list_induction_tac⌜list1:'a LIST⌝
        THEN asm_rewrite_tac[⌢_def, length_def, plus_assoc_thm]);
pop_thm()
));
```

SML

```
val list_rel_thm = save_thm("list_rel_thm", (
push_goal([], ⌜
        ∀list•ListRel list =
                {(i, x) | 1 ≤ i ∧ i ≤ Length list ∧ Nth list i = x}
⌝);
a(rewrite_tac [get_spec⌜ListRel⌝, get_spec⌜$..⌝,
                get_spec⌜$◁⌝, get_spec⌜Graph⌝, length_def]);
a(PC_T "hol1"(REPEAT strip_tac) THEN asm_rewrite_tac[]);
pop_thm()
));
```

SML

```
val list_rel_cons_thm = save_thm("list_rel_cons_thm", (
push_goal([], ⌜
        ListRel [] = {}
∧       ∀x list•ListRel (Cons x list) =
                {(i, y) | ∃j•j + 1 = i ∧ (j, y) ∈ ListRel list} ∪ {(1, x)}
⌝);
a(rewrite_tac[list_rel_thm, length_def, nth_def]);
a(PC_T "hol1"(REPEAT strip_tac) THEN_TRY asm_rewrite_tac[]);
(* *** Goal "1" *** *)
a(DROP_ASM_T⌜1 ≤ Fst x⌝ ante_tac THEN asm_rewrite_tac[]);
(* *** Goal "2" *** *)
a(swap_nth_asm_concl_tac 3 THEN
        CASES_T ⌜Fst x' = 1⌝ (fn th => rewrite_thm_tac th THEN strip_asm_tac th));
(* *** Goal "2.1" *** *)
a(DROP_ASM_T ⌜¬ x' = (1, x)⌝ (strip_asm_tac o once_rewrite_rule[pair_clauses]));
a(swap_nth_asm_concl_tac 1 THEN asm_rewrite_tac[]);
(* *** Goal "2.2" *** *)
a(GET_ASM_T⌜1 ≤ Fst x'⌝
        (strip_asm_tac o rewrite_rule[≤_def, ∀_elim⌜i:ℕ⌝plus_order_thm]));
a(TOP_ASM_T(rewrite_thm_tac o eq_sym_rule));
a(cases_tac ⌜¬1 ≤ i⌝);
```

```
(* *** Goal "2.2.1" *** *)
a(POP_ASM_T (strip_asm_tac o rewrite_rule[¬_≤_thm, less_def, ≤_def]));
a(DROP_ASM_T ⌜i + 1 = Fst x'⌝ante_tac THEN asm_rewrite_tac[]);
a(swap_asm_concl_tac ⌜1 ≤ Fst x'⌝);
a(GET_ASM_T⌜1 = Fst x'⌝ (strip_asm_tac o eq_sym_rule));
(* *** Goal "2.2.2" *** *)
a(swap_asm_concl_tac⌜Fst x' ≤ Length list + 1⌝);
a(GET_ASM_T⌜i + 1 = Fst x'⌝(rewrite_thm_tac o eq_sym_rule));
a(contr_tac THEN asm_fc_tac[]);
(* *** Goal "3" *** *)
a(bc_tac[≤_trans_thm] THEN ∃_tac⌜j⌝ THEN REPEAT strip_tac);
a(rewrite_tac[≤_def] THEN ∃_tac⌜1⌝ THEN asm_rewrite_tac[]);
(* *** Goal "4" *** *)
a(bc_tac[≤_trans_thm] THEN ∃_tac⌜j+1⌝ THEN REPEAT strip_tac);
a(asm_rewrite_tac[]);
(* *** Goal "5" *** *)
a(DROP_ASM_T ⌜1 ≤ j⌝ (strip_asm_tac o rewrite_rule[≤_def]));
a(LEMMA_T⌜Fst x' = (1+i)+1⌝ rewrite_thm_tac THEN asm_rewrite_tac[]);
pop_thm()
));
```

SML

```
val list_rel_⌢_thm = save_thm("list_rel_⌢_thm", (
push_goal([], ⌜
        ∀list1 list2•
                ListRel (list1 ⌢ list2)
        =       ListRel list1 ∪
                {(i, y) | ∃j•Length list1 + j= i ∧ (j, y) ∈ ListRel list2}
⌝);
a(strip_tac THEN list_induction_tac⌜list1:'a LIST⌝);
(* *** Goal "1" *** *)
a(rewrite_tac[list_rel_cons_thm, ⌢_def, length_def]);
a(conv_tac (MAP_C prove_∃_conv));
a(PC_T"hol1" (REPEAT strip_tac THEN all_asm_ante_tac THEN rewrite_tac[]));
(* *** Goal "2" *** *)
a(rewrite_tac[list_rel_cons_thm, ⌢_def, length_def]);
a(PC_T"hol1" (REPEAT strip_tac));
(* *** Goal "2.1" *** *)
a(swap_nth_asm_concl_tac 3 THEN asm_rewrite_tac[]);
a(PC_T"hol1" (REPEAT strip_tac));
(* *** Goal "2.1.1" *** *)
a(spec_nth_asm_tac 3 ⌜j⌝);
(* *** Goal "2.1.2" *** *)
a(SWAP_ASM_CONCL_T ⌜j + 1 = Fst x'⌝
        (rewrite_thm_tac o eq_sym_rule o conv_rule ¬_¬_conv));
```

```
a(spec_nth_asm_tac 2 ⌜j'⌝);
a(SWAP_ASM_CONCL_T ⌜¬ (Length list1 + 1) + j' = Fst x'⌝
        (rewrite_thm_tac o eq_sym_rule o conv_rule ¬_¬_conv));
a(rewrite_tac[plus_assoc_thm]);
(* *** Goal "2.2" *** *)
a((POP_ASM_T (strip_asm_tac o rewrite_rule[pair_clauses])));
(* *** Goal "2.3" *** *)
a(swap_nth_asm_concl_tac 2 THEN GET_NTH_ASM_T 4 rewrite_thm_tac);
a(PC_T"hol1" (REPEAT strip_tac));
a(∃_tac⌜j⌝ THEN PC_T"hol1" (REPEAT strip_tac));
(* *** Goal "2.4" *** *)
a((POP_ASM_T (strip_asm_tac o rewrite_rule[pair_clauses])));
(* *** Goal "2.5" *** *)
a(swap_nth_asm_concl_tac 2 THEN GET_NTH_ASM_T 4 rewrite_thm_tac);
a(PC_T"hol1" (REPEAT strip_tac));
a(∃_tac⌜Length list1+j⌝ THEN
        rewrite_tac[prove_rule[plus_assoc_thm]⌜(Length list1 + j)+1=(Length list1 + 1) + j⌝]
                THEN PC_T"hol1" (REPEAT strip_tac));
a(∃_tac⌜j⌝ THEN PC_T"hol1" (REPEAT strip_tac));
pop_thm()
));
```

SML

```
val list_rel_singleton_thm = save_thm("list_rel_singleton_thm", (
push_goal([], ⌜
        ∀ x•
        ListRel ([x]) = {(1, x)}
⌝);
a(rewrite_tac[list_rel_thm, length_def, nth_def]);
a(PC_T"hol1" (REPEAT strip_tac));
(* *** Goal "1" *** *)
a(swap_nth_asm_concl_tac 2);
a(lemma_tac ⌜Fst x' = 1⌝);
(* *** Goal "1.1" *** *)
a(bc_tac[≤_antisym_thm] THEN REPEAT strip_tac);
(* *** Goal "1.2" *** *)
a(asm_rewrite_tac[]);
a(swap_nth_asm_concl_tac 2);
a(asm_rewrite_tac[pair_clauses]);
(* *** Goal "2" *** *)
a(asm_rewrite_tac[]);
(* *** Goal "3" *** *)
a(asm_rewrite_tac[]);
(* *** Goal "4" *** *)
a(asm_rewrite_tac[]);
```

```
pop_thm()
));
```

SML

```
val list_rel_⌢_singleton_thm = save_thm("list_rel_⌢_singleton_thm", (
push_goal([], ⌜
        ∀list x•
        ListRel (list ⌢ [x]) = ListRel list ∪ {(Length list + 1, x)}
⌝);
a(rewrite_tac[list_rel_⌢_thm, list_rel_singleton_thm]);
a(conv_tac (MAP_C prove_∃_conv));
a(PC_T"hol1" (REPEAT strip_tac));
(* *** Goal "1" *** *)
a(POP_ASM_T (ante_tac o rewrite_rule[pair_clauses]));
a(asm_rewrite_tac[]);
(* *** Goal "2" *** *)
a(asm_rewrite_tac[]);
pop_thm()
));
```

SML

```
val dom_list_rel_thm = save_thm("dom_list_rel_thm", (
push_goal([], ⌜
        ∀list• Dom(ListRel list) = 1 .. Length list
⌝);
a(rewrite_tac[list_rel_thm, dot_dot_def, dom_def]
        THEN PC_T"hol1"(REPEAT strip_tac));
a(prove_∃_tac THEN REPEAT strip_tac);
pop_thm()
));
```

SML

```
val dom_id_ran_id_thm = save_thm("dom_id_ran_id_thm", (
push_goal([], ⌜
        ∀a• Dom(Id a) = a ∧ Ran(Id a) = a
⌝);
a(rewrite_tac[get_spec⌜Dom⌝, get_spec⌜Ran⌝, get_spec⌜Id⌝]
        THEN PC_T"hol1"(prove_tac[]));
pop_thm()
));
```

SML

```
val dot_dot_size_thm = save_thm("dot_dot_size_thm", (
push_goal([], ⌜
        ∀i j•  i .. j ∈ Finite
        ∧      #(i .. j) = if j < i then 0 else (j − i) + 1
⌝);
a(REPEAT ∀_tac);
a(lemma_tac⌜∃list•
                i .. j = Elems list
        ∧       list ∈ Distinct
        ∧       (if j < i then 0 else (j − i) + 1) = Length list⌝);
(* *** Goal "1" *** *)
a(cases_tac ⌜j < i⌝ THEN asm_rewrite_tac[]);
(* *** Goal "1.1" *** *)
a(∃_tac ⌜[]⌝);
a(POP_ASM_T (strip_asm_tac o rewrite_rule[less_def, ≤_def]));
a(POP_ASM_T (rewrite_thm_tac o eq_sym_rule));
a(rewrite_tac[plus_assoc_thm, dot_dot_def, distinct_def, length_def, elems_def]);
a(PC_T"hol1" (REPEAT strip_tac));
a(lemma_tac ⌜j + 1 + i' ≤ j⌝);
a(bc_tac[≤_trans_thm] THEN ∃_tac ⌜x⌝ THEN REPEAT strip_tac);
(* *** Goal "1.2" *** *)
a(conv_tac(ONCE_MAP_C eq_sym_conv));
a(lemma_tac⌜∃f•
                f 0 = [i]
        ∧       ∀m•f (m+1) = Cons(i+m+1)(f m)⌝ THEN1 prove_∃_tac);
a(DROP_ASM_T ⌜¬j < i⌝ (strip_asm_tac o rewrite_rule[¬_less_thm, ≤_def]));
a(POP_ASM_T (rewrite_thm_tac o eq_sym_rule));
a(∃_tac⌜f i'⌝ THEN induction_tac⌜i'⌝);
(* *** Goal "1.2.1" *** *)
a(asm_rewrite_tac[dot_dot_def, distinct_def, length_def, elems_def]);
a(PC_T"hol1" (REPEAT strip_tac) THEN_TRY asm_rewrite_tac[]);
a(swap_asm_concl_tac⌜¬ x = i⌝ THEN bc_tac[≤_antisym_thm] THEN REPEAT strip_tac);
(* *** Goal "1.2.2" *** *)
a(asm_rewrite_tac[dot_dot_def, distinct_def, length_def, elems_def]);
a(PC_T"hol1" (REPEAT strip_tac) THEN_TRY asm_rewrite_tac[]);
(* *** Goal "1.2.2.1" *** *)
a(rewrite_tac[plus_assoc_thm1] THEN rename_tac[(⌜i⌝, "ii")]);
a(bc_tac[≤_trans_thm] THEN ∃_tac⌜ii+i'⌝ THEN REPEAT strip_tac THEN rewrite_tac[]);
(* *** Goal "1.2.2.2" *** *)
a(swap_asm_concl_tac⌜x ≤ i + i' + 1⌝
        THEN rewrite_tac[plus_assoc_thm1, ≤_plus1_thm]
        THEN REPEAT strip_tac);
a(asm_rewrite_tac[plus_assoc_thm]);
(* *** Goal "2" *** *)
```

```
a(asm_rewrite_tac[fin_set_thm4]);
a(fc_tac[size_thm2]);
pop_thm()
));
```

SML

```
val enumerate_thm = save_thm("enumerate_thm", (
push_goal([], ⌜
        ∀a•     Enumerate a = {(m, n) | n ∈ a ∧ #{i | i ∈ a ∧ i ≤ n} = m}
⌝);
a(rewrite_tac[get_spec⌜Enumerate⌝, get_spec⌜$..⌝]
        THEN PC_T"hol1"(REPEAT strip_tac));
(* *** Goal "1" *** *)
a(LEMMA_T ⌜{i|i ∈ a ∧ i ≤ Snd x} = Elems l⌝ rewrite_thm_tac);
(* *** Goal "1.1" *** *)
a(PC_T"hol1"(REPEAT strip_tac) THEN asm_fc_tac[]);
a(spec_nth_asm_tac 3 ⌜x'⌝);
(* *** Goal "1.2" *** *)
a(fc_tac[size_thm2]);
a(asm_rewrite_tac[]);
(* *** Goal "2" *** *)
a(all_asm_ante_tac THEN intro_∀_tac(⌜Snd x⌝, ⌜m:ℕ⌝)
        THEN intro_∀_tac(⌜Fst x⌝, ⌜n:ℕ⌝));
a(REPEAT_N 3 strip_tac);
a(lemma_tac ⌜{i|i ∈ a ∧ i ≤ m} ∈ Finite⌝);
a(rename_tac[(⌜a⌝, "aa")] THEN bc_tac[size_thm4]);
a(∃_tac⌜0 .. m⌝ THEN rewrite_tac[dot_dot_size_thm]);
a(rewrite_tac[dot_dot_def] THEN PC_T"hol1"(REPEAT strip_tac));
(* *** Goal "2.2" *** *)
a(fc_tac[fin_set_thm1]);
a(strip_tac THEN ∃_tac⌜list⌝);
a(REPEAT strip_tac);
(* *** Goal "2.2.1" *** *)
a(fc_tac[size_thm2]);
a(swap_nth_asm_concl_tac 2 THEN asm_rewrite_tac[]);
(* *** Goal "2.2.2" *** *)
a(GET_NTH_ASM_T 3 (rewrite_thm_tac o eq_sym_rule));
a(PC_T"hol1"(REPEAT strip_tac));
pop_thm()
));
```

SML

```
val squash_id_thm = save_thm("squash_id_thm", (
push_goal([], ⌜
        ∀a• Squash(Id a) = Enumerate a
```

```
⌝);
a(rewrite_tac[get_spec⌜Squash⌝, dom_id_ran_id_thm, enumerate_thm,
        get_spec⌜$R_⨾_R⌝, get_spec⌜Id⌝]);
a(PC_T"hol1"(REPEAT strip_tac));
(* *** Goal "1" *** *)
a(POP_ASM_T (asm_rewrite_thm_tac o eq_sym_rule));
(* *** Goal "2" *** *)
a(POP_ASM_T (asm_rewrite_thm_tac o eq_sym_rule));
(* *** Goal "3" *** *)
a(∃_tac⌜Snd x⌝ THEN REPEAT strip_tac);
pop_thm()
));
```

SML

```
val squash_thm = save_thm("squash_thm", (
push_goal([], ⌜
        ∀r• Squash r = {(m, y) | ∃n• (n, y) ∈ r ∧ #{i | i ∈ Dom r ∧ i ≤ n} = m}
⌝);
a(rewrite_tac[get_spec⌜Squash⌝, dom_def, get_spec⌜$R_⨾_R⌝, enumerate_thm]);
a(PC_T"hol1"(REPEAT strip_tac));
(* *** Goal "1" *** *)
a(∃_tac⌜y⌝ THEN asm_rewrite_tac[]);
(* *** Goal "2" *** *)
a(∃_tac⌜n⌝ THEN asm_rewrite_tac[]);
a(∃_tac⌜Snd x⌝ THEN asm_rewrite_tac[]);
pop_thm()
));
```

SML

```
val ∪_▷_thm = save_thm("∪_▷_thm", (
push_goal([], ⌜
        ∀r s; a• (r ∪ s) ▷ a = (r ▷ a) ∪ (s ▷ a)
⌝);
a(rewrite_tac [▷_def]);
a(PC_T"hol1"(REPEAT strip_tac));
pop_thm()
));
```

SML

```
val dom_∪_thm = save_thm("dom_∪_thm", (
push_goal([], ⌜
        ∀r s• Dom (r ∪ s) = Dom r ∪ Dom s
⌝);
a(rewrite_tac [dom_def]);
a(PC_T"hol1"(prove_tac[]));
```

```
(* *** Goal "1" *** *)
a(∃_tac⌜y⌝ THEN REPEAT strip_tac);
(* *** Goal "2" *** *)
a(∃_tac⌜y⌝ THEN REPEAT strip_tac);
pop_thm()
));
```

The proof of the following theorem is rather long (about 120 lines). However, there is a great deal of repetition in it. The first split gives 6 subgoals. Of these, the proofs of 1 and 2 are identical and almost identical with that of 5. Similarly, the proofs of 3 and 4 are identical and only differ in the first line or two from that of 6. There may well be a way of abbreviating all of this (perhaps by more sophisticated preparation before the split), but cut-and-paste seemed to be quicker at the time.

SML

```
val enumerate_∪_thm = save_thm("enumerate_∪_thm", (
push_goal([], ⌜
∀a b• (∀i j•i ∈ a ∧ j ∈ b ⇒ i < j) ⇒
                Enumerate (a ∪ b)
        =       Enumerate a ∪ {(m, n) | n ∈ b ∧ ∃j•#a + j = m ∧ (j, n) ∈ Enumerate b}
⌝);
a(rewrite_tac [enumerate_thm]);
a(PC_T"hol1"(REPEAT strip_tac));
(* *** Goal "1" of 6 *** *)
a(lemma_tac⌜{i|i ∈ a ∪ b ∧ i ≤ Snd x} = {i|i ∈ a ∧ i ≤ Snd x}⌝);
(* *** Goal "1.1" *** *)
a(PC_T"hol1"(REPEAT strip_tac));
a(asm_fc_tac[] THEN asm_fc_tac[]);
a(swap_asm_concl_tac⌜Snd x < x'⌝ THEN asm_rewrite_tac[¬_less_thm]);
(* *** Goal "1.2" *** *)
a(LIST_DROP_NTH_ASM_T [2,3] (MAP_EVERY ante_tac) THEN asm_rewrite_tac[]
        THEN REPEAT strip_tac);
(* *** Goal "2" *** *)
a(lemma_tac⌜{i|i ∈ a ∪ b ∧ i ≤ Snd x} = {i|i ∈ a ∧ i ≤ Snd x}⌝);
(* *** Goal "2.1" *** *)
a(PC_T"hol1"(REPEAT strip_tac));
a(asm_fc_tac[] THEN asm_fc_tac[]);
a(swap_asm_concl_tac⌜Snd x < x'⌝ THEN asm_rewrite_tac[¬_less_thm]);
(* *** Goal "2.2" *** *)
a(LIST_DROP_NTH_ASM_T [2,3] (MAP_EVERY ante_tac) THEN asm_rewrite_tac[]
        THEN REPEAT strip_tac);
(* *** Goal "3" *** *)
a(prove_∃_tac);
a(DROP_NTH_ASM_T 2 (asm_rewrite_thm_tac o eq_sym_rule));
a(lemma_tac ⌜a ∈ Finite ∧ {i|i ∈ b ∧ i ≤ Snd x} ∈ Finite⌝);
(* *** Goal "3.1" *** *)
a(lemma_tac ⌜a ⊆ 0 .. Snd x ∧ {i|i ∈ b ∧ i ≤ Snd x} ⊆ 0 .. Snd x⌝);
(* *** Goal "3.1.1" *** *)
```

```
a(rewrite_tac[dot_dot_def] THEN PC_T"hol1"(REPEAT strip_tac));
a(asm_fc_tac[] THEN asm_fc_tac[]);
a(swap_asm_concl_tac⌜x' < Snd x⌝ THEN asm_rewrite_tac[¬_less_thm]);
a(fc_tac[≤_cases_thm]);
(* *** Goal "3.1.2" *** *)
a(fc_tac[once_rewrite_rule[taut_rule⌜∀p q•p ∧ q ⇔ q ∧ p⌝]size_thm4]
        THEN REPEAT strip_tac THEN fc_tac[dot_dot_size_thm]);
(* *** Goal "3.2" *** *)
a(strip_asm_tac(list_∀_elim[⌜a⌝, ⌜{i|i ∈ b ∧ i ≤ Snd x}⌝]size_thm7));
a(POP_ASM_T (rewrite_thm_tac o eq_sym_rule));
a(LEMMA_T ⌜a ∩ {i|i ∈ b ∧ i ≤ Snd x} = {}⌝ rewrite_thm_tac);
(* *** Goal "3.2.1" *** *)
a(PC_T"hol1"(REPEAT strip_tac));
a(asm_fc_tac[] THEN asm_fc_tac[]);
(* *** Goal "3.2.2" *** *)
a(LEMMA_T ⌜a ∪ {i|i ∈ b ∧ i ≤ Snd x} = {i|i ∈ a ∪ b ∧ i ≤ Snd x}⌝
        rewrite_thm_tac);
(* *** Goal "3.2.2.1" *** *)
a(PC_T"hol1"(REPEAT strip_tac));
a(asm_fc_tac[] THEN asm_fc_tac[]);
a(swap_asm_concl_tac⌜x' < Snd x⌝ THEN asm_rewrite_tac[¬_less_thm]);
a(fc_tac[≤_cases_thm]);
(* *** Goal "3.2.2.2" *** *)
a(rewrite_tac[size_thm1]);
(* *** Goal "4" *** *)
a(prove_∃_tac);
a(DROP_NTH_ASM_T 2 (asm_rewrite_thm_tac o eq_sym_rule));
a(lemma_tac ⌜a ∈ Finite ∧ {i|i ∈ b ∧ i ≤ Snd x} ∈ Finite⌝);
(* *** Goal "4.1" *** *)
a(lemma_tac ⌜a ⊆ 0 .. Snd x ∧ {i|i ∈ b ∧ i ≤ Snd x} ⊆ 0 .. Snd x⌝);
(* *** Goal "4.1.1" *** *)
a(rewrite_tac[dot_dot_def] THEN PC_T"hol1"(REPEAT strip_tac));
a(asm_fc_tac[] THEN asm_fc_tac[]);
a(swap_asm_concl_tac⌜x' < Snd x⌝ THEN asm_rewrite_tac[¬_less_thm]);
a(fc_tac[≤_cases_thm]);
(* *** Goal "4.1.2" *** *)
a(fc_tac[once_rewrite_rule[taut_rule⌜∀p q•p ∧ q ⇔ q ∧ p⌝]size_thm4]
        THEN REPEAT strip_tac THEN fc_tac[dot_dot_size_thm]);
(* *** Goal "4.2" *** *)
a(strip_asm_tac(list_∀_elim[⌜a⌝, ⌜{i|i ∈ b ∧ i ≤ Snd x}⌝]size_thm7));
a(POP_ASM_T (rewrite_thm_tac o eq_sym_rule));
a(LEMMA_T ⌜a ∩ {i|i ∈ b ∧ i ≤ Snd x} = {}⌝ rewrite_thm_tac);
(* *** Goal "4.2.1" *** *)
a(PC_T"hol1"(REPEAT strip_tac));
a(asm_fc_tac[] THEN asm_fc_tac[]);
```

```
(* *** Goal "4.2.2" *** *)
a(LEMMA_T ⌜a ∪ {i|i ∈ b ∧ i ≤ Snd x} = {i|i ∈ a ∪ b ∧ i ≤ Snd x}⌝
        rewrite_thm_tac);
(* *** Goal "4.2.2.1" *** *)
a(PC_T"hol1"(REPEAT strip_tac));
a(asm_fc_tac[] THEN asm_fc_tac[]);
a(swap_asm_concl_tac⌜x' < Snd x⌝ THEN asm_rewrite_tac[¬_less_thm]);
a(fc_tac[≤_cases_thm]);
(* *** Goal "4.2.2.2" *** *)
a(rewrite_tac[size_thm1]);
(* *** Goal "5" *** *)
a(POP_ASM_T (rewrite_thm_tac o eq_sym_rule));
a(LEMMA_T ⌜{i|i ∈ a ∧ i ≤ Snd x} = {i|i ∈ a ∪ b ∧ i ≤ Snd x}⌝
        rewrite_thm_tac);
a(PC_T"hol1"(REPEAT strip_tac));
a(asm_fc_tac[] THEN asm_fc_tac[]);
a(swap_asm_concl_tac⌜Snd x < x'⌝ THEN asm_rewrite_tac[¬_less_thm]);
(* *** Goal "6" *** *)
a(LIST_DROP_NTH_ASM_T [1,2] (rewrite_tac o map eq_sym_rule));
a(lemma_tac ⌜a ∈ Finite ∧ {i|i ∈ b ∧ i ≤ Snd x} ∈ Finite⌝);
(* *** Goal "6.1" *** *)
a(lemma_tac ⌜a ⊆ 0 .. Snd x ∧ {i|i ∈ b ∧ i ≤ Snd x} ⊆ 0 .. Snd x⌝);
(* *** Goal "6.1.1" *** *)
a(rewrite_tac[dot_dot_def] THEN PC_T"hol1"(REPEAT strip_tac));
a(asm_fc_tac[] THEN asm_fc_tac[]);
a(swap_asm_concl_tac⌜x' < Snd x⌝ THEN asm_rewrite_tac[¬_less_thm]);
a(fc_tac[≤_cases_thm]);
(* *** Goal "6.1.2" *** *)
a(fc_tac[once_rewrite_rule[taut_rule⌜∀p q•p ∧ q ⇔ q ∧ p⌝]size_thm4]
        THEN REPEAT strip_tac THEN fc_tac[dot_dot_size_thm]);
(* *** Goal "6.2" *** *)
a(strip_asm_tac(list_∀_elim[⌜a⌝, ⌜{i|i ∈ b ∧ i ≤ Snd x}⌝]size_thm7));
a(POP_ASM_T (rewrite_thm_tac o eq_sym_rule));
a(LEMMA_T ⌜a ∩ {i|i ∈ b ∧ i ≤ Snd x} = {}⌝ rewrite_thm_tac);
(* *** Goal "6.2.1" *** *)
a(PC_T"hol1"(REPEAT strip_tac));
a(asm_fc_tac[] THEN asm_fc_tac[]);
(* *** Goal "6.2.2" *** *)
a(LEMMA_T ⌜a ∪ {i|i ∈ b ∧ i ≤ Snd x} = {i|i ∈ a ∪ b ∧ i ≤ Snd x}⌝
        rewrite_thm_tac);
(* *** Goal "6.2.2.1" *** *)
a(PC_T"hol1"(REPEAT strip_tac));
a(asm_fc_tac[] THEN asm_fc_tac[]);
a(swap_asm_concl_tac⌜x' < Snd x⌝ THEN asm_rewrite_tac[¬_less_thm]);
a(fc_tac[≤_cases_thm]);
```

```
(* *** Goal "6.2.2.2" *** *)
a(rewrite_tac[size_thm1]);
pop_thm()
));
```

SML

```
val elems_set_comp_thm = save_thm("elems_set_comp_thm", (
push_goal([], ⌜
        ∀list• Elems list = {x | ∃i•1 ≤ i ∧ i ≤ Length list ∧ Nth list i = x}
⌝);
a(strip_tac);
a(list_induction_tac⌜list⌝ THEN rewrite_tac[length_def, nth_def, elems_def]);
(* *** Goal "1" *** *)
a(PC_T"hol1"(REPEAT strip_tac));
a(asm_ante_tac⌜1 ≤ i⌝ THEN asm_rewrite_tac[]);
(* *** Goal "2" *** *)
a(asm_rewrite_tac[] THEN PC_T"hol1"(REPEAT strip_tac));
(* *** Goal "2.1" *** *)
a(∃_tac⌜1⌝ THEN asm_rewrite_tac[]);
(* *** Goal "2.2" *** *)
a(∃_tac⌜i+1⌝ THEN asm_rewrite_tac[]);
a(cases_tac⌜¬i = 0⌝ THEN1 asm_rewrite_tac[]);
a(asm_ante_tac⌜1 ≤ i⌝ THEN asm_rewrite_tac[]);
(* *** Goal "2.3" *** *)
a(cases_tac ⌜i = 1⌝);
(* *** Goal "2.3.1" *** *)
a(DROP_NTH_ASM_T 3 ante_tac THEN asm_rewrite_tac[] THEN strip_tac);
a(DROP_NTH_ASM_T 3 ante_tac THEN asm_rewrite_tac[]);
(* *** Goal "2.3.2" *** *)
a(strip_asm_tac(∀_elim⌜i⌝ℕ_cases_thm));
(* *** Goal "2.3.2.1" *** *)
a(asm_ante_tac⌜1 ≤ i⌝ THEN asm_rewrite_tac[]);
(* *** Goal "2.3.2.2" *** *)
a(strip_asm_tac(∀_elim⌜i'⌝ℕ_cases_thm));
a(asm_ante_tac ⌜¬ i = 1⌝ THEN asm_rewrite_tac[]);
(* *** Goal "2.3.2.2.2" *** *)
a(DROP_NTH_ASM_T 5 ante_tac THEN asm_rewrite_tac[] THEN strip_tac);
a(∃_tac⌜i''+1⌝ THEN asm_rewrite_tac[]);
a(asm_ante_tac⌜i ≤ Length list + 1⌝ THEN asm_rewrite_tac[]);
pop_thm()
));
```